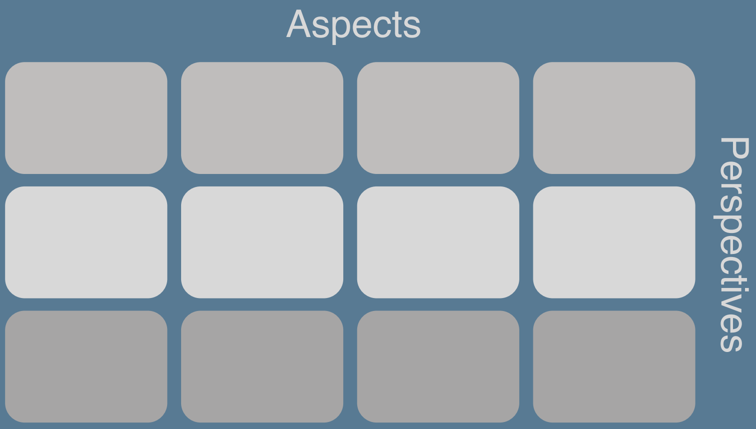
Stefan Strecker, Jürgen Jung (Eds.)

Informing Possible Future Worlds

Essays in Honour of Ulrich Frank



λογος

Informing Possible Future Worlds

Informing Possible Future Worlds

Essays in Honour of Ulrich Frank

Edited by Stefan Strecker and Jürgen Jung

Logos Verlag Berlin

Bibliographic information published by the Deutsche Nationalbibliothek

The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available on the Internet at http://dnb.d-nb.de.

Financial support for publishing and printing by the FernUniversität in Hagen and for open access publishing by the Frankfurt University of Applied Sciences is acknowledged.

Logos Verlag Berlin GmbH 2024

ISBN 978-3-8325-5768-3
DOI 10.30819/5768

Logos Verlag Berlin GmbH
Georg-Knorr-Str. 4, Geb. 10,
12681 Berlin, Germany

Tel.: +49 (0)30 / 42 85 10 90
Fax: +49 (0)30 / 42 85 10 92
https://www.logos-verlag.com

Für Ulrich Frank zu seinem 65. Geburtstag

Preface

Stefan Strecker and Jürgen Jung

For more than 35 years, Ulrich Frank has shaped Wirtschaftsinformatik as a scientific discipline through thoughtful and sophisticated research contributions and his numerous contributions to the scientific community.

Informing Possible Future Worlds is the Festschrift in honour of Ulrich Frank on the occasion of his 65th birthday. The Festschrift includes twenty-three essays written by friends, colleagues, and fellow researchers in recognition of Ulrich's contributions to Wirtschaftsinformatik research and the scientific community. Each essay is a personal and unique 'birthday present' to Ulrich Frank written exclusively for the Festschrift. From original research contributions to more personal reflections, the essays cover a wide range of topics, themes, and fields – just like Ulrich Frank's contributions.

For more than 35 years, Ulrich Frank has shaped and promoted Wirtschaftsinformatik as a scientific discipline through his thoughtful and sophisticated research contributions and his numerous contributions to the scientific community. He has initiated, engaged in and promoted scientific discourse nationally and internationally, inspired, encouraged and guided young researchers, and played an outstanding part in the Wirtschaftsinformatik community.

Starting with his Diplomarbeit in 1982 on 'Die Problematisierung von Zielbildungsprozessen in Unternehmungen durch die Betriebswirtschaftslehre unter besonderer Berücksichtigung des Verhältnisses von Wissenschaft und Praxis' (supervised by Erwin Grochla at the Universität zu Köln), Ulrich has cultivated his interest not only in Betriebswirtschaftslehre and Wirtschaftsinformatik, Software Engineering and Conceptual Modelling, but also in the Philosophy of Science, the Philosophy of Language, the Sociology of Knowledge and in Organisational Sociology. For his Diplomarbeit, he read Albert, Feyerabend, Mittelstrass, and Popper, among others, and reflected on organisational goal-setting ('Zielbildungsprozesse') from multiple, complementary perspectives – a lifelong leitmotiv he later incorporated in his Multi-Perspective Enterprise Modelling (MEMO) method.

An avid reader and bibliophile, Ulrich must have learned about Niklas Luhmann, the German sociologist working at Universität Bielefeld, Ulrich's hometown, likely in his teens, and started his own personal discourse with Luhmann's writings, and from there on expanded his readings. At about the same time, Ulrich started to program computers, developed and sold his first software application, and, most importantly, discovered Smalltalk, the programming language created by Alan Kay and others at Xerox PARC's Learning Research Group – which has strongly influenced him to this day and which he has admired ever since.

Following his discovery of Smalltalk, he took a deep dive into Object-Oriented Programming and Object-Oriented Modelling that took him to take a minor in Angewandte Informatik (Applied Informatics) at the Universität zu Köln during his graduate studies.

His doctoral research with Alfred Kieser at Universität Mannheim intensified his studies of organisations and information systems, and, in 1986, culminated in his doctoral thesis ‘Expertensysteme: Neue Automatisierungspotentiale im Büro- und Verwaltungsbereich?’ in which he investigates expert systems and studies artificial intelligence applications in information systems and their use in organisations. His postdoctoral research at the Gesellschaft für Mathematik und Datenverarbeitung (GMD) led to his Opus Magnum on ‘Multiperspektivische Unternehmensmodellierung – Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung’ accepted for his Habilitation at Universität Marburg in 1993. In 1994, Ulrich Frank became a tenured Full Professor for Wirtschaftsinformatik at the Universität Koblenz-Landau.

Starting from the foundational research on Enterprise Modelling in his Opus Magnum, he began to research and develop MEMO, a meta modelling method and family of graphical languages for Enterprise Modelling he and his research groups have refined ever since. After ten years at Koblenz-Landau, he accepted a call at Universität Duisburg-Essen (UDE) where he has held the Lehrstuhl für Wirtschaftsinformatik und Unternehmensmodellierung (Chair for Business Informatics and Enterprise Modelling) since 2004. At UDE, he has reworked MEMO, in particular his earlier versions of MEMO MML, the meta modelling language, and MEMO OrgML, a modelling language for business process and organisation modelling, and, with his research group, has developed and researched further (domain-specific) modelling languages and corresponding modelling methods. Building on his earlier research on a graphical modelling tool built in Smalltalk while at GMD, Ulrich’s research groups at Koblenz-Landau and Duisburg-Essen have developed and researched software modelling tools for Enterprise Modelling implementing graphical editors for MEMO languages to demonstrate their use and facilitate their learning.

His research on MEMO led him to realise blocking limitations in current meta modelling (and corresponding programming and implementation) approaches and led to his recent research on Multi-Level Modelling (MLM) and to him developing the Flexible Multi-Level Modelling and Execution Language (FMML^x), a meta^x modelling language and run-time environment for executable MLM and its theoretical foundations. His research on FMML^x pushes the boundaries of conventional conceptual (meta) modelling and has introduced new ways of thinking about programming as modelling and modelling as programming.

Among the manifold contributions by Ulrich Frank, only a selected few are highlighted in the constricted space of this preface not, in any way, doing him and his work justice. However, the essays in this Festschrift shed light on his contributions and deliver insight into his work. We are indebted to the authors of the contributed essays for making this Festschrift possible in the first place and, even more so, for making it such an enlightening read. Thank you all for your contribution and for the very productive collaboration.

Furthermore, we would like to thank Alexander C. Bock and Mario Nolte for their initial support and Volkhard Buchholtz at Logos Verlag, Berlin, for assistance with the production process.

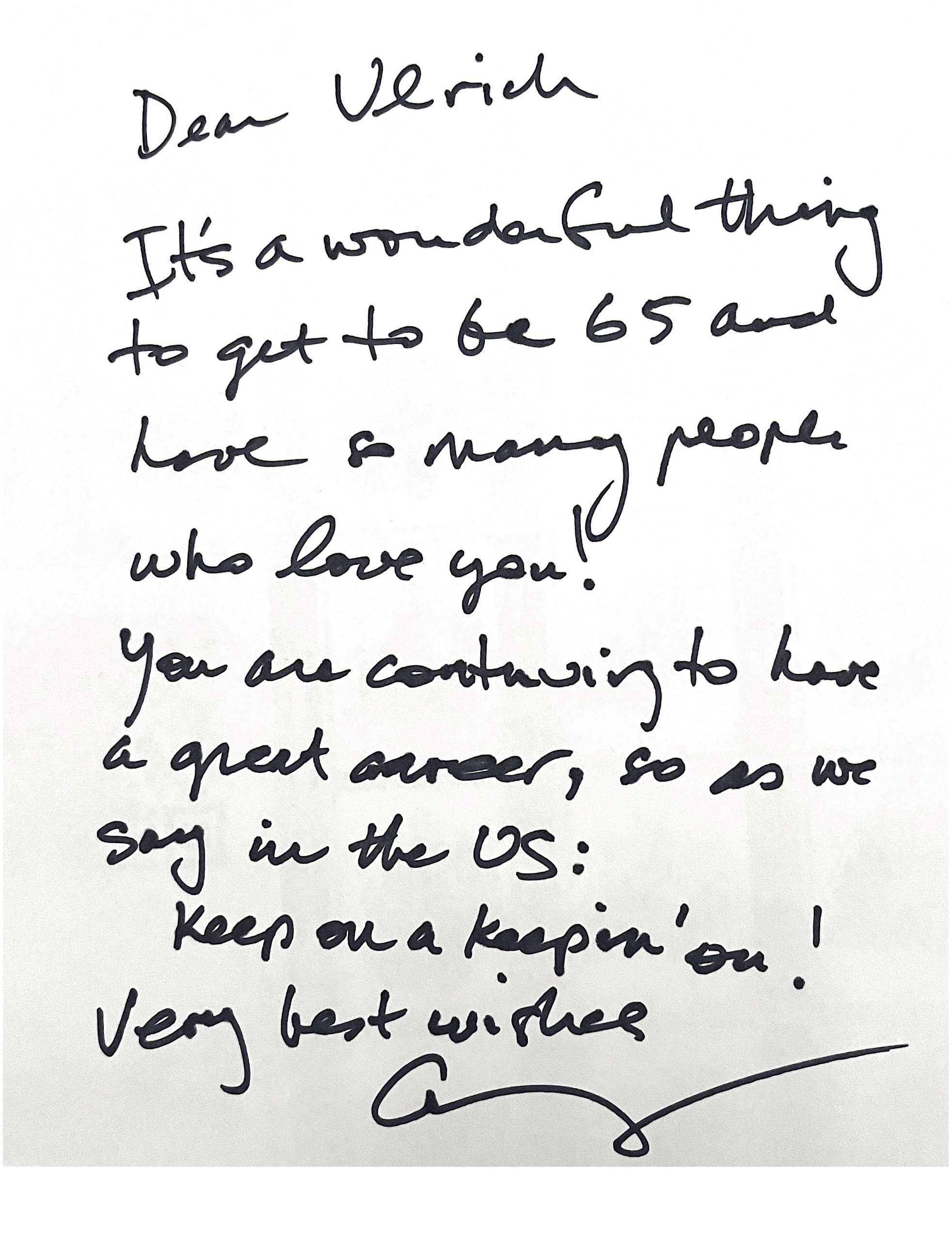
The editing and publication of Ulrich Frank’s Festschrift is a great honour for us. It is our heartfelt wish to express our deeply felt gratitude to Ulrich. Ulrich, you have been an outstanding academic mentor to both of us. This Festschrift expresses our sincere gratitude for your guidance. You have always encouraged us to embrace new challenges and provided guidance for moving forward with a high degree of commitment. Your dedication, guidance, and your continuing support has enabled us to develop and find our places in the academic community. As we know of your admiration for Smalltalk, we now have a *very special* ‘birthday present’ for you: If you turn the page, you will find Alan Kay sending you his birthday greetings in a handwritten letter he sends you via this Festschrift. Alan concludes his letter with an encouraging aphorism we see as the guiding theme for your Festschrift and for the years to come:

Keep on a keepin’ on!

On behalf of all contributors to your Festschrift, we wish you a Happy Birthday!

Stefan Strecker, FernUniversität in Hagen
Jürgen Jung, Frankfurt University of Applied Sciences
November 2023

Birthday Greetings by Alan Kay



Dear Ulrich

It's a wonderful thing to get to be 65 and have so many people who love you!

You are continuing to have a great career, so as we say in the US:

Keep on a keepin' on!

Very best wishes

[illegible]

Contents

Kapitel 1

Theorieverständnisse in Wirtschaftsinformatik und Betriebswirtschaftslehre

Stephan Zelewski, Naciye Akca und Malte L. Peters

Auf der Grundlage der zahlreichen Beiträge, die FRANK zu wissenschaftstheoretischen und auch wissenschaftssoziologischen Grundfragen der Wirtschaftsinformatik verfasst hat, wird das Theorieverständnis im „Mainstream“ von Wirtschaftsinformatik und Betriebswirtschaftslehre kritisch hinterfragt. Zur Verdeutlichung werden ein „schwaches“ und ein „starkes“ Theorieverständnis miteinander kontrastiert. Beide Theorieverständnisse werden im Rahmen des konventionellen Theorienkonzepts des „statement view“ erläutert. Zur Schärfung des „starken“ Theorieverständnisses, dem zufolge eine wissenschaftliche Theorie mindestens eine nicht-triviale nomische Hypothese umfassen muss, werden anhand des klassischen Transportkostenmodells die Schwierigkeiten aufgezeigt, die mit der Identifizierung einer solchen nomischen Hypothese verbunden sind. Zu diesem Zweck wird das klassische Transportkostenmodell als eine „Miniaturtheorie“ formalsprachlich rekonstruiert und erweitert. Die Argumentation wird in einen „Thesenstreit“ eingebettet, der von den Thesen der Theorielosigkeit und Theorieunmöglichkeit bis zur These einer starken nomisch-formalsprachlichen Theoriefundierung reicht.

1.1 Theorien in Wirtschaftsinformatik und Betriebswirtschaftslehre: ein kurzer Überblick

Zum Selbstverständnis sowohl der Wirtschaftswissenschaften im Allgemeinen als auch der Wirtschaftsinformatik und der Betriebswirtschaftslehre im Besonderen gehört oftmals der Anspruch, dass es sich um eine *theoretisch wohl-fundierte* Realwissenschaft handele.[1] Ähnlich häufig wird aber ebenso argumentiert, dass in den vorgenannten Realwissenschaften primär nicht auf eine theoretische Fundierung abgezielt werden solle, weil der Diskurs über

1 In diesem Beitrag wird – abgesehen von einigen Verweisen auf „ökonomische“ oder allgemein wirtschaftswissenschaftlich ausgerichtete Quellen – nur auf die Spezialfälle von Wirtschaftsinformatik und Betriebswirtschaftslehre als „verdeutlichende Beispiele“ explizit eingegangen. Dies ist schlicht dem Umstand geschuldet, dass sich die Verfasser vornehmlich im Bereich der Betriebswirtschaftslehre verorten (mit einer „interessierten Affinität“ zum Bereich der Wirtschaftsinformatik), sodass sie sich als nicht hinreichend kompetent fühlen, um sich zur theoretischen Fundierung der Volkswirtschaftslehre, des Wirtschaftsingenieurwesens und weiterer wirtschaftswissenschaftlicher „Bindestrich-Wissenschaften“ zu äußern.

Theorien mit „szientistischen" Vorverständnissen aus dem Bereich der Naturwissenschaften befrachtet sei, die dem „Wesen" von Wirtschaftswissenschaften einerseits sowie von Wirtschaftsinformatik und Betriebswirtschaftslehre andererseits als Kulturwissenschaften grundsätzlich fremd wären. Diese – angebliche oder tatsächliche – fundamentale Differenz zwischen Natur- und Kulturwissenschaften erinnert an die weit zurückliegende, aber nichtsdestoweniger noch immer aktuelle Debatte über die „zwei Kulturen"[2] in den (Real-) Wissenschaften.

Der zuvor skizzierte Disput im wissenschaftlichen Grundlagen- oder Basisbereich – in Anlehnung an Kuhn könnte auch von „paradigmatischen" Grundüberzeugungen[3] der betroffenen Wissenschaftsdisziplinen die Rede sein – wird noch dadurch erschwert, dass oftmals kein gemeinsames Verständnis darüber geteilt wird, was unter dem Begriff einer *wissenschaftlichen Theorie*[4] genau zu verstehen sei.[5] So kann es dazu kommen, dass zwar die Vertreter[6] einer Wissenschaftsdisziplin deren theoretische Fundierung in Anspruch nehmen, aber von Kritikern bestritten wird, dass die betroffene Disziplin überhaupt über „echte" Theorien[7] verfüge.

Vor dem Hintergrund der zuvor skizzierten Grundsatzdebatten eröffnet sich nicht nur aus wissenschaftstheoretischer, sondern auch aus wissenschaftshistorischer und vor allem wissenschaftssoziologischer Perspektive ein „vermintes" Terrain. Wer sich in den Diskurs über die theoretische Fundierung beispielsweise von Wirtschaftsinformatik oder Betriebswirtschaftslehre einmischt, muss mit Vorhaltungen (metaphorisch gesprochen: „verbalen Minen") rechnen, die ein weit gespanntes semantisches Feld umspannen: Zuweilen ist von „beklagenswerter Theorielosigkeit" von allzu einseitig auf pragmatischen Erfolg abzielenden Wissenschaftsdisziplinen die Rede.[8] Auch der Szientismusvorwurf gegenüber einem

2 Vgl. Snow (2012), S. 2 ff. Vgl. auch Reichertz (2016), S. 106 ff., insbesondere S. 107; Kühn (2002), S. 12 f. u. 39 ff., insbesondere S. 57 ff.

3 Vgl. Kuhn (2017), S. 25 ff. und passim, z. B. S. 155 ff. Vgl. auch Wolf (2020), S. 26 ff. und passim, wie z. B. S. 423 u. 425; Moulines (2008), S. 104 ff.; Gethmann (2004b), S. 33 ff.

4 Sofern Missverständnisse ausgeschlossen erscheinen, wird im Folgenden der Einfachheit halber des Öfteren nur kurz von Theorien und zugrunde liegenden Theorieverständnissen gesprochen, auch wenn strenggenommen jeweils wissenschaftliche Theorien bzw. Theorieverständnisse gemeint sind.

5 Vgl. Bichler u. a. (2016), S. 292 f., 294 („we lack a common conception of theory"), 296 f., 302 f. u. 312 f. (in Bezug auf die Wirtschaftsinformatik); Heinrich (2011), S. 231 („Im Übrigen wird ‚Theorie', wenn dieses Wort im Kontext mit dem Gegenstandsbereich der Wirtschaftsinformatik …vorkommt …eher umgangssprachlich denn fachsprachlich gemeint sein."). Vgl. auch Balzer (2009), S. 46 f.

6 In diesem Beitrag wird der „einfachen Lesbarkeit" zuliebe auf gendergerechte Formulierungsweisen verzichtet und stattdessen das generische Maskulinum verwendet. Es schließt stets sowohl weibliche Personen als auch Personen mit nicht-binärer Geschlechtswahrnehmung („divers") ein. Beispielsweise sind mit „Vertretern" sowohl Vertreterinnen als auch Vertreter sowie mit „Verfassern" sowohl Verfasserinnen als auch Verfasser gemeint.

7 Unter einer „echten" Theorie wird in diesem Beitrag stets eine Theorie verstanden, die den Begriff einer Theorie im „strengen" wissenschaftlichen Begriffsverständnis erfüllt. Was unter diesem „strengen" wissenschaftlichen Verständnis des Theoriebegriffs – oder kurz: wissenschaftlichen Theorieverständnis – konkret zu verstehen ist, darauf wird im Folgenden aus den zwei Perspektiven eines „schwachen" und eines „starken" Theorieverständnisses näher eingegangen werden.

8 Vgl. Heinrich (2005), S. 105 u. 107 ff. Zwar erstreckt sich die empirische Auswertung von Heinrich (538 Fachpublikationen in der Zeitschrift „Wirtschaftsinformatik" in den Jahren 1990 bis 2003) vornehmlich auf die Erörterung von – vor allem wissenschaftstheoretisch – fundierten Forschungsmethoden in Fachpublikationen der Wirtschaftsinformatik. Aber es finden sich in Heinrich (2005) auch einige markante, nach Einschätzung der Verfasser sogar provokante Thesen zur mangelnden Theorieverwendung in der Wirtschaftsinformatik: „Nur wenig hat sich in 14 Jahren Wirtschaftsinformatik ereignet, was wissenschaftstheoretisch und insbesondere was forschungsmethodisch bedeutsam ist. … Wirtschaftsinformatik zeigt sich also primär deskriptiv und gestaltend, kaum erklärend (und damit kaum *Theorie* bildend) … Nirgends ist von einer *Theorie* der Informationssysteme

Wissenschaftsverständnis, das sich angeblich einseitig und „unangemessen“ an naturwissenschaftlichen Vorbildern orientiert, lässt – wie bereits eingangs erwähnt – nicht lange auf sich warten. Extreme Formulierungen wie „Theorienverbohrtheit“ und „Formalisierungsfetischismus“ (vor allem in Bezug auf „harte“, formalsprachlich ausformulierte Theorien) sind zwar seltener anzutreffen, spielen aber in „hitzigen“, ad hoc geführten Debatten mitunter eine rhetorisch nicht zu unterschätzende Rolle.

In diesem Diskursumfeld hat Herr Kollege Frank, der mit der vorliegenden Festschrift für sein wissenschaftliches Lebenswerk geehrt werden soll, eine Fülle von tiefschürfenden, die innerdisziplinäre Diskussion befruchtenden und mitunter auch – im positiven Sinne – pointierten bis provokanten Beiträgen zum Wissenschaftsverständnis der Wirtschaftsinformatik verfasst. Sie können in der hier gebotenen Kürze nicht im Einzelnen erörtert werden.[9]

die Rede, von einer Theorie über die Gegenstände der Wirklichkeit ... Niemand unter den Wirtschaftsinformatikern hat bisher die Fähigkeit besessen, von der komplexen und komplizierten Wirklichkeit dieses Erkenntnisobjekts in einem Ausmaße zu *abstrahieren*, wie dies zur Entwicklung einer *Theorie* erforderlich ist.“ (S. 110, kursive Hervorhebungen durch die Verfasser) sowie „Neben der Beschreibungsaufgabe widmet sich Wirtschaftsinformatik mit erstaunlicher Intensität der Gestaltungsaufgabe ...Es ist ziemlich atemberaubend, wie sich Wirtschaftsinformatiker als Gestalter präsentieren, ohne sich darüber zu äußern, auf welchen wissenschaftlichen Erklärungen – um nicht zu sagen: Theorien – ihre Produktentwicklungen und Gestaltungsempfehlungen beruhen.“ (S. 112, kursive Hervorhebungen durch die Verfasser). Zumindest ein Mangel an „eigenständigen“ Theorien der Wirtschaftsinformatik, die nicht aus anderen Disziplinen übernommen werden, klingt in dem Überblicksbeitrag von Bichler u. a. (2016) an: „Some argue that IS needs to develop its own theories, which are distinct from reference disciplines. ... In any case, the current state of the discussion on theory in IS appears *unsatisfactory*.“ (S. 293, kursive Hervorhebungen durch die Verfasser) sowie „The question arises whether the subarea business and decision analytics in BISE involves or needs its *own theories*, or if it is sufficient to be based on theories of neighboring disciplines“ (S. 312, kursive Hervorhebungen durch die Verfasser). Vgl. ebenso Frank (2010), S. 36 („Mangel gehaltvoller Theorien“) und Frank (2001), S. 51 („Bedauerlicherweise sind Theorien, die ... generelle Zusammenhänge beschreiben und gleichzeitig bewährt sind, in der Wirtschaftsinformatik bisher kaum zu verzeichnen.“). Vgl. darüber hinaus im Sinne von „interpretierbaren“ Hinweisen beispielsweise Heinrich (2011), S. 230 (im Sinne des Fehlens einer „Theorie der Wirtschaftsinformatik“) und S. 291; Frank (2017b), S. 5728 („It seems that the debate on research methods in general, and the use of *theories* in particular, is not more than *background noise* to the ongoing reproduction of what is taken as common wisdom by many“ [kursive Hervorhebungen durch die Verfasser]).

9 Das wissenschaftlich geprägte Œuvre von Frank umfasst eine Vielfalt von Beiträgen, in denen Frank „mehr oder minder“ ausführlich und explizit erkenntnis- oder wissenschaftstheoretische Themen adressiert. Er argumentiert stets sehr prägnant und tiefgründig sowohl auf wissenschaftstheoretischer (vor allem „methodologischer“) als auch auf wissenschaftssoziologischer (in Bezug auf den „real existierenden Wissenschaftsbetrieb“) Ebene. Vgl. hierzu Frank (2021a), S. 228 ff. (u. a. auf S. 229 mit neun prägnanten Thesen zur „Einstellung“ von Frank zur Wirtschaftsinformatik als „transdisziplinärem“ Fachgebiet, die ab der vierten These auch einen deutlichen Bezug zum prononcierten Wissenschaftsverständnis von Frank aufweisen, sowie in Kapitel sechs auf S. 241 f. mit einer deutlichen Kritik am „real existierenden Wissenschaftsbetrieb“); Frank (2021b), S. 120 (philosophische Reflexionen zum aktuellen „Hype“ der „digitalen Transformation“), insbesondere S. 124 ff. (methodologische Herausforderungen, wie z. B. im Hinblick auf die Verwendung von Sprache und Wahrheitskonzepten, bis hin zu Zweifeln hinsichtlich des „Endes von Theorien“ und der epistemischen Belastbarkeit von induktiven „Inferenzen“ im Kontext von Künstlichen Neuronalen Netzen der „modernen“ KI-Forschung) sowie S. 132 ff. (vor allem hinsichtlich eines Theorieverständnisses, das in Bezug auf Zukunftsentwürfe erweitert werden soll; darauf wird in Kürze zurückgekommen); Frank und Bock (2020), S. 1 f., 5 ff., insbesondere S. 17 ff. u. 21 f. (eine bemerkenswerte Analyse der Herausforderungen, die sich aus der allseits diskutierten „digitalen Transformation“ von Wirtschaft und Verwaltung für die inter-, vor allem aber transdisziplinäre Kooperation [oder Konfrontation?] zwischen Wirtschaftsinformatik (und Informatik) einerseits sowie Betriebswirtschaftslehre (vor allem der betriebswirtschaftlichen Organisationsforschung) andererseits nicht nur aus wissenschaftstheoretischer, sondern vor allem auch aus wissenschaftssoziologischer Perspektive [wie etwa hinsichtlich der institutionellen Aufstellungen der vorgenannten Disziplinen] ergeben, mit einigen markanten Thesen, wie etwa zur „Persistenz“ einer auf ihrem historisch gewachsenen Selbstverständnis beharrenden Betriebswirtschaftslehre und zu wissenschaftstheoretischen Implikationen des Einsatzes von Produkten der Erforschung Künstlicher Intelligenz in betrieblichen Anwendungskontexten im Zusammenhang mit „Big

Stattdessen wird nur in exemplarischer Weise auf gezielte, aber sehr gehaltvolle Provokationen[10] im wissenschaftlichen Diskurs eingegangen. Dazu gehören die Sorge hinsichtlich einer „Krise“ oder zumindest „latenten Krise“[11] von Information Systems (Research) und Wirtschaftsinformatik, die Kritik an einer „Trivialisierung von Forschungsergebnissen“[12] im Kontext empirischer Forschung, die Vorbehalte gegenüber einem „theory fetish“ mit

Data“ und „Machine Learning“ (S. 18 ff.): insgesamt betrachtet, ein außerordentlich „spannender“ Beitrag, dem die Verfasser eine möglichst große Resonanz vor allem auch im Bereich der Betriebswirtschaftslehre wünschen); Frank (2019), S. 41 ff. (Analyse der digitalen Transformation, u. a. auch aus epistemologischer und methodologischer Perspektive); Frank (2017b), S. 5727 ff.; Olbrich u. a. (2017), S. 2 ff. (vor allem im Hinblick auf die problematische Replikation der Ergebnisse empirischer Studien); Bichler u. a. (2016), S. 291 ff. (mit einer detaillierten und zugleich kritischen Diskussion des Theorieverständnisses der Wirtschaftsinformatik, allerdings lassen sich die persönlichen Sichtweisen von Frank nicht klar „isolieren“, weil der Beitrag gemeinsam mit zahlreichen Koautoren verfasst wurde); Frank (2015), S. 372 ff. (eine Auseinandersetzung mit dem Modellbegriff, u. a. auch im Hinblick auf Theorien); Frank u. a. (2013), S. 279 ff. (mit einer bemerkenswerten Diskussion nicht nur der praktischen Auswirkungen [„impact“] der Wirtschaftsinformatik, sondern auch ihrer Reflexion aus der Perspektive gesellschaftlicher Verantwortung [„social responsibility“]); Frank (2014), S. 824 ff. (Reflexionen zum aktuellen Stand und „Selbstwertgefühl“ der Wirtschaftsinformatik aus wissenschaftstheoretischer und -soziologischer Perspektive); Frank u. a. (2014), S. 49 ff. (wissenschaftstheoretische, insbesondere methodische Fundierung der – beispielsweise konzeptuellen – Modellierung von Informationssystemen); Frank (2013b), S. 461 f. und Frank (2013a), S. 456 ff. (jeweils zur kritischen Reflexion der Position der Wirtschaftsinformatik hinsichtlich ihrer aktuellen Selbst- und Fremdwahrnehmung im Forschungsbetrieb); Frank (2011b), S. 117 ff. und Frank (2011a), S. 114 f. (jeweils zur Bedeutung stilisierter Fakten für die Wirtschaftsinformatik); Frank (2010), S. 35 ff. (Fundierung der Wirtschaftsinformatik hinsichtlich ihrer Forschungsmethoden); Frank (2009), S. 161 ff. (vor allem hinsichtlich eines erweiterten Theorieverständnisses in Bezug auf den Entwurf „möglicher“ – vor allem auch „besserer“ – „Welten“, auf das in Kürze zurückgekommen wird, einschließlich eines Appells an die kritische Reflexion des aktuell „Gegebenen“ seitens der Wirtschaftsinformatik, wie z. B. auf S. 167); Frank u. a. (2009), S. 3 ff. (hinsichtlich epistemologischer und methodologischer Grundlagen von Simulationen im Bereich der Sozial- und Wirtschaftswissenschaften); Frank (2008), S. 42 ff.; Frank u. a. (2008), S. 390 ff. (wissenschaftstheoretischer und -soziologischer Vergleich zwischen US-amerikanischem Information Systems (Research) und deutschsprachiger Wirtschaftsinformatik, u. a. im Hinblick auf ihre Forschungsgegenstände und Forschungsmethoden, unterstützt von institutionellen Analysen und ausführlichen Experteninterviews); Frank (2007a), S. 156 ff. (wissenschaftstheoretische Fundierung der Wirtschaftsinformatik), Frank (2007b), S. 404 f.; Frank (2006c), S. 2 ff. (mit einer kritischen Reflexion des aktuellen Stands von Information Systems [Research] und Wirtschaftsinformatik als wissenschaftlichen Disziplinen sowie ihrer philosophischen [insbesondere erkenntnistheoretischen] und methodologischen Grundlagen); Frank (2006a), S. 133 ff.; Frank (2003a), S. 278 ff. (ein engagiertes Plädoyer zur „Wiederbelebung der Wissenschaftstheorie“ im Bereich der Wirtschaftsinformatik); Frank (2002), S. 7 ff.; Frank (2001), S. 49 ff. (Verortung der Status quo der Wirtschaftsinformatik aus wissenssoziologischen und -theoretischen Blickwinkeln, vor allem auch hinsichtlich ihres Verhältnisses zu ihren Nachbardisziplinen der Betriebswirtschaftslehre und der Informatik); Frank (2000), S. 36 ff., insbesondere S. 44 ff. (wissenschaftstheoretisch fundierte Evaluation von Artefakten der Wirtschaftsinformatik); Frank (1999), S. 130 ff. (wissenschaftstheoretische und wissenschaftssoziologische Herausforderungen der Wirtschaftsinformatik, vor allem im Hinblick auf formale Sprachen in Theorien der Wirtschaftsinformatik); Frank u. a. (1999), S. 71 ff. (kritische Reflexion der Erkenntnismöglichkeiten und -probleme der Aktionsforschung für die Wirtschaftsinformatik); Frank (1998a), S. 2 ff. (Evaluation von Artefakten der Wirtschaftsinformatik); Frank (1998b), S. 98 ff.; Frank (1998c), S. 91 ff. (eine Analyse der Herausforderungen an die Disziplin „Wirtschaftsinformatik“ aus wissenschaftstheoretischer Perspektive); Frank (1997), S. 22 ff. (eine wissenschaftstheoretische Auseinandersetzung mit empirischen und „alternativen“ Forschungsmethoden der Wirtschaftsinformatik). Vgl. darüber hinaus die von Frank herausgegebenen und inhaltlich maßgeblich inspirierten Multigrafien Frank (2003b) und Frank (2004a).

10 Vgl. Frank (2017b), S. 5728 („thought provoking narratives … to launch a further attempt to challenge the dominant model of research in IS.“).

11 Vgl. Frank (2006c), S. 3 ff., insbesondere S. 4.

12 Frank (2021a), S. 240. Vgl. auch Frank (2017b), S. 5728 („… the information content of models of theories tested in IS is often systematically reduced, sometimes close to tautology, to avoid immediate refutation.“), Avison und Malaurent (2014), S. 329 f. (S. 329: „risk of triviality“); Frank (2010), S. 36 („erstaunlich triviale Erkenntnisse“); Frank (2008), S. 44; Frank (2007a), S. 165 („Trivialisierungstendenz“ [im Original kursiv hervorgehoben]; Frank (2006c), S. 24 f. Vgl. auch als weniger deutlich formulierten „Vorläufer“, mit jedoch gleicher inhaltlicher

der Einstellung „theory is king“[13], das Plädoyer für den gestaltungsorientierten „Entwurf möglicher Welten“[14] in Abgrenzung von „neo-positivistischer“ oder „behavioristischer“ Fokussierung auf das empirisch „Gegebene“ sowie die Postulate von „Muße“[15] im wissenschaftlichen „Elfenbeinturm“[16] bis hin zu einem „intellektuellen Hedonismus“[17]. Im Wesentlichen ging es Frank immer wieder um die – vor allem auch (selbst)kritische – Diskussion von methodologischen Grundsatzfragen der Wirtschaftsinformatik. Beispielsweise waren sie in die Debatte über „rigor versus relevance“[18] eingebettet,[19] von der Grundsatzdiskussionen in der Wirtschaftsinformatik in den letzten Jahren weitaus deutlicher geprägt wurden als ähnliche Diskurse in der Betriebswirtschaftslehre. Frank spannte dabei einen weiten inhaltlichen Bogen, der bis zur Konzipierung eines „Bezugsrahmens“, „Ordnungsrahmens“ oder „Frameworks“ für die „Konfigurierung von Forschungsmethoden“ reichte,[20] in dem sich unterschiedliche Wissenschaftsverständnisse nicht nur, aber insbesondere der Wirtschaftsinformatik systematisch verorten lassen.

In seinem wissenschaftstheoretischen Œuvre hat sich Frank zwar seltener unmittelbar und explizit mit der *theoretischen* Fundierung[21] der Wirtschaftsinformatik und ihrem disziplinären *Theorieverständnis*[22] auseinandergesetzt. Aber zwei wesentlichen Ausnahmen gebührt gesonderte Beachtung.

Stoßrichtung der Kritik an statistisch-methodisch aufgeblähten, aber kaum substanziell bemerkenswerten empirischen Forschungsergebnissen Frank (1997), S. 25 f.

13 Vgl. zu "‘theory fetish"’ und "‘theory is king"’ (Frank 2017b), S. 5727 u. 5735; Frank (2017a), S. 1. Vgl. auch Bichler u. a. (2016), S. 292 u. 294 sowie – distanziert – S. 307; Avison und Malaurent (2014), S. 327 (Beitragstitel), 330 u. 334 ("‘the emphasis on theory has gone too far."’).

14 Frank (2021a), S. 240. Vgl. auch Frank (2021b), S. 127, 130 u. 132. Darauf wird in Kürze zurückgekommen.

15 Vgl. Frank (2008), S. 51; Frank (2006b), S. 18; Frank (2004b), S. 30 („Muße im Sinne von Zeit für Kontemplation“); Frank (2003a), S. 289; Frank (2001), S. 52. Vgl. ebenso Freimann (1994), S. 21.

16 Vgl. Frank (2008), S. 52; Frank (2006b), S. 18; Frank (2004b), S. 31 („Kultivierung des vielgeschmähten Elfenbeinturms“); Frank (2003a), S. 289; Frank (2002), S. 9; Frank (2001), S. 52; Frank (1997), S. 33. Vgl. zur Elfenbeinturm-Metapher wissenschaftlicher Forschung auch Kuckertz (2012), S. 813 (distanziert); Freimann (1994), S. 20 f.

17 Frank (2021a), S. 242. Vgl. auch Frank (2003a), S. 290.

18 Vgl. zur Debatte über „rigor versus relevance“, die des Öfteren auch unter der Bezeichnung „bridging the relevance gap“ diskutiert wird, Chukwuere u. a. (2018), S. 31 ff., insbesondere S. 35 ff.; Flickinger u. a. (2014), S. 100 ff. (ein sehr breit angelegter Überblick, vor allem aus betriebswirtschaftlich-strategischer Perspektive); Kuckertz (2012), S. 804 ff.; Wolf und Rosenberg (2012), S. 178 ff.; Nicolai u. a. (2011), S. 54 ff.; Österle u. a. (2011), S. 8; Österle u. a. (2010), S. 665 f.; Hodgkinson und Rousseau (2009), S. 534 ff. (eine detaillierte Auseinandersetzung mit den Ausführungen in Kieser und Leiner (2009)); Kieser und Leiner (2009), S. 517 ff.; Straub und Ang (2008), S. V ff.; Gulati (2007), S. 775 ff.; Dodge u. a. (2005), S. 287 ff.; Kieser und Nicolai (2005), S. 275 ff.; Mertens (2005), S. 1738 ff. (speziell im Hinblick auf die deutsche Wirtschaftsinformatik); Baldridge u. a. (2004), S. 1063 ff.; Nicolai (2004), S. 100 ff., insbesondere S. 103 ff. u. 105 ff.; Hodgkinson u. a. (2001), S. 42 ff.

19 Vgl. Frank (2017b), S. 5727; Frank (2014), S. 824; Frank (2007a), S. 161; Frank (2007b), S. 404 f.; Frank (2006c), S. 1, 3 f. u. 24.

20 Vgl. Frank (2010), S. 38 ff.; Frank (2007a), S. 162 f. u. 170 ff., insbesondere S. 176 ff.; Frank (2006c), S. 1, 13, 22, 31 f. (Anforderungen an ein solches „Framework“), 40 ff. u. 62 f., insbesondere S. 42 ff., 44 ff. u. 48 ff. Vgl. auch die Erwähnungen in Frank (2009), S. 172; Frank (2008), S. 49.

21 Zuweilen beklagt Frank die mangelnde Fundierung von Information Systems (Research) (IS) und Wirtschaftsinformatik (WI) mittels eigener, IS- bzw. WI-spezifischer Theorien; vgl. Frank (2010), S. 36 („Mangel gehaltvoller Theorien“); Frank (2007a), S. 159 f.; Frank (2006c), S. 12. In die gleiche Richtung weist die empirische, „essayistische“ Analyse von Heinrich; vgl. Heinrich (2011), S. 230.

22 Die ausführlichste und präziseste Explikation seines Theorieverständnisses bietet Frank nach Wissen der Verfasser in Frank (2006c), S. 38; Frank (2000), S. 44 f.; Frank (1999), S. 135 f. Die dort angeführten theoriedefinierenden Merkmale entsprechen großenteils dem „schwachen“ Theorieverständnis, reichen aber teilweise auch darüber hinaus, wie z. B. das Merkmal der Minimalität. An einigen Stellen verwendet Frank den wissenschaftlichen Theoriebegriff, ohne ihn inhaltlich näher zu füllen; vgl. beispielsweise Frank (2008), S. 42; Frank (2006c), S. 3, 12 u. 24; Frank (2001), S. 50 f.; Frank (1998c), S. 100 f. Vgl. auch Bichler u. a. (2016), S. 292 f.;

Vor allem sind die neueren Überlegungen von Frank zu einem erweiterten Theorieverständnis[23] zu erwähnen. Es zielt darauf ab, Theorien nicht nur im Hinblick auf die Beschreibung und vor allem Erklärung der aktuell wahrgenommenen („faktischen") Realität zu entwerfen, sondern insbesondere – u. a. mithilfe konzeptueller Modelle[24] – auch so zu gestalten, dass sie ebenso die Entwürfe zukünftig möglicher Realitäten oder „Welten" („possible future worlds") unterstützen.[25] In dieser Hinsicht grenzt sich Frank sehr pointiert von einem konventionellen Theorieverständnis ab, das er mehrfach und sehr distanziert als „behavioristisch" und „neo-positivistisch" oder – seltener – als „szientistisch" klassifiziert.[26] Er hält diesem Theorieverständnis vor, es schränke sich einseitig auf die Rekonstruktion des Faktischen ein. Demgegenüber plädiert Frank für ein zukunftsoffenes, gestaltungsorientiertes Theorieverständnis, das in der doppelten Attribuierung durch „possible" und „future" akzentuiert wird. Auf dieses erweiterte Theorieverständnis wird im hier vorgelegten Beitrag nicht näher eingegangen.[27]

Johansson (2016), S. 189. Natürlich ist zu konzedieren, dass es in wissenschaftlichen Beiträgen sicherlich nicht nötig ist, verwendete Fachbegriffe stets näher zu definieren. An anderen Stellen nimmt Frank hinsichtlich der Klärung des Theorieverständnisses eine ambivalent anmutende Position ein. Einerseits betont Frank den großen Stellenwert, den das Theorieverständnis für eine wissenschaftliche Disziplin spielt. Vgl. beispielsweise Frank (2017b), S. 5727 („... the concept of theory which is at the core of modern science. ... First, there is the problem of defining a concept of theory that permits a clear distinction of theories and other knowledge offerings."). Andererseits wehrt er sich dagegen, auf ein präzise definiertes Theorieverständnis festgelegt zu werden. Vgl. Frank (2017b), S. 5729 („The conception of theory that I will briefly introduce now is based on two assumptions: First, it is not possible to define a concept of theory that would allow for an unambiguous distinction of theories from other knowledge contributions.") sowie ähnlich Frank (2006c), S. 38. Der Fairness halber sei eingeräumt, dass Frank auf dieses Spannungsverhältnis ausdrücklich eingeht; vgl. Frank (2017b), S. 5729.

23 Vgl. Frank (2021b), S. 132 ff. (S. 132: „extended conception of theory") sowie S. 136; Frank (2017b), S. 5728 ff. u. 5734 f.; Frank (2017a), S. 1. Begleitende Gedanken, insbesondere zu „possible worlds" oder „möglichen Welten" und zur Vergangenheitsorientierung des konventionellen (von Frank als behavioristisch oder neo-positivistisch bezeichneten) Theorieverständnisses, finden sich z. B. in Frank und Bock (2020), S. 16 ff.; Frank (2019), S. 41 f., 49 u. 52; Frank (2015), S. 374; Frank (2014), S. 826 f.; Frank (2011b), S. 119; Frank (2011a), S. 115; Frank (2009), S. 162 ff., insbesondere S. 165 ff.; Frank (2007a), S. 158 f. u. 162; Frank (2006c), S. 11, 24, 30 u. 41; Frank (2001), S. 54 f.; Frank und Schauer (2001), S. 9 f.; Frank (1997), S. 27. Vgl. auch Bichler u. a. (2016), S. 293.

24 Vgl. Frank (2017b), S. 5730 f. An dieser Stelle bleibt jedoch unklar, ob Frank konzeptuelle Modelle als eine spezielle Variante von Theorien im Sinne seines erweiterten Theorieverständnisses auffasst – oder ob konzeptuelle Modelle ein eigenständiges „epistemologisches" Konstrukt neben Theorien des erweiterten Theorieverständnisses darstellen. Die Andeutungen in Frank (2017b), S. 5733, legen nahe, dass er konzeptuelle Modelle als Bestandteile eines erweiterten Theorieverständnisses einordnet.

25 Hierbei geht es Frank vor allem darum, nicht nur in „wertneutraler" Weise mögliche zukünftige Welten als „reine" Optionen zu entwerfen, sondern auch hinsichtlich ihrer Erwünschtheit („bessere zukünftige Welten") im Hinblick auf unterschiedliche wirtschaftliche und gesellschaftliche Interessen kritisch zu beurteilen (wie sie aus betriebswirtschaftlicher Sicht von entsprechenden „Stakeholdern" artikuliert werden). Vgl. vor allem Frank (2009), S. 170 f.

26 Vgl. Frank (2021a), S. 239 ff.; Frank (2021b), S. 127 u. 130; Frank und Bock (2020), S. 21 f.; Frank (2019), S. 41; Frank (2017b), S. 5727 ff. u. 5735; Frank (2017a), S. 1; Frank (2010), S. 36 f.; Frank (2009), S. 163; Frank (2008), S. 41 f. u. 47 f.; Frank (2007a), S. 163; Frank (2006c), S. 2 ff., 6 ff. u. 30; Frank (2002), S. 7; Frank (1999), S. 140 u. 142; Frank u. a. (1999), S. 78; Frank (1998c), S. 97; Frank (1997), S. 23, 26 f. u. 32. Vgl. daneben auch den Multi-Autoren-Beitrag Bichler u. a. (2016), S. 292 f. Aus der Sicht der Verfasser lässt sich darüber diskutieren, ob die schlichte Klassifizierung eines Sachverhalts als „behavioristisch", „neo-positivistisch" oder „szientistisch" eine sachlich-neutrale Feststellung darstellt oder bereits eine pejorative (Ab-)Wertung enthält. Immerhin findet sich vor allem in Frank (2007a), S. 163 ff.; Frank (2006c), S. 22 ff.; Frank (1998c), S. 97 f. u. 100, eine inhaltliche, sehr kritische Auseinandersetzung mit behavioristischen Forschungspositionen, die weit über eine schlichte Sachverhaltsklassifizierung hinausreicht.

27 Die Verfasser möchten an dieser Stelle lediglich kritisch anmerken, dass ihnen die Notwendigkeit einer solchen Erweiterung des (wie auch immer präzisierten) Theorieverständnisses nicht unmittelbar einleuchtet. Denn eine

Daneben standen in einem interdisziplinären Forschungsseminar, das im Jahr 2017 von Frank an der Essener Fakultät für Wirtschaftswissenschaften veranstaltet wurde, u. a. die Fragen nach theoretischer Fundierung und Theorieverständnis der Wirtschaftswissenschaf-

Theorie (im hier vertretenen starken nomischen Theorieverständnis) ist wegen ihres ubiquitären (nicht nur räumlich, sondern vor allem auch zeitlich „umfassenden") Geltungsanspruchs keineswegs auf die Beschreibung und Erklärung des Vergangenen und Gegenwärtigen („Faktischen") beschränkt, sondern schließt mittels ihrer nicht-trivialen nomischen Hypothesen auch die Gesamtheit aller denkmöglichen Zukünfte ein, sofern es sich vorstellen lässt, sie auf der Grundlage der vorgenannten Hypothesen kausal zu bewirken („gestalten"). Aus diesem Grund vermögen die Verfasser nicht unmittelbar nachvollziehen, worin der prinzipielle (epistemologische?) Unterschied zwischen angeblich rückwärtsgewandten, nur die „Faktizität" reflektierenden Theorien einerseits und vorwärtsgerichteten, die Gestaltung von Zukunftsentwürfen („possible future worlds") unterstützenden Theorien andererseits liegen soll. Zumindest haben sie in den Beiträgen von Frank kein „Distinktionsmerkmal" gefunden, das es gestatten würde, Theorien „an sich" – also ohne die bewertende Interpretation ihrer vielleicht wohlwollenden oder auch missliebigen Betrachter – als rückwärtsgewandt bzw. vorwärtsgerichtet zu identifizieren. Zwar verweist z. B. Frank (2021b), S. 133, auf Erweiterungen der klassischen Logik um mehrwertige Wahrheitsprädikate jenseits des binären „tertium non datur" und die „Vielwelteninterpretation" der Quantenphysik. Aber diese Verweise sind, trotz ihrer unbestreitbaren Interessantheit, nach Einschätzung der Verfasser im Kontext des Zukunftsbezugs von Theorien nicht erforderlich. Denn auch konventionelle Theorien erweisen sich als „zukunftsoffen", weil sie alle möglichen „Zukünfte" umfassen, die sich vorstellen lassen, solange sich diese „Zukünfte" auf der Grundlage der nicht-trivialen nomischen Hypothesen einer Theorie erklären oder (instrumentell verstanden) gestalten lassen. Wegen dieser Zukunftsoffenheit schließen auch konventionelle (sowohl starke als auch schwache) Theorieverständnisse die Betrachtung von „possible future worlds", in keiner Weise aus. Aus den vorgenannten Gründen vermögen die Verfasser nicht unmittelbar nachzuvollziehen, warum in dieser Hinsicht für die theoretisch fundierte Zukunftsgestaltung – wie von Frank postuliert – ein grundsätzlich *erweitertes* Theorieverständnis erforderlich sein sollte.

Zwar könnte Frank darauf abzielen, dass Theorien im Hinblick auf die Gestaltung möglicher Zukünfte „offener" formuliert sein sollten als Theorien, die „nur" auf die Beschreibung oder Erklärung von Faktischem (Vergangenem und Gegenwärtigem) abzielen. Aber eine solche Unterschiedlichkeit zwischen einerseits zukunftsorientierten und andererseits vergangenheits- und gegenwartsfokussierten Theorien ist den Verfassern nicht bekannt. Sie sollte, wenn sie für die Argumentation von Frank ausschlaggebend ist, von Frank oder Dritten explizit herausgearbeitet werden. Dies betrifft insbesondere „charakteristische" Merkmale, die es unmittelbar gestatten, eine Theorie als entweder vergangenheits- und gegenwartsfokussiert (im Rahmen des konventionellen Theorieverständnisses) oder als zukunftsoffen (Theorie im Sinne des erweiterten Theorieverständnisses von Frank) zu klassifizieren. Ein solches Merkmal erwähnt Frank (2017b), S. 5728, explizit: Seiner Ansicht nach würden Theorien der Wirtschaftsinformatik („Information Systems [Research]" oder kurz „IS") im Gegensatz zu naturwissenschaftlichen Theorien es grundsätzlich nicht gestatten, Prognosen (für das zukünftige Verhalten von Informationssystemen oder ihrer Benutzer) aufzustellen. Die anschließende Argumentation von Frank zugunsten eines erweiterten Theorieverständnisses lässt sich zwar auf der Grundlage dieser zentralen Unterstellung „im Prinzip" nachvollziehen. Aber die Verfasser verstehen die These einer „grundsätzlichen" Prognoseuntauglichkeit von Theorien der Wirtschaftsinformatik nicht unmittelbar, zumal sich die von Frank angeführten Argumente (Kontingenz, freier Wille und Nichtübertragbarkeit von Erkenntnissen über vergangene Sachverhalte auf die Zukunft) in Zweifel ziehen lassen. Dies betrifft insbesondere die Relevanz von „stochastischen" gesetzesartigen Aussagen, die sich in den Wirtschaftswissenschaften auf Kollektive von Akteuren (anstelle einzelner Individuen) beziehen und in der weit verbreiteten Kritik an „Neo-Positivismus", „Behaviorismus" und „Szientismus" oftmals übersehen werden, sowie die gedankliche Notwendigkeit von „gesetzesartigen" Wirkungsprognosen (!) für Handlungen, die zur „rationalen" Gestaltung zukünftiger Informationssysteme und des Verhaltensspielraums ihrer Benutzer in irgendeiner Weise beitragen sollen. Vgl. zur (Gegen-)Kritik an der Kritik behavioristischer Erkenntnispositionen in der Wirtschaftsinformatik auch die kurzen, aber stichhaltigen Ausführungen in Heinrich (2011), S. 234. Vgl. ebenso die grundsätzliche Akzeptanz behavioristischer Erkenntnispositionen in der Wirtschaftsinformatik (nach ihrer vorherigen Kritisierung auf S. 664 f.) Österle u. a. (2010), S. 666 (dort als „verhaltensorientierte Forschung" angesprochen); Österle u. a. (2011), S. 8. Diese Argumente und Gegenargumente lassen sich in der hier gebotenen „Kürze" nicht detailliert entfalten. Daher wäre an anderer Stelle über die These von Frank hinsichtlich der – angeblichen oder tatsächlichen – Prognoseuntauglichkeit von Theorien der Wirtschaftsinformatik ausführlicher zu diskutieren. Allein der Anstoß zu einer solchen Diskussion verdeutlicht die große nicht nur wissenschaftstheoretische, sondern auch wissenschaftssoziologische (oder „pragmatische") Relevanz der grundlegenden Überlegungen von Frank zu einem erweiterten Theorieverständnis.

ten im Brennpunkt der Diskussion, an der sich – nach Erinnerung eines der Verfasser (Zelewski) – vor allem Vertreter der Wirtschaftsinformatik und der Betriebswirtschaftslehre „vehement“ beteiligten.[28] Vor allem hinsichtlich des „maßgeblichen“ wissenschaftlichen Theoriebegriffs ließ sich in der „lebhaft“ geführten Diskussion keine Einigung erzielen. Der hier vorgelegte Beitrag versteht sich als Fortsetzung der damals von Frank inspirierten Diskussion um die wissenschaftstheoretischen Fundamente von Wirtschaftsinformatik und Betriebswirtschaftslehre.[29]

Um die Diskussion im vorliegenden Beitrag inhaltlich zuzuspitzen und anschließende Diskussionen zu stimulieren, werden fünf – bewusst pointiert formulierte und somit „mutige“ – Thesen vorangestellt, auf die in den nachfolgenden Kapiteln zurückgekommen wird:[30]

- *These der Theorielosigkeit*:[31] Wirtschaftsinformatik und Betriebswirtschaftslehre verfügen in ihrem Basisbereich über keine „wissenschaftlich akzeptable“ theoretische Fundierung.[32]

28 Vgl. dazu das Abstrakt zu einem der Vorträge zum Forschungsseminar (Frank 2017a), S. 1.

29 Nicht nur in den nachfolgenden Thesen, sondern im gesamten vorliegenden Beitrag wird hinsichtlich der theoretischen Fundierung und in Bezug auf das wissenschaftliche Theorieverständnis zwischen Wirtschaftsinformatik und Betriebswirtschaftslehre nicht grundsätzlich unterschieden. Dies liegt an der subjektiven Einschätzung der Verfasser, dass sich die beiden Disziplinen hinsichtlich der beiden vorgenannten Aspekte des wissenschaftlichen Grundlagen- oder Basisbereichs nicht wesentlich voneinander unterscheiden. Über diese Einschätzung lässt sich natürlich streiten. Sollten entsprechende „Streitschriften“ durch diesen Beitrag angeregt worden sein, wird dies von den Verfassern ausdrücklich begrüßt.

30 Es ließen sich weitere Thesen hinzufügen, die in diesem Beitrag jedoch nicht weiter berücksichtigt werden, weil sie aus der Sicht der Verfasser keinen substanziellen Beitrag zum Theorieverständnis leisten. Dazu gehört z. B. die „radikale“ These der *Theorieüberflüssigkeit*, die prominent von Anderson (2008), o. S., vertreten wird. Anderson vertritt die These, dass in den „modernen“ Wissenschaften die Konstruktion und die Anwendung von „kausalen“ und „erklärenden“ Theorien (und Modellen, die Anderson weitgehend synonym zu Theorien verwendet) obsolet wäre, weil es angesichts von neuartigen Entwicklungen der Informatik – wie etwa „Big Data“, Supercomputern sowie Künstlichen Neuronalen Netzen (insbesondere Deep Learning Networks) – möglich wäre, gehaltvolle Wissenschaft ohne Rekurs auf Theorien zu betreiben. Stattdessen reichten mustererkennende Algorithmen, die Korrelationen (anstelle von Kausalitäten) in großen Datenmengen mittels maschinellen Lernens aufzudecken vermögen, vollkommen aus, um gehaltvolle wissenschaftliche Erkenntnisse zu gewinnen. Anderson führt zum Beleg seiner These auch mehrere durchaus prägnante Beispiele aus dem „modernen Wissenschaftsbetrieb“ an. Vgl. zur Rezeption dieser theoriebezogenen Obsoleszenzthese auch Frank (2021b), S. 129; Frank und Bock (2020), S. 20; Frank (2019), S. 48 f. (dort aber in anderem Kontext und ohne Bezug auf Anderson).

31 Vgl. die Hinweise in der o. a. Fußnote 8 mit explizitem Bezug auf die Wirtschaftsinformatik. Vgl. darüber hinaus Eichhorn (1979), S. 86, 97 u. 102 (indirekt, mit der damals [Publikationsdatum: 1979] aufgestellten Behauptung, weder mikro- noch makroökonomische Theorien im Besonderen noch wirtschaftswissenschaftliche Theorien im Allgemeinen besäßen, abgesehen von technisch-naturwissenschaftlichen Zusammenhängen, jeweils eine [nicht-triviale] nomische Hypothese). Vgl. auch die Argumentation in Albert (1957), S. 67 ff., zum Fehlen strenger oder „klassischer“ (S. 68) Theorien im Bereich der Sozialwissenschaften: „Wenn man die in den Sozialwissenschaften auftretenden Theorien [betrachtet; Ergänzung durch die Verfasser] … Sie machen gar nicht den Anspruch, generelle Hypothesen von raum-zeitlich unbeschränkter Gültigkeit zu enthalten … sind gewöhnlich Regelmässigkeiten, die sich bei näherer Überprüfung als kulturell und historisch relativ herausstellen.“ (S. 67). Daher schlägt Albert die Auseinandersetzung mit „abgeschwächten“ Quasi-Theorien vor; darauf wird noch zurückgekommen.

32 Von einer „wissenschaftlich akzeptablen“ theoretischen Fundierung ist hier genau dann die Rede, wenn die Fundierung einer Wissenschaftsdisziplin auf mindestens einer Theorie basiert, die den Anforderungen eines – noch näher zu explizierenden – wissenschaftlichen Theorieverständnisses entspricht. Die wissenschaftliche Akzeptabilität einer theoretischen Fundierung kann in Abhängigkeit vom jeweils explizit zugrunde gelegten oder auch nur implizit präsupponierten wissenschaftlichen Theorieverständnis variieren. Auf diese immanente Varianz der hier vorgetragenen Thesen relativ zum wissenschaftlichen Theorieverständnis wird im Folgenden nicht mehr explizit hingewiesen, auch wenn sie weiterhin implizit vorausgesetzt wird.

- *These der Theorieunmöglichkeit*:[33] Wirtschaftsinformatik und Betriebswirtschaftslehre können prinzipiell keine theoretische Fundierung aufweisen, weil Theorien – vor allem dann, wenn sie (nicht-triviale) nomische Hypothesen umfassen sollen – als Erkenntnisinstrumente für diese Wissenschaftsdisziplinen unangemessen sind.
- *These der schwachen Theoriefundierung*:[34] Wirtschaftsinformatik und Betriebswirtschaftslehre verfügen in ihrem Basisbereich über eine „wissenschaftlich akzeptable" theoretische Fundierung, aber der hierbei zugrunde gelegte wissenschaftliche Theoriebegriff wird nicht streng definiert, sondern oftmals ohne Bezug auf nomische Hypothesen und überwiegend natürlichsprachlich[35] umschrieben.
- *These der starken nomischen Theoriefundierung*:[36] Wirtschaftsinformatik und Betriebswirtschaftslehre verfügen in ihrem Basisbereich über eine „wissenschaftlich akzeptable" theoretische Fundierung und der hierbei zugrunde gelegte wissenschaftliche Theoriebegriff wird in dem Sinne streng definiert, dass eine wissenschaftliche Theorie mindestens eine nicht-triviale nomische Hypothese umfassen muss.

33 Vgl. dazu die nachfolgende Fußnote zur „schwachen" Theoriefundierung. Die dort angesprochene Ansicht, dass (gehaltvolle, nicht-triviale) nomische Hypothesen in den Bereichen der Wirtschaftswissenschaften im Allgemeinen sowie der Wirtschaftsinformatik und Betriebswirtschaftslehre im Besonderen unangemessen seien, lässt sich zur These der Theorieunmöglichkeit erweitern, wenn „mögliche" Theorien mit einem Theorieverständnis verknüpft werden, das mindestens eine (nicht-triviale) nomische Hypothese als essenziellen Theoriebestandteil postuliert. Dieses Postulat ist zwar nicht notwendig, wie aus den nachfolgenden Erläuterungen zum „schwachen" Theorieverständnis hervorgehen wird. Aber es wird von Autoren, die gegen die Relevanz von Theorien für die Wirtschaftswissenschaften opponieren, oftmals als implizite Präsupposition verwendet.

34 Die These der schwachen Theoriefundierung wird vor allem von Wissenschaftlern vertreten, die grundsätzlich bestreiten, dass für die Bereiche der Wirtschaftswissenschaften im Allgemeinen sowie der Wirtschaftsinformatik und Betriebswirtschaftslehre im Besonderen (gehaltvolle, nicht-triviale) nomische Hypothesen aufgestellt werden können. Diese Ansicht ist weit verbreitet, insbesondere als „Abgrenzungsreflex" gegenüber als untauglich angesehenen Theorieverständnissen der Naturwissenschaften, deren Übernahme in die Wirtschaftswissenschaften oftmals als „Szientismus", „Neo-Positivismus" oder „Behaviorismus" stigmatisiert wird. Übliche Argumente hierfür sind, dass sich erstens die (angeblich) „deterministischen" Naturgesetze der Naturwissenschaften grundsätzlich nicht auf die Kulturwissenschaften, wie z. B. die Wirtschaftswissenschaften, übertragen lassen. Die stochastische „Natur" zahlreicher nomischer Hypothesen z. B. der „modernen" Physik, wie etwa der Quantenphysik, und auch der Evolutionstheorie werden hierbei weitgehend ignoriert. Zweitens wird oftmals auf den „freien Willen" der Akteure („Subjekte") in den Kultur-, Sozial- oder Wirtschaftswissenschaften verwiesen, der es unangemessen erscheinen lasse, das Verhalten dieser Akteure mittels „deterministischer" nomischer Hypothesen antizipieren zu wollen. Vgl. Wolf (2020), S. 6 f. u. 20 f. (distanziert); Heinrich (2011), S. 235; Frank (2009), S. 163. Auch in dieser Hinsicht liegt nach Ansicht der Verfasser ein „Zerrbild" wirtschaftswissenschaftlicher Forschung vor, weil die hohe Relevanz „lediglich" stochastischer nomischer Hypothesen im Hinblick auf das kollektive Verhalten großer Anzahlen von Akteuren weitgehend ausgeblendet wird. Leider besteht in der hier gebotenen Kürze nicht der Raum, um auf die „üblichen" Argumente des o. a. „Abgrenzungsreflexes" näher einzugehen. Vielleicht bietet sich im Anschluss an die herausfordernden Beiträge von Frank zu wissenschaftstheoretischen Grundlagen der Wirtschaftsinformatik eine Gelegenheit („Agenda"), um die zuvor nur angedeuteten Kontroversen im wirtschaftswissenschaftlichen Kontext eingehender zu diskutieren.

35 Vgl. zu erheblichen Vorbehalten gegenüber formalsprachlichen Theorieformulierungen in den Sozial- und Wirtschaftswissenschaften Frank (1999), S. 138 ff. u. 153 f. (ohne die Vorteile formalsprachlicher Formulierungen grundsätzlich zu verneinen; vgl. S. 137 u. 145); Frank (1997), S. 28 f. Beispielsweise fasst Frank (1999), S. 140, sehr prägnant zusammen: „Will man den Schwierigkeiten einer formalsprachlichen Beschreibung der Realität entgehen, bleibt allein die Aufbereitung von Erkenntnissen mit Hilfe einer natürlichen Sprache." Analog hierzu vertritt Frank (1997), S. 29, die Ansicht, „daß es Phänomene gibt, die sich … einer formalisierten Beschreibung … entziehen, über die aber dennoch vernünftig geredet werden kann." Vgl. auch zu einer „abgewogenen Position" Moulines (2008), S. 44 f.

36 Diese These entspricht weitgehend dem „starken" Theorieverständnis, auf das später ausführlicher eingegangen wird. Es wird lediglich darauf verzichtet, eine „logische" und formalsprachlich präzisierte Ordnung des Aussagenzusammenhangs einer Theorie zu fordern.

- *These der starken nomisch-formalsprachlichen Theoriefundierung*:[37] Wirtschaftsinformatik und Betriebswirtschaftslehre verfügen in ihrem Basisbereich über eine „wissenschaftlich akzeptable" theoretische Fundierung und der hierbei zugrunde gelegte wissenschaftliche Theoriebegriff wird in dem Sinne streng definiert, dass eine wissenschaftliche Theorie mindestens eine nicht-triviale nomische Hypothese umfassen muss und entweder in formalsprachlicher Weise ausformuliert sein muss oder sich zumindest in formalsprachlicher Weise rekonstruieren lässt.

Die voranstehenden fünf Thesen zielen vor allem darauf ab, die Diskussion über die theoretische Fundierung von Wirtschaftsinformatik und Betriebswirtschaftslehre im Sinne eines „Thesenstreits" zu stimulieren. Daher wurden sie möglichst „einfach" und pointiert formuliert. Als „epistemischer Preis" wurde hierfür in Kauf genommen, dass diese Thesen weder den Bereich denkmöglicher Einstellungen zur theoretischen Fundierung einer Wissenschaftsdisziplin vollständig abdecken noch überschneidungsfrei formuliert sind. Beispielsweise kommt es zu Thesenüberschneidungen dadurch, dass die These der Theorieunmöglichkeit die These der Theorielosigkeit impliziert und dass die These der starken nomisch-formalsprachlichen Theoriefundierung eine (formalsprachenbezogene) Spezialisierung der These der starken nomischen Theoriefundierung darstellt. Um zu den o. a. Thesen Stellung zu beziehen, wird im Folgenden auf zweistufige Weise argumentiert.

Auf der ersten Stufe wird auf den wissenschaftlichen Theoriebegriff eingegangen. Ohne den Anspruch zu erheben, den facettenreichen und „abgründigen" Theoriebegriff aus wissenschaftlicher Perspektive auszuloten,[38] werden als Diskussionsgrundlage zwei Gruppen von Theorieverständnissen[39] angesprochen, die sich – so die Intention der Verfasser – als möglichst anschlussfähig an die „vorherrschende" Theoriediskussion in Wirtschaftsinformatik und Betriebswirtschaftslehre erweisen. Diese beiden Theorieverständnisse liegen vor allem der *dritten* These der schwachen Theoriefundierung einerseits sowie der *vierten* und *fünften* These der nomischen bzw. der nomisch-formalsprachlichen Theoriefundierung andererseits zugrunde.

Auf der zweiten Stufe wird das zweite, nomisch und formalsprachlich ausgerichtete Theorieverständnis vorausgesetzt, um anhand eines „schlichten", aber dennoch „abgründigen" Beispiels aufzuzeigen, das sich in der Tat eine theoretische Fundierung von Wirtschaftsinformatik und Betriebswirtschaftslehre nachweisen lässt. Die Argumentation beruht auf der exemplarischen Rekonstruktion eines betriebswirtschaftlichen[40] Modells als nomische

37 Diese These entspricht abermals weitgehend dem „starken" Theorieverständnis, das bereits in der voranstehenden Fußnote erwähnt wurde, betrachtet jedoch die „logische" und formalsprachlich präzisierte Ordnung des Aussagenzusammenhangs einer Theorie als essenziell.

38 Darauf wird in Kapitel 1.2 zurückgekommen.

39 Obwohl es sich strenggenommen jeweils um *Gruppen* von Theorieverständnissen handelt, deren Mitglieder gruppenintern zahlreiche Nuancen aufweisen (auf die hier nicht ausführlich eingegangen wird), wird im Folgenden der Einfachheit halber nur kurz von *Theorieverständnissen* gesprochen. Darüber hinaus wird zwischen *Theorieverständnissen* einerseits und *Theorienkonzepten* andererseits differenziert. Der Begriff „Theorienkonzepte" ist auf einer übergeordneten, abstrakten Ebene angesiedelt, in der es um *grundsätzliche* Konzipierungen des Theoriebegriffs geht. In dieser Hinsicht wird zwischen den Theorienkonzepten des „statement view" einerseits und des „non statement view" andererseits unterschieden. Innerhalb eines Theorienkonzepts (im hier vorgelegten Beitrag innerhalb des „statement view") lassen sich unterschiedliche Theorieverständnisse identifizieren, die zwar hinsichtlich des zugrunde liegenden Theorienkonzepts übereinstimmen, es aber durch zusätzliche Anforderungen an den Theoriebegriff inhaltlich ausdifferenzieren.

40 Über die Fokussierung auf ein *betriebswirtschaftliches* Modell lässt sich angesichts eines Beitrags, der explizit Betriebswirtschaftslehre und Wirtschaftsinformatik adressiert, trefflich streiten. Zur Rechtfertigung des betriebswirtschaftlichen Modellbezugs lassen sich zumindest drei Argumente anführen. Erstens hat einer der

und formalwissenschaftlich ausformulierte „Miniaturtheorie“[41] im Sinne der o. a. *fünften* These. Mittels dieser Rekonstruktion eines exemplarischen theoretischen Fundaments werden die *erste* und die *zweite* These der Theorielosigkeit bzw. der Theorieunmöglichkeit für die Betriebswirtschaftslehre direkt und für die hinsichtlich des Aspekts der theoretischen Fundierung eng verwandte[42] Wirtschaftsinformatik zumindest indirekt widerlegt.

1.2 Skizze zweier Mainstream-Theorieverständnisse

In der hier gebotenen Kürze ist es nicht möglich, den wissenschaftlichen Theoriebegriff in einem „angemessenen“ Überblick zu würdigen.[43] Dies ist kein Anliegen dieses Beitrags. Stattdessen werden nur zwei wissenschaftliche Theorieverständnisse[44] skizziert, die aufgrund ihrer deutlichen „semantischen Differenz“ aus der Perspektive der Verfasser hoffentlich ein möglichst großes Spektrum unterschiedlicher Verständnisvarianten im Sinne von definitorischen Antipoden zumindest ansatzweise überdecken. Außerdem wird darauf hingewiesen, dass die beiden Theorieverständnisse nur aus der Perspektive des etablierten Theorienkonzepts des „statement view“ – oftmals auch als „received view“ bezeichnet – vorgetragen werden. Der alternative „non statement view“ der analytischen Wissenschaftstheorie bleibt zunächst unberücksichtigt,[45] weil er sich weit außerhalb des „Mainstreams“ der ökomischen Forschung, insbesondere auch der Wirtschaftsinformatik[46] und Betriebswirtschaftslehre, befindet.

Verfasser (Zelewski) an anderer Stelle und „vor langer Zeit“ eine analoge Argumentation zur Rekonstruktion einer theoretischen Fundierung der *Wirtschaftsinformatik* vorgelegt. Es handelte sich damals um die Rekonstruktion eines theoretischen Fundaments für das „Produktivitätsparadoxon der Informationstechnik“ im Anschluss an diesbezügliche Überlegungen von Stickel; vgl. Zelewski (1999), S. 32 ff. Diese theoretische Rekonstruktionsarbeit wird im hier vorgelegten Beitrag nicht schlicht wiederholt, sondern durch ein neuartiges Beispiel ergänzt. Zweitens empfiehlt sich das hier diskutierte betriebswirtschaftliche Beispiel durch seine „Übersichtlichkeit“ und „intuitive Verständlichkeit“, weil es sich im Rahmen des weithin vertrauten Theorienkonzepts des „statement view“ (darauf wird zurückgekommen) behandeln lässt. Die zuvor erwähnte theoretische Rekonstruktion des „Produktivitätsparadoxons der Informationstechnik“ erfolgte hingegen auf der Basis des formalsprachlich anspruchsvolleren, aber auch inhaltlich schwerer verständlichen (und daher weniger „anschlussfähigen“) Theorienkonzepts des „non statement view“ des strukturalistischen Theorienkonzepts. Drittens wurde bereits auf die Einschätzung der Verfasser dieses Beitrags hingewiesen, dass sich Wirtschaftsinformatik und Betriebswirtschaftslehre hinsichtlich der Diskussion ihrer theoretischen Fundierung nicht wesentlich unterscheiden, sodass ein Beispiel aus dem Bereich der Betriebswirtschaftslehre als „pars pro toto“ für den Gesamtbereich von Wirtschaftsinformatik und Betriebswirtschaftslehre dienen kann.

41 Auf den Begriff der „Miniaturtheorie“ wird später zurückgekommen.

42 Vgl. dazu die Fußnoten 29 und 40 (drittes Argument).

43 Vgl. zu allgemeinen Diskussionen des wissenschaftlichen, insbesondere auch wirtschaftswissenschaftlichen, Theoriebegriffs beispielsweise Brühl (2021), S. 191 ff.; Wolf (2020), S. 2 ff. u. 13 ff.; Wiltsche (2013), S. 101 ff.; Balzer (2009), S. 46 ff., 56 ff. u. 142 ff.; Popper (2005), S. 36 ff. u. 47 ff.; Eichhorn (1979), S. 80 u. 84 ff.

44 Auf das Attribut „wissenschaftlich“ wird, wie bereits an früherer Stelle angekündigt, des Öfteren der Einfachheit halber verzichtet, weil in diesem Beitrag ausschließlich auf wissenschaftliche Theorieverständnisse und hiermit korrespondierende „Definitionen“ (Anführungszeichen, weil es sich um keine ausformulierten Begriffsdefinitionen, sondern lediglich um Definitionsskizzen handelt) des „wissenschaftlichen“ Theoriebegriffs eingegangen wird.

45 Auf den „non statement view“ der analytischen Wissenschaftstheorie wird später im Sinne eines Ausblicks kurz zurückgekommen.

46 Frank gehört zu den wenigen Wissenschaftlern der Wirtschaftsinformatik, die auf den „non statement view“ als alternatives Theorienkonzept explizit eingehen; vgl. Frank (2006c), S. 57; Frank (1999), S. 150 f. (allerdings mit erheblichen Vorbehalten gegenüber dem „non statement view“ auf S. 151, wie etwa der Kritik „seiner rigiden formalen Anforderungen“ sowie der zusammenfassenden Würdigung: „Darüber hinaus ist m. E. der Versuch, eine formale Rekonstruktion wissenschaftlicher Erkenntnis ... im Falle der Wirtschaftswissenschaften allzu

Auf der einen Seite steht ein *„schwaches“*[47] Theorieverständnis.[48] Sowohl das Theorieverständnis als auch die zugehörigen Theorien werden überwiegend „nur“ *natürlichsprachlich* spezifiziert.[49] Auf die Anforderung (nicht-trivialer) *nomischer Hypothesen* wird in der Regel *verzichtet.*[50]

Aus der Perspektive dieses Theorieverständnisses wird eine wissenschaftliche Theorie im Wesentlichen[51] durch vier essenzielle (oder konstituierende)[52] Anforderungen geprägt: Allgemeinheit[53], Abstrahierung (oder Abstraktion)[54], Widerspruchsfreiheit (oder Konsis-

ambitioniert.“ [Satzbau aus dem Original unverändert übernommen]). Vgl. auch den „Multi-Autoren-Beitrag“ (Bichler u. a. 2016), S. 292.

47 Das Attributepaar „schwaches“ versus „strenges“ Theorieverständnis darf nicht als ein implizites Werturteil missverstanden werden. Vielmehr geht es nur darum, wie „stark“ die Kriterien oder Anforderungen formuliert sind, die von einer Theorie erfüllt werden müssen, um dem jeweils betrachteten Theorieverständnis zu genügen. Die Verfasser dieses Beitrags gehen davon aus, dass die Anforderungen an das „strenge“ Theorieverständnis wesentlich stärker sind als die Anforderungen an das „schwache“ Theorieverständnis, wie z. B. hinsichtlich der Existenz mindestens einer nicht-trivialen nomischen Hypothese und der systematischen Ordnung des Aussagenzusammenhangs mittels (formalsprachlicher) Junktoren und Quantoren. Diese Anforderungen werden nur an das „strenge“, aber nicht an das „schwache“ Theorieverständnis gestellt. Über diese Einschätzung der Anforderungsstärke lässt sich natürlich diskutieren. Am Rande sei angemerkt, dass Frank (2006c), S. 41, von „abstractions of the factual (e. g., … ‚weak‘ theories)“ spricht. Zwar lassen sich seine Ausführungen zu Theorien weitgehend dem „schwachen“ Theorieverständnis zuordnen, aber es bleibt in diesem Kontext offen, ob er mit „weak“ tatsächlich die Attribuierung des hier vorgestellten Theorieverständnisses meint.

48 Vgl. zu Beispielen für dieses „schwache“ Theorieverständnis, in denen die nachfolgend angeführten vier Charakteristika „mehr oder minder“ deutlich angesprochen werden, beispielsweise Heinrich (2011), S. 99; Frank (2006c), S. 38 f.

49 Vgl. Balzer (2009), S. 58 f. Frank (2006c), S. 38, fordert immerhin die Formulierung einer Theorie mittels einer *präzisen* Sprache. Dies lässt zwar zunächst noch offen, ob es sich um eine formal- oder natürlichsprachliche Theorieformulierung handeln soll. Aber an späterer Stelle stellt er klar: „It does not imply the use of a formal language.“ Vgl. ebenso Frank (1999), S. 138 ff. u. 153 f., zu der Einschätzung, dass für gehaltvolle Theorien der Wirtschaftsinformatik eine formalsprachliche Theorieformulierung im Allgemeinen nicht ausreiche.

50 Dies entspricht der o. a. These der schwachen Theoriefundierung. Vgl. daher die Ausführungen in Fußnote 34. Vgl. des Weiteren hinsichtlich der im „schwachen“ Theorieverständnis üblichen Ablehnung der „grundsätzlichen Möglichkeit“ oder zumindest der „Angemessenheit“ nomischer Hypothesen, vor allem in den Bereichen von Kulturwissenschaften, Wirtschaftswissenschaften und Wirtschaftsinformatik, beispielsweise Frank (2017b), S. 5728 f. (hinsichtlich der dort behaupteten „Prognoseunmöglichkeit“ für Theorien der Wirtschaftsinformatik); Frank (2009), S. 163; Frank (2007a), S. 173. Die späteren Ausführungen zu substanziellen Einschränkungen hinsichtlich des Anspruchs auf (nicht-triviale) nomische Hypothesen, die z. B. im Zusammenhang mit „Quasi-Theorien“ und „Quasi-Gesetzen“ erfolgen, werden nicht zum „schwachen“ Theorieverständnis gezählt, weil jenen Ausführungen weiterhin das „starke“ Theorieverständnis zugrunde gelegt wird, dessen (nicht-trivialen) nomischen Hypothesen lediglich hinsichtlich ihres raum-zeitlich unbeschränkten („ubiquitären“) Geltungsanspruchs auf begrenzte Zeit- und Raumbereiche eingeschränkt („relativiert“) werden.

51 Über dieses Wesentlichkeitsurteil lässt sich mit guten Gründen streiten. Es stellt eine subjektive Gewichtung derjenigen Theorieanforderungen dar, die im State of the Art des „schwachen“ Theorieverständnisses nach Einschätzung der Verfasser überwiegend genannt werden.

52 Attribute oder Substantive, die in diesem Beitrag mit einem „oder“ verknüpft sind (in der Regel mit verdeutlichender Einklammerung), werden als synonyme Bezeichnungen für denselben gemeinten Sachverhalt behandelt, um die Anschlussfähigkeit an Bezeichnungsvarianzen in der einschlägigen Fachliteratur zumindest ansatzweise zu wahren.

53 Vgl. Wolf (2020), S. 2, 15 f. u. 19 f.; Heinrich (2011), S. 99; Frank (2006c), S. 38; Frank (2000), S. 44; Frank (1999), S. 135.

54 Vgl. Frank (2021a), S. 240 (dort allerdings nicht mit explizitem Theoriebezug, sondern im Hinblick auf Modelle [der Wirtschaftsinformatik], die mit zunehmender Abstraktion den Blick auf „mögliche Welten“ – vgl. S. 239 f. – eröffnen sollen, die mittels Gestaltungshandlungen erschlossen werden können); Frank (2017b), S. 5729; Frank (2015), S. 374; Frank (2010), S. 37; Frank (2009), S. 162; Frank (2008), S. 50 f. (ohne expliziten Theoriebezug); Frank (2007a), S. 173 u. 175 (ohne expliziten Theoriebezug); Frank (2007b), S. 405 (ohne expliziten Theoriebezug); Frank (2006c), S. 38 u. 49; Frank (2006a), S. 134; Frank (1999), S. 149 u. 155.

tenz)[55] sowie Falsifizierbarkeit[56] (oder empirische Überprüfbarkeit oder bislang ausstehende Falsifizierung oder Widerlegbarkeit). Mitunter kommen auch weitere Merkmale[57] hinzu, wie insbesondere die empirische Bewährung[58] sowie die Neuartigkeit[59] (oder Originalität[60]).

Erstens wird unter *Allgemeinheit* verstanden, dass sich eine Theorie nicht auf Einzelfälle beziehen soll, sondern auf eine möglichst große Klasse von empirischen Phänomenen („Einzelfällen"), die sich unter die Reichweite oder (synonym verstanden) den Anwendungsbereich einer Theorie subsumieren lassen. Je größer diese theoriespezifische Klasse empirischer Phänomene ausfällt (Anwendungsbreite der Theorie), desto höher wird die Theoriegüte eingeschätzt.[61]

Neben der Anforderung der Abstrahierung wird zuweilen auch die Anforderung der Idealisierung genannt. Zwischen Abstrahierung und Idealisierung kann durchaus begrifflich unterschieden werden. Beispielsweise wird die Idealisierung explizit erwähnt in Balzer und Brendel (2019), S. 20; Schurz (2011), S. 182; Albert (1976), Sp. 4689. Zur Verdeutlichung sei auf die häufige Verwendung von linearen Funktionen in Modellen (als „Miniaturtheorien") der Wirtschaftsinformatik und Betriebswirtschaftslehre verwiesen. Sie lassen sich als Idealisierungen gegenüber empirisch beobachtbaren nicht-linearen Phänomenzusammenhängen auffassen. Allerdings könnte auch argumentiert werden, dass mittels linearer Funktionen von „zufälligen, nicht-systematischen" Nicht-Linearitäten abstrahiert wird, sodass sich solche Idealisierungen von Abstrahierungen nicht grundsätzlich unterscheiden. Aus diesem Grund wird in diesem Beitrag der Einfachheit halber zwischen Idealisierungen und Abstrahierungen im Kontext des „schwachen" Theorieverständnisses nicht unterschieden, ohne die möglichen semantischen Differenzen anlässlich einer vertieften Inhaltsanalyse zu verkennen.

55 Vgl. Wolf (2020), S. 2, 15, 46 u. 48; Popper (2005), S. 48 (jedoch nicht im Kontext des „schwachen" Theorieverständnisses); Frank (1999), S. 136; Eichhorn (1979), S. 83 f., 84 f. u. 98; Albert (1976), Sp. 4680 (jedoch nicht im Kontext des „schwachen" Theorieverständnisses).

56 Vgl. Brühl (2021), S. 191; Wolf (2020), S. 17 f.; Frank (2017b), S. 5727 (allerdings mit Vorbehalten); Balzer (2009), S. 57 f., Frank (2007a), S. 174 f. (ohne expliziten Theoriebezug); Frank (2006c), S. 38; Popper (2005), S. 17 ff., 52, 54 ff., 84, 90 ff., 267 u. 506 ff. (jedoch nicht im Kontext des „schwachen" Theorieverständnisses); Frank (2000), S. 45; Frank (1999), S. 135 u. 149; Albert (1991), S. 57 ff. u. 139 (jedoch nicht im Kontext des „schwachen" Theorieverständnisses); Eichhorn (1979), S. 86; Albert (1976), Sp. 4677, 4680 f. u. 4684 (jedoch nicht im Kontext des „schwachen" Theorieverständnisses).

57 Die Liste weiterer Merkmale, die von einer Theorie im „schwachen" Theorieverständnis gefordert werden, lässt sich nahezu beliebig verlängern. Vgl. beispielsweise Brühl (2021), S. 191 (Kohärenz); Wolf (2020), S. 2 ff. u. 13 ff.; Frank (2017b), S. 5727 (Kausalität, Präzision und Objektivität, jedoch jeweils in distanzierter Weise vorgetragen) sowie S. 5729 f. u. 5733 f. (rationale Rechtfertigung: laut S. 5733 in Bezug auf „possible future worlds" vor allem mittels des rationalen Nachweises ihrer Realisierbarkeit oder „Machbarkeit" [„feasibility"] und ihrer Wünschenswertheit [„attractiveness"]); Frank (2015), S. 374 (Begründung); Frank (2010), S. 37 (Begründung); Frank (2009), S. 162 (Begründung); Frank (2007a), S. 173 f. u. 175 (Begründung); Frank (2006c), S. 38 (Rechtfertigung der theoriezugehörigen Aussagen); Frank (2006a), S. 134 (Begründung im Sinne der Rechtfertigung der theoriezugehörigen Aussagen); Popper (2005), S. 48 u. 115 (z. B. Freiheit von überflüssigen Bestandteilen und Einfachheit).

58 Vgl. Frank (2006c), S. 38; Frank (2000), S. 45; Frank (1999), S. 136; Albert (1991), S. 139 (jedoch nicht im Kontext des „schwachen" Theorieverständnisses); Eisenhardt (1989), S. 548.

59 Vgl. Frank (2000), S. 45; Frank (1999), S. 135; Eisenhardt (1989), S. 548 ("the goal is a new theory … presents new, perhaps framebreaking, insights").

60 Vgl. Frank (2021a), S. 240 (dort allerdings nicht mit explizitem Theoriebezug); Frank (2017b), S. 5729 f.; Frank (2015), S. 374; Frank (2010), S. 37; Frank (2009), S. 162; Frank (2008), S. 50 f. (ohne expliziten Theoriebezug); Frank (2007a), S. 173 u. 175 (ohne expliziten Theoriebezug); Frank (2007b), S. 405 (ohne expliziten Theoriebezug); Frank (2006c), S. 49 u. 51 ff.; Frank (2006a), S. 134.

61 Nur am Rande sei hier ein „definitorisches Problem" des hier skizzierten „schwachen" Theorieverständnisses erwähnt. Die Anforderung der Allgemeinheit besitzt sowohl essenziellen Charakter hinsichtlich des Theorieverständnisses (Grundsatzfrage: Liegt eine wissenschaftliche Theorie vor – „ja" oder „nein"?) als auch akzidenziellen Charakter (untergeordnete Frage: Falls eine Theorie vorliegt, wie hoch soll ihre Güte in Bezug auf ihre Anwendungsbreite beurteilt werden?).

Darüber hinaus wird auf eine zweite „Eigenart" des „schwachen" Theorieverständnisses hingewiesen: In wissenschaftstheoretischen Diskursen werden die Anwendungsbreite und die Präzision einer Theorie (neben weiteren Kriterien) in der Regel als gleichberechtigte Kriterien zur Beurteilung der Güte einer Theorie

Zweitens bezieht sich die *Abstrahierung* auf den Sachverhalt, dass jede Theorie von „idiosynkratischen" Besonderheiten von Einzelfällen absehen („abstrahieren") soll, um allgemeine „Muster", „Regularitäten" oder „Strukturen"[62] zu identifizieren, die allen Einzelfällen (aus dem Anwendungsbereich einer Theorie) gemeinsam zugrunde liegen.[63]

Drittens wird von einer „sinnvollen" Theorie nahezu immer *Widerspruchsfreiheit* gefordert. Diese Anforderung erscheint prima facie „unverzichtbar", weil aus einem in sich widersprüchlichen Aussagensystem jede beliebige Behauptung gefolgert werden kann („ex falso [sequitur] quodlibet" oder „ex contradictione sequitur quodlibet").[64] Dem Vorwurf der Beliebigkeit mag sich kaum ein Vertreter wissenschaftstheoretischer Reflexionen aussetzen. Diese Einschätzung gilt allerdings nicht streng, weil von einigen wenigen Vertretern – wie z. B. Rescher – auch „logische" Aussagenzusammenhänge für grundsätzlich verteidigbar gehalten werden, die selbstwidersprüchlich anmutende (inkonsistente) Aussagen umfassen dürfen.[65] Auf solche „Nischenpositionen" wird hier jedoch nicht weiter eingegangen, weil sie weder für das „schwache" noch für das „starke" Theorieverständnis eine Rolle spielen.

Viertens wird *Falsifizierbarkeit* oftmals mit dem „modernen" Theorieverständnis des Kritischen Rationalismus oder Realismus im Anschluss an z. B. Popper bzw. Albert als essenzielles Theoriemerkmal in Verbindung gebracht. Es dient vor allem dazu, um theoretisches („wissenschaftliches") Wissen gegenüber anderem Wissen, wie z. B. „Metaphysik" (einschließlich religiös geprägtem Offenbarungswissen), Spekulationen und – „modern gewendet" – „Verschwörungstheorien", abzugrenzen.

herangezogen. Im hier betrachteten „schwachen" Theorieverständnis wird aber nur das Gütekriterium der Anwendungsbreite mittels des Definitionsmerkmals der Allgemeinheit in die Theoriedefinition einbezogen. Das Gütekriterium der Präzision findet hingegen keinen Eingang in die Theoriedefinition. Diese Asymmetrie hinsichtlich der definitorischen Berücksichtigung von Anwendungsbreite und Präzision einer Theorie vermögen die Verfasser nicht unmittelbar nachzuvollziehen.

62 Der Bezug auf Muster, Regularitäten oder Strukturen ist an dieser Stelle für das „schwache" Theorieverständnis sehr wichtig, weil sich seine Anhänger oftmals gegen die Bezugnahme auf „nomische Hypothesen" (wie im „starken" Theorieverständnis üblich) vehement wehren, weil ihrer Ansicht nach „nomische Hypothesen" in wirtschaftswissenschaftlichen Theorien wegen des angeblichen Determinismus und Szientismus des nomischen Theorieverständnisses grundsätzlich verfehlt sind. Die Verfasser möchten an dieser Stelle lediglich anregen zu bedenken, ob die Redeweisen von „Mustern", „Regularitäten" oder „Strukturen" tatsächlich eine grundsätzliche semantische Differenz gegenüber „nomischen Hypothesen" zu begründen vermag. Zumindest bleiben die Anhänger der vorgenannten Redeweisen eine präzise Antwort darauf schuldig, inwiefern sich die von ihnen reklamierten „Muster", „Regularitäten" oder „Strukturen" (oder was auch immer) von „nomischen Hypothesen" substanziell unterscheiden.

63 Aus dieser Festlegung wird deutlich, dass die Anforderungen der Allgemeinheit und der Abstrahierung nicht unabhängig voneinander sind, sondern positiv miteinander korreliert sind: Je abstrakter eine Theorie formuliert ist, je mehr sie also von den Besonderheiten von Einzelfällen absieht, desto größer ist – zumindest tendenziell – ihre Allgemeinheit. Über diese „Non-Orthogonalität" der begrifflichen Anforderungen der Allgemeinheit und der Abstrahierung mag aus „definitionstheoretischer" Hinsicht gestritten werden; sie wird aber in diesem Beitrag nicht weiter thematisiert.

64 Vgl. Hoffmann (2013), S. 344; Popper (2005), S. 48 u. 67 f.; Rescher und Brandom (1980), S. 21 (distanziert); Rescher (1977), S. 64 (distanziert).

65 Vgl. Rescher und Brandom (1980) hinsichtlich „The Logic of Inconsistency" (Titel des Werks), insbesondere S. 3 f., 6 ff., 21 ff., 24 ff., 43 ff., 56 ff. u. 136 ff. Vgl. darüber hinaus die – auch formalsprachlich ausformulierte – „dialektische Logik" von Rescher (1977), S. 61 ff. Dort spricht Rescher explizit von „tolerance of selfcontradiction" und „abandoning the law of contradiction" (beides auf S. 62) sowie von „powerfully convincing arguments for mutually inconsistent conclusions" (S. 63) – bis hin zur Aufkündigung des kurz zuvor angesprochenen Prinzips der Vermeidung von „ex falso quodlibet" (S. 64). Vgl. auch Rescher (1977), S. 69 ff. zu „inconsistency-tolerant logics".

Über eine solche Demarkation genuin wissenschaftlichen Wissens in Theoriegestalt lässt sich streiten.[66] Vor allem lässt sich einwenden, dass es sich um das „spezielle“ Theorieverständnis einer „Wissenschaftsschule“ (Paradigma) handelt, die zwar in den Wirtschaftswissenschaften relativ (in Vergleich zu anderen wissenschaftlichen Disziplinen) weit verbreitet sein mag, sich aber auf keinen Fall einer breiten oder gar „dominanten“ Akzeptanz erfreuen kann. Hinzu kommt, dass von Vertretern falsifikationistischer Erkenntnispositionen oftmals übersehen wird, dass die Falsifikation von Hypothesen ihrerseits der Präsupposition als „wahr“ unterstellter Hilfstheorien, wie z. B. Messtheorien, bedarf, also eine inhärent-verifikationistische Komponente umfasst. Daher lassen sich gute Argumente zugunsten einer „holistischen“ Erkenntnisposition[67] anführen, aus deren Sicht nur Theorienkomplexe als „Ganzes“ hinsichtlich ihrer empirischen Geltungsansprüche untersucht und dann allenfalls in Bezug auf inhärente Widersprüche (u. a. im Hinblick auf theoriekomplexinterne „Protokollaussagen“ über „die“ theoriekomplexintern „interpretierte“ Realität) widerlegt werden können. Im Hinblick auf solche gravierenden erkenntnistheoretischen Vorbehalte aus theorieholistischer Perspektive scheuen die Verfasser davor zurück, die Anforderung der Falsifizierbarkeit als eine essenzielle Anforderung des „schwachen“ Theorieverständnisses vorauszusetzen. Darüber hinaus wird des Öfteren die *empirische Bewährung* einer Theorie gefordert. So sehr diese Anforderung prima facie einleuchtend erscheinen mag, so erscheint sie den Verfassern dennoch als essenzielle Anforderung eines Theorieverständnisses verfehlt. Denn eine Theorie „an sich“ sollte hinsichtlich der Beurteilung, ob sie die Anforderungen an eine Theorie im wissenschaftlichen Begriffsverständnis erfüllt, unabhängig von ihrer empirischen Bewährung sein. Daher stellt der Aspekt der empirischen Bewährung lediglich einen akzidenziellen Theorieaspekt dar, der nicht darüber entscheidet, ob eine Theorie „an sich“ vorliegt oder nicht (essenzielle Anforderung), sondern „lediglich“ dazu dient, die Güte[68] – eines bereits als Theorie akzeptierten Konstrukts – zu beurteilen.[69]

Aus ähnlichen Erwägungen rechnen die Verfasser auch die *Neuartigkeit* oder *Originalität* einer Theorie – präziser gesprochen: die Neuartigkeit der von einer Theorie vermittelten Erkenntnisse – nicht zu den essenziellen Anforderungen eines Theorieverständnisses. Denn eine Theorie „an sich“ liegt aus der Perspektive der Verfasser auch dann vor, wenn sie zwar keine neuartigen oder originellen Erkenntnisse vermittelt, aber Bekanntes in allgemeiner und abstrakter Weise zu erklären vermag. Daher sollte das Merkmal der Neuartigkeit oder Originalität für den Theoriebegriff abermals „nur“ akzidenziellen Charakter besitzen, sich jedoch anbieten, um die Güte einer Theorie zu beurteilen.[70]

66 Frank würde in dieser Hinsicht eventuell abermals von einem „neo-positivistischen“ Theorieverständnis sprechen, das bereits an früherer Stelle erwähnt wurde. Vgl. zu erheblichen Zweifeln von Frank am falsifikationistischen Forschungsprogramm im Sinne von Popper beispielsweise Frank (2017b), S. 5728.

67 Vgl. zu „holistischen“ Erkenntnispositionen, die z. B. als Duhem-[Neurath-]Quine-These bekannt geworden sind, Brühl (2021), S. 201 f.; Römpp (2018), S. 148 ff. u. 162 ff.; Wiltsche (2013), S. 41 ff., 47 f., 105 f., 109, 119 u. 172; Schurz (2011), S. 189 ff.; Duhem (1998), S. 243 ff.; Stegmüller (1987), S. 266 ff. Vgl. auch am Rande Zelewski (1993), S. 151 ff. u. 402 ff., zu den dort aus der Perspektive des strukturalistischen Theorienkonzepts („non statement view“) diskutierten „Theoriennetzen“ bzw. „Theorie-Holonen“. Vgl. ebenso Balzer (2009), S. 146 ff. (Theoriennetze) u. 149 ff. (Theorie-Holone).

68 Vgl. Eisenhardt (1989), S. 548.

69 Beispielsweise sei auf die „Superstring-Theorie“ aus den Bereichen von Physik und Kosmologie verwiesen, die sich aufgrund ihrer „inhärenten Besonderheiten“, wie z. B. eines weitaus höherdimensionalen Erfahrungsraums als der „klassischen“ vierdimensionalen Raumzeit, derzeit noch weitgehend einer empirischen Überprüfung – a fortiori auch einer empirischen Bewährung – entzieht. Vgl. Römpp (2018), S. 282. Dennoch besteht in der relevanten Wissenschaftler-Community kein Zweifel daran, der „Superstring-Theorie“ die Qualität einer wissenschaftlichen (sogar „starken“) Theorie zuzuerkennen.

70 In diesem Sinne lassen sich auch die Ausführungen von Eisenhardt (1989), S. 548, interpretieren.

Ein Beispiel für das „schwache" Theorieverständnis stellt die „Prospect Theory"[71] dar. Sie wird des Öfteren vor allem in betriebswirtschaftlichen Argumentationszusammenhängen als „Vorbild" für ökonomisch relevante Theorien genannt,[72] die somit auch für Wirtschaftsinformatik und Betriebswirtschaftslehre von Interesse sind. Solche Theorien[73] lassen sich u. a. dadurch charakterisieren, dass sie einerseits „allgemeine" Aussagen (Hypothesen) über das Verhalten von Akteuren in ökonomisch relevanten Handlungszusammenhängen treffen, andererseits sich aber auf weitgehend natürlichsprachliche Argumentationen beschränken, die allenfalls durch „grafische Illustrationen" bereichert werden, in denen „tendenzielle Zusammenhänge" (wie z. B. „Verhaltensmuster") visualisiert werden, allerdings (überwiegend) ohne numerisch konkret zu werden und ohne den expliziten Anspruch auf das Vorliegen einer nomischen Hypothese zu erheben.

Auf der anderen Seite konkurriert ein *„starkes"*[74] Theorieverständnis.[75] Es erhebt deutlich präzisere („stärkere") Anforderungen an eine wissenschaftliche Theorie. In einer – von den Verfassern pointiert formulierten – Definition lässt sich eine Theorie aus der Perspektive dieses Theorieverständnisses auffassen:

- als ein *systematischer Aussagenzusammenhang*

- mit mindestens einer *nicht-trivialen nomischen Hypothese.*

71 Vgl. Kahneman und Tversky (1979), S. 274 ff. (mit einem expliziten Theorieanspruch z. B. auf S. 274, 277, 286 u. 288 sowie mit nicht-trivialen nomischen Hypothesen beispielsweise auf S. 276 [in Gleichungsform], 279 u. 283 [jeweils nur angedeutet in der Gestalt einer grafischen „Prinzipskizze"]); Tversky und Kahneman (1991), S. 1039 ff., insbesondere S. 1046 ff.; Tversky und Kahneman (1992), S. 299 ff. u. 309 ff. Vgl. auch Wolf (2020), S. 156; Sander (2018), S. 61 ff. Zu einer aktuellen Anwendung der „Prospekt Theorie" in einem betriebswirtschaftlichen Kontext vgl. Khan u. a. (2021), S. 3385 f.

72 Vgl. beispielsweise Schmiel und Weitz (2019), S. 9 u. 14 (zwar nicht als „Vorbild" angesprochen, aber der Argumentation – neben einer weiteren Theorie – als Basis zugrunde gelegt, was eine deutliche Präferenz – u. a. – zugunsten der Prospect Theory belegt); Sander (2018), S. 64, 144 u. 231 (jeweils nur ansatzweise, vor allem im Hinblick auf die bevorzugenswerte empirische Bewährung im Vergleich zu anderen Varianten der Entscheidungstheorie); vgl. auch ansatzweise das Vorwort von Schmiel in Sander (2018), S. VI („Vorschlag für ein *theoretisches* Fundament von Steuerwirkungen, das an sozialwissenschaftliche Handlungshypothesen *anschlussfähig* ist." [kursive Hervorhebungen durch die Verfasser]); Schmiel (2016), S. 382.

73 Breite Überblicke über ökonomische, vor allem betriebswirtschaftliche Theorien der hier angesprochenen Art –also im Rahmen des „schwachen" Theorieverständnisses – finden sich beispielsweise (ohne dass dort das „schwache" Theorieverständnis explizit angesprochen wird) in Wolf (2020), S. 59 ff. u. 301 ff.; Macharzina und Wolf (2018), S. 44 ff. u. 70 ff. (dort als „theoretische Ansätze" thematisiert, die gemäß S. 45 zwar Theorien darstellen, aber deren „Vorläufigkeit" betonen).

74 Das Attributepaar „schwaches" versus „strenges" Theorieverständnis darf nicht als ein implizites Werturteil missverstanden werden. Vielmehr geht es nur darum, wie „stark" die Kriterien oder Anforderungen formuliert sind, die von einer Theorie erfüllt werden müssen, um dem jeweils betrachteten Theorieverständnis zu genügen. Die Verfasser dieses Beitrags gehen davon aus, dass die Anforderungen an das „strenge" Theorieverständnis wesentlich stärker sind als die Anforderungen an das „schwache" Theorieverständnis, wie z. B. hinsichtlich der Existenz mindestens einer nicht-trivialen nomischen Hypothese und der systematischen Ordnung des Aussagenzusammenhangs mittels (formalsprachlicher) Junktoren und Quantoren. Diese Anforderungen werden nur an das „strenge", aber nicht an das „schwache" Theorieverständnis gestellt. Über diese Einschätzung der Anforderungsstärke lässt sich natürlich diskutieren.

75 Vgl. zu Beispielen für dieses „starke" Theorieverständnis, das in den nachfolgend angeführten Quellen „mehr oder minder deutlich" angesprochen wird, Brühl (2021), S. 191 f. u. 310; Wiltsche (2013), S. 41; Popper (2005), S. 36 ff.; Albert (1976), Sp. 4679 f.; Albert (1957), S. 62 ff. Zum „starken" Theorieverständnis zählen insbesondere alle (weiteren) Beiträge, die es als eine essenzielle Anforderung an eine Theorie erachten, sie müsse mindestens eine nicht-triviale nomische Hypothese umfassen. Vgl. dazu die Quellen, die in Fußnote 85 (zusätzlich) angeführt werden.

Die Anforderung des *systematischen Aussagenzusammenhangs* referenziert zunächst explizit das zugrunde liegende „etablierte“ Theorienkonzept des „statement view“, weil „Aussagen“ mit „statements“ gleichzusetzen sind.[76] Darüber hinaus verbergen sich hinter der Formulierung „systematischer Aussagenzusammenhang“ mindestens zwei wichtige weitere Aspekte.

Erstens handelt es sich bei Theorien – aus dieser Perspektive – stets um *sprachliche Artefakte*[77], die zumindest zunächst nichts mit „der“ Realität zu tun haben. Dies eröffnet ein weites Erkenntnisfeld mit „abgründigen“ Problemen, weil sprachliche Artefakte ein überaus komplexes Erkenntnisobjekt darstellen. Sie unterliegen sprachphilosophischen Herausforderungen – wie z. B. im Anschluss an Wittgenstein mit seinem viel zitierten, aber nicht leicht verständlichen Tractatus Logico-Philosophicus[78] – bis hin zu Herausforderungen der Meta-Mathematik und der Informatik, wie etwa der Unentscheidbarkeit (oder Unvollständigkeit) von Arithmetik und Prädikatenlogik[79] bzw. der Unentscheidbarkeit des Halteproblems für „generische“ Turing-Automaten. Diese „abgründigen“ Probleme stellen Theorien im Sinne des „starken“ Theorieverständnisses vor erhebliche erkenntnistheoretische Herausforderungen, die im „schwachen“ Theorieverständnis noch nicht einmal ansatzweise „mitschwingen“, weil dort die genuin *sprachliche* Verfasstheit von Theorien überhaupt nicht zur Diskussion steht.

Zweitens wird auf den *systematischen* Zusammenhang der theoriekonstituierenden Aussagen großen Wert gelegt. Im Gegensatz zu zahlreichen, vor allem betriebswirtschaftlich geprägten Beiträgen, die sich „irgendwie“ mit Theorien (vor allem im vorgenannten „schwachen“ Theorieverständnis) befassen, spielt die Systematisierung des Aussagenzusammenhangs im „starken“ Theorieverständnis eine herausragende Rolle. Über den Systematisierungsbegriff lässt sich natürlich streiten. Er wird hier im Sinne einer „logischen“ und formalsprachlich präzisierten Ordnung des Aussagenzusammenhangs präzisiert.[80]

76 Vgl. zur Charakterisierung von Theorien als Aussagenzusammenhänge, Aussagensysteme, sprachliche Gebilde und Ähnliches, die auch vom „schwachen“ Theorieverständnis geteilt wird und daher keinen Gegensatz zwischen „schwachem“ und „starkem“ Theorieverständnis darstellt, z. B. Wolf (2020), S. 2 u. 13; Balzer (2009), S. 46; Frank (2006c), S. 38; Frank (1999), S. 134; Eichhorn (1979), S. 80; Albert (1957), S. 63.

77 Vgl. Frank (2017b), S. 5729; Frank (2006c), S. 41. Auch Frank spricht gern von Artefakten, allerdings überwiegend in Bezug auf andere Objekte, wie z. B. Informationssysteme und Software. Vgl. Frank (2021a), S. 231; Frank (2021b), S. 124 f.; Frank (2019), S. 43; Frank (2015), S. 374; Frank (2010), S. 37; Frank (2009), S. 162 u. 166 f.; Frank (2007a), S. 160; Frank (2006c), S. 1, 11 f., 30 f., 40 f., 43 f. u. 53; Frank und Schauer (2001), S. 10 f.; Frank (2000), S. 35 u. 38 ff.; Frank (1999), S. 154.

78 Vgl. die besonders bekannt gewordenen Einsichten in Wittgenstein (1996): „Die Grenzen meiner Sprache bedeuten die Grenzen meiner Welt.“ (S. 148, Abschnitt 5.6), „Meine Sätze erläutern dadurch, daß sie der, welcher mich versteht, am Ende als unsinnig erkennt, wenn er durch sie – auf ihnen – über sie hinausgestiegen ist. (Er muss sozusagen die Leiter wegwerfen, nachdem er auf ihr hinaufgestiegen ist.) Er muß diese Sätze überwinden, dann sieht er die Welt richtig.“ (S. 166, Abschnitt 6.54, bekannt geworden auch als „Leiter-Metapher“) sowie „Wovon man nicht sprechen kann, darüber muss man schweigen.“ (S. 166, Abschnitt 7). Siehe auch die entsprechenden Passagen in der Original-Publikation von 1921 auf S. 246 (zum Abschnitt 5.6) sowie S. 262 (zu den Abschnitten 6.54 und 7). Vgl. darüber hinaus einen Verweis auf den Abschnitt 5.6 z. B. in Frank (2009), S. 164.

79 Vgl. Hoffmann (2013), S. 119 ff. u. 289 ff. zum ersten Unvollständigkeitssatz sowie S. 340 ff. zum zweiten Unvollständigkeitssatz (insbesondere S. 131 ff. u. 143 ff., 292 ff., 319 f., 339 f. u. 340); Spies (2004), S. 217 ff., insbesondere S. 223 f.

80 Es wird eingeräumt, dass es für den oben geforderten „systematischen Aussagenzusammenhang“ nicht notwendig ist, den Aussagenzusammenhang einer Theorie in „logischer“, vor allem formalsprachlich präzisierter Weise zu ordnen. Stattdessen lässt sich die „logische“ Ordnung eines Aussagenzusammenhangs auch auf natürlichsprachliche Weise ausdrücken. Aber natürliche Sprachen weisen so mannigfaltige Möglichkeiten zu Vagheit, Mehrdeutigkeit und Intransparenz auf, dass hier im Hinblick auf ein „starkes“ Theorieverständnis

Dies bedeutet einerseits, dass es sich bei den Aussagen einer Theorie um Aussagen im *aussagenlogischen* Sinn handeln muss, also um „wahrheitsfähige“ Behauptungen oder Aussagen (statements), die sich prinzipiell als wahr oder falsch nachweisen lassen müssen (metaphysische Aussagen, die sich hinsichtlich ihres Wahrheitswerts aus empirischer Sicht als unüberprüfbar erweisen, werden also ausgeschlossen). Andererseits geht es insbesondere um die Verknüpfung der Aussagen, um einen „systematischen“ Aussagenzusammenhang zu konstituieren. An diese Aussagenverknüpfung werden zwei präzise Anforderungen gestellt.

Zunächst müssen die Aussagen mittels aussagenlogischer Junktoren verknüpft sein. Dazu gehören vor allem die Junktoren „und“ (Konjugat: $\wedge$), „oder“ (Adjugat: $\vee$), „nicht“ (Negat: $\neg$), „wenn, dann“ (Subjugat: $\rightarrow$) sowie „genau dann, wenn“ (Bijugat: $\leftrightarrow$).[81] Mittels dieser (aussagen-)logischen Anforderung erfolgt eine deutliche Abgrenzung gegenüber „Theorieumschreibungen“ im Rahmen des „schwachen“ Theorieverständnisses, in denen Theorien mittels natürlichsprachlicher Texte „irgendwie“ erläutert werden, ohne den präzisen (aussagen-)logischen Zusammenhang der einzelnen Textkomponenten (Aussagen) klar erkennen zu lassen. In dieser Hinsicht wird kurz auf die These verwiesen, bei (u. a.) betriebswirtschaftlichen Texten handele es sich um „narrative“ Veranstaltungen, bei denen es nicht um argumentative Stringenz und – im hier erörterten Kontext – um „logische Ordnung“ gehe, sondern lediglich um eine „spannende“ und Aufmerksamkeit erweckende „Story“.[82]

Andererseits reicht die „klassische“ Aussagenlogik nicht aus, um nicht-triviale nomische Hypothesen in logisch stringenter Weise auszudrücken. Dafür sind mindestens[83] die Ausdrucksmittel der *Prädikatenlogik* (erster Stufe)[84] erforderlich. Sie werden benötigt, um auszudrücken, dass ein Sachverhalt mit nomischem Anspruch räumlich und zeitlich „ubiquitär“, also „überall“ und „jederzeit“ gilt. Solche ubiquitären Geltungsansprüche entziehen

die formalsprachliche Strenge für die Formulierung des systematisch geordneten Aussagenzusammenhangs gefordert wird.

81 Von diesen Junktoren sind streng genommen nur zwei Junktoren notwendig; entweder „und“ sowie „nicht“ oder „oder“ sowie „nicht“. Alle übrigen Junktoren können aus diesen elementaren Junktoren zusammengesetzt werden. Vgl. Hermes (1991), S. 51 („zum Fall „und“ sowie „nicht“) u. S. 158 f. (zur Einführung weiterer Junktoren). Aber es vereinfacht die „logische“ Formulierung von Theorien (und von Modellen als „Miniaturtheorien“) erheblich, auf die o. a. zusätzlichen Junktoren als äquivalente „logische Abkürzungen“ zurückzugreifen.

82 Vgl. zu diesem narrativen oder rhetorischen Verständnis der Betriebswirtschaftslehre z. B. McCloskey (1995), S. 1320 ff.; McCloskey (1984), S. 98 ff. (z. B. S. 98: „Economics ...is merely a disciplined inquiry ... a collection of literary forms, not a science.“ sowie S. 99: „the view that economics is literary”, ähnlich auch auf S. 104 f.); McCloskey (1983), S. 482 ff. u. 493 ff., insbesondere S. 499 f. u. 512 ff. Aber auch andere wissenschaftstheoretisch renommierte Autoren sprechen sich zugunsten solcher narrativer Veranstaltungen anstelle „konventioneller Theorien“ aus. Vgl. Rorty (2021), S. 16 f. („großzügig“ interpretiert, vor allem S. 16: „Wendung gegen die Theorie und zur Erzählung“); Frank (2017b), S. 5731.

83 Später wird darauf eingegangen, dass auch „höhere“ Logiken als die Prädikatenlogik (erster Stufe) erforderlich sein können. Dies wird im Hinblick auf eine non-monotone epistemische Logik verdeutlicht werden. Aber auch andere logische Erweiterungen können diskutiert werden. Dies betrifft vor allem Prädikatenlogiken zweiter Stufe, in denen logische Aussagen nicht nur über „Individuen“ (Prädikatenlogik erster Stufe), sondern auch über Prädikate (Relationen, Funktionen usw.) zulässig sind. Vgl. Spies (2004), S. 226 f.; Hermes (1991), S. 140 f.

84 Die Prädikatenlogik erster Stufe lässt nur Quantifizierungen (mittels All- oder Existenzquantoren) über Individuenvariablen zu. Quantifizierungen von „übergeordneten“ logischen Ausdrücken, vor allem Prädikaten, Relationen und Funktionen, werden hingegen ausgeschlossen. Prädikatenlogiken höherer Stufen – vor allem zweiter Stufe – gestatten Quantifizierungen der vorgenannten „übergeordneten“ logischen Ausdrücke. Beispielsweise erfordert die prädikatenlogische Spezifizierung der Gleichheitsrelation („Identität“) eine Prädikatenlogik zweiter Stufe. Auch die „Frame Logic“, die im Rahmen der Erforschung Künstlicher Intelligenz (KI) zur prädikatenlogischen Spezifizierung von KI-Modellen verwendet wird, impliziert eine Prädikatenlogik zweiter Stufe.

sich den Ausdrucksmitteln der Aussagenlogik. Sie können erst mittels der Prädikatenlogik (erster Stufe) spezifiziert werden. Sie erfordern das prädikatenlogische Ausdrucksmittel des Allquantors (∀). Darüber hinaus ist auch das prädikatenlogische Ausdrucksmittel des Existenzquantors (∃) erforderlich, um Aussagen hinsichtlich des Vorliegens von „optimalen“ (minimalen oder maximalen) Problemlösungen „logisch korrekt“ ausdrücken zu können. Darauf wird zurückgekommen.

Die Anforderung der mindestens einen *nicht-trivialen nomischen Hypothese*[85] stellt den konzeptionellen Kern des „starken“ Theorieverständnisses dar. In dieser Hinsicht geht es

85 Vgl. Wolf (2020), S. 4 ff. u. 18 ff. (überwiegend nur als „Wenn-Dann-Aussagen“ thematisiert, aber z. B. auf S. 5 f., 18 u. 20 auch hinsichtlich ihrer Gesetzesartigkeit oder ihres nomischen Charakters explizit angesprochen); Bichler u. a. (2016), S. 299; Johansson (2016), S. 173 ff.; Schmiel (2016), S. 379 („Hypotheses on the effects of taxes have to meet critical rationalist requirements. In the case of the explanation. ... We need assumptions and *quasi-nomological hypotheses* from which we can deduce the result we want to explain. ... We use the term ‚*quasi-nomological* hypotheses‘ because the question *whether (economic) law* exists will not be considered further. ... it is reasonable to assume that there are *at least regularities* in economic facts and circumstances. ...a scientific explanation or prediction requires assumptions and *quasi-nomological hypotheses* to be empirically confirmed“ [kursive Hervorhebungen durch die Verfasser]), S. 380 f., 382 („In order to *explain* these deviations, we need hypotheses which substantiate the *quasi-nomological hypothesis* regarding tax effects.“ [kursive Hervorhebungen durch die Verfasser]), S. 383 u. 386; Balzer (2009), S. 46 f.; Moulines (2008), S. 94 ff.; Popper (2005), S. 36 ff., 84 u. 267; Frank (2000), S. 45 (nur ansatzweise als räumlich und zeitlich möglichst *invariante* Sachverhaltseigenschaften); Frank (1999), S. 135 (nur ansatzweise als räumlich und zeitlich möglichst invariante Sachverhaltseigenschaften und später auf S. 145 deutlich relativiert: „Aus erkenntnistheoretischer Sicht ... mit einer deutlichen *Verschiebung des Anspruchs* an die Forschung verbunden. Schließlich steht *weniger* das Streben nach ehernen *Gesetzen* des für Entscheidungen in Unternehmen relevanten Handelns im Vordergrund, sondern die (Re-) Konstruktion von Begriffen.“ [kursive Hervorhebungen durch die Verfasser]). Vgl. auch, jedoch mit substanziellen Einschränkungen, Eichhorn (1979), S. 86 (Einschränkung, dass es bislang [Publikationsdatum: 1979] weder mikro- noch makroökonomischen Theorien gelungen sei, abgesehen von technisch-naturwissenschaftlichen Zusammenhängen – jeweils eine [nicht-triviale] nomische Hypothese aufzustellen, sodass in den Wirtschaftswissenschaften „der Theoriebegriff sehr viel weiter gefaßt ... als in der Wissenschaftstheorie“ werden müsse); Albert (1976), Sp. 4678 f. u. 4681; Albert (1957), S. 63 u. 65 sowie vor allem S. 67 ff. u. 75 f. (mit der Einschränkung, dass in den Sozialwissenschaften in der Regel keine [nomischen] Hypothesen mit Anspruch auf raum-zeitlich unbeschränkte Geltung verfolgt würden, sondern lediglich Hypothesen oder „Regelmäßigkeiten“ [S. 67], die als „Quasi-Gesetze“ [S. 68 ff.] auf bestimmte Zeiträume [„Epochen“] und räumliche Regionen [„Kulturen“] eingeschränkt, d. h. relativiert sind, sodass sich auf der Grundlage solcher Hypothesen nur „Quasi-Theorien“ [S. 67 f.] aufstellen lassen und gegenüber den Naturwissenschaften eine „Erweiterung des Theoriebegriffs“ [S. 68] erforderlich wäre). Am Rande sei erwähnt, dass sich die o. a. Bezugnahmen von Schmiel (2016) auf quasi-nomische Hypothesen auf die zuvor skizzierten Einschränkungen des starken Theorieverständnisses, die beispielsweise von Eichhorn und Albert betont wurden, zurückführen lassen (ohne dass sich Schmiel explizit darauf bezieht).

Die Ansicht, Theorien sollten (nicht-triviale) nomische Hypothesen umfassen, findet sich sogar bei Habermas, der als einer der herausragenden Protagonisten der *Kritischen Theorie* sicherlich nicht im Verdacht steht, dem betriebswirtschaftlichen Mainstream-Verständnis von Theorien im Sinne des Kritischen Rationalismus oder Realismus zu folgen. Bemerkenswert ist beispielsweise folgendes Zitat aus Habermas (2020), S. 158: „Die systematischen Handlungswissenschaften, nämlich *Ökonomie*, Soziologie und Politik, haben, wie die empirisch-analytischen Naturwissenschaften, das Ziel, *nomisches Wissen* hervorzubringen. ... Eine kritische Sozialwissenschaft wird sich freilich dabei nicht bescheiden. Sie bemüht sich darüber hinaus, zu prüfen, wann die theoretischen Aussagen *invariante Gesetzmäßigkeiten* des sozialen Handelns überhaupt und wann sie ideologisch festgefrorene, im Prinzip aber veränderliche Abhängigkeitsverhältnisse erfassen.“ (kursive Hervorhebungen durch die Verfasser) sowie weiter auf S. 159: „Ein kritisch vermitteltes *Gesetzeswissen* kann auf diesem Wege das Gesetz selbst durch Reflexion zwar nicht außer Geltung, aber außer Anwendung setzen.“ (kursive Hervorhebungen abermals durch die Verfasser). Allerdings soll auch die kritische Distanz gegenüber „vorschnellen“ Verfestigungen angeblich „nomischer“ Sachverhalte, die im zweiten und dritten Satz des voranstehenden Zitats deutlich wird, keineswegs unterschlagen werden. Vgl. dazu auch die Anführung und Kommentierung des Habermas-Zitats durch Frank (1998c), S. 109 f., hinsichtlich seiner Relevanz für die Wirtschaftsinformatik. Vgl. ebenfalls Frank (2007a), S. 172, zur „Entwicklung von Aussagen mit generellem Geltungsanspruch“ seitens der Kritischen Theorie im Anschluss an Habermas.

zunächst nicht um „triviale“ nomische Hypothesen[86], wie sie aus definitorischen, logischen oder mathematischen Argumentationszusammenhängen vertraut sind. Vielmehr sind für eine realwissenschaftliche Theorie „nicht-triviale“ nomische Hypothesen von Interesse, die sich auf Sachverhalte jenseits rein definitorischer, logischer oder mathematischer Argumentationszusammenhänge erstrecken. Diese nicht-trivialen nomischen Hypothesen sind die „Essenz“ einer realwissenschaftlichen Theorie. Sie stellen (neben den vorgenannten Anforderungen) das „primäre“ Prüfkriterium dar, anhand dessen zu entscheiden ist, ob eine Theorie im wissenschaftlichen Theorieverständnis vorliegt – oder nicht. Um die Anforderung, dass mindestens eine nicht-triviale nomische Hypothese vorliegt, überprüfen zu können, bedarf es entsprechender (Überprüfungs-)Kriterien. Sie erstrecken sich zunächst in materieller Hinsicht darauf, dass es sich nicht um triviale Hypothesen definitorischer, logischer oder mathematischer Art (siehe oben) handelt. Diese nicht-trivialen nomischen Hypothesen behaupten etwas über den jeweils betrachteten Realitätsausschnitt, was sich hinsichtlich ihres Geltungsanspruchs empirisch überprüfen – also belegen oder widerlegen – lässt. Darüber hinaus bedarf es für eine nicht-triviale nomische Hypothese in (formal-)logischer Hinsicht der Anforderung, dass es sich um eine „ubiquitäre“ (prädikatenlogische) Aussage handelt, die einen Sachverhalt „immer“ (zeitlich) und „überall“ (räumlich oder branchenbezogen)[87] postuliert. Für derart nicht-triviale nomische Hypothesen sind prädikatenlogische Formeln erforderlich, die mit Allquantoren ($\forall$) über denkmögliche Zeit- und Raumpunkte (oder Branchen)[88] „quantifizieren“. Diese Anforderung einer „Allquantifizierung“ stellt eine prägnante Anforderung an eine prädikatenlogisch rekonstruierte Theorie dar. Es handelt sich um eine „conditio sine qua non“. Falls sie im Rahmen einer (prädikatenlogischen) Theorierekonstruktion[89] nicht erfüllt wird, liegt im Sinne des „starken“ Theorieverständnisses keine wissenschaftliche Theorie vor. Daher wird im Folgenden großer Wert darauf gelegt zu überprüfen, ob eine Theorie derart vorliegt, dass sie mindestens eine allquantifizierte, nicht-triviale nomische Hypothese umfasst.

86 Beispiele sind die „trivialen“ („a priori“ vor jeder empirischen Erfahrung gültigen), d. h. allgemeingültigen, logischen „Gesetze“ von De Morgan für die äquivalente Umformung der Negation von Kon- oder Adjugaten sowie der „Fundamentalsatz der Algebra“ hinsichtlich der Existenz von genau N komplexen Lösungen für jedes N-stellige Polynom.

87 In Theorien der Wirtschaftsinformatik und der Betriebswirtschaftslehre spielen „Raumpunkte“ in der Regel keine Rolle. Stattdessen sind raumbezogene Aussagen nach Einschätzung der Verfasser am besten in Bezug auf „Branchen“ umzuformulieren. Über diese Interpretation der Raumdimension lässt sich natürlich diskutieren.

88 Die Gleichsetzung von Raumpunkten mit Branchen lässt sich kritisch diskutieren. Sie stellt ein „Interpretationsangebot“ der Verfasser dar, weil sich Theorien (und Modelle) der Wirtschaftsinformatik und der Betriebswirtschaftslehre in der Regel nicht auf Raumbereiche, sondern – wenn überhaupt – auf Branchen („Industrien“) erstrecken.

89 Um Missverständnissen vorzubeugen, wird betont, dass es im hier vorgestellten Diskurskontext nicht um Theorien geht, wie sie in der einschlägigen Fachliteratur von Wirtschaftsinformatik und Betriebswirtschaftslehre in der Regel präsentiert werden (bis hin zu „narrativen Veranstaltungen“). Stattdessen beziehen sich die Verfasser stets auf die (wohlwollende) Möglichkeit der „theoretischen Rekonstruktion“ der jeweils betroffenen Ausführungen in den zitierten Quellen.

1.3 Rekonstruktion einer Theorie aus Perspektive des „statement view“

1.3.1 Das klassische Transportkostenmodell

Das „klassische“ Transportkostenmodell[90] für die Auswahl eines „optimalen“ – d. h. hier transportkostenminimalen – Standorts stellt ein „altehrwürdiges“ betriebswirtschaftliches Auswahl- oder Entscheidungsmodell dar, das seit etwas mehr als hundert Jahren (!) im Hinblick auf seine maßgeblichen „Schöpfer“’ als STEINER-WEBER-Modell[91] weithin bekannt ist. Es wird im Folgenden nur kurz vorgestellt, weil es sich aus „modellierungstheoretischer“ Hinsicht um ein „banales“ Modell handelt.[92] Dennoch bietet es interessante Ansatzpunkte, um die „theoretische Fundierung“ und die „Rekonstruktion eines theoretischen Fundaments“ eines Modells im Sinne einer „Miniaturtheorie“ näher zu untersuchen.

Das „klassische“ Transportkostenmodell mit K_T als Transportkosten, k_T als Transportkostensatz [€/Tonnenkilometer], m_n als zu transportierender Gütermenge [Tonnen] zwischen fest vorgegebenen N Beschaffungs- oder Absatzorten O_n mit $n = 1, \dots, N$ mit den Koordinaten (x_n, y_n) in einem zweidimensionalen kartesischen Koordinatensystem und mit (x_S, y_S) als Koordinaten des auszuwählenden transportkostenminimalen („optimalen“) Standorts O_S einer Betriebsstätte, wie z. B. einer Fabrik, lässt sich hinsichtlich seiner Zielfunktion wie folgt darstellen:[93]

$$K_T(x_S, y_S) = k_T \cdot \sum_{n=1}^{N} \left(m_n \cdot \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2} \right) \rightarrow min! \tag{1.1}$$

90 Vgl. Eiselt und Sandblom (2022), S. 268 u. 281 ff.; Hohmann (2022), S. 114 ff.; Corsten und Gössinger (2016), S. 418 ff.; Chopra und Meindl (2014), S. 166 ff.; Domschke und Scholl (2008), S. 174 ff.; Hansmann (2006), S. 112 ff.; Domschke (1996), Sp. 1916 ff.; Domschke und Drexl (1996), S. 164 u. 167 ff.; Zahn und Schmid (1996), S. 305 ff.; Steiner (1993), S. 122 ff.; Kappler und Rehkugler (1991), S. 227 (sehr knapp); Lüder (1990), S. 40 ff.; Melzer (1979), S. 142 f.; Hansmann (1974), S. 23 ff.

91 Vgl. zur Bezeichnung STEINER-WEBER-Modell, STEINER-WEBER-Problem, STEINER-WEBER-Ansatz u. ä. Komposita Hohmann (2022), S. 114; Corsten und Gössinger (2016), S. 418; Domschke und Scholl (2008), S. 174; Hansmann (2006), S. 113; Domschke und Drexl (1996), S. 168; Domschke (1996), Sp. 1917; Zahn und Schmid (1996), S. 305; Steiner (1993), S. 123; Lüder (1990), S. 40; Melzer (1979), S. 142; Katz (1974), S. 89. Zuweilen wird auch von einem FERMAT-WEBER-Modell gesprochen; vgl. beispielsweise Beck und Sabach (2015), S. 2 u. 4; Brimberg (2003), S. 199 f.; Cánovas u. a. (2002), S. 327; Brimberg (1995), S. 71 f.; Chandrasekaran und Tamir (1989), S. 293. In diesem Fall steht kein Transportkostenmodell im Vordergrund, weil vom Transportkostensatz abstrahiert wird und die Transportmengen nur als „dimensionslose“ Gewichte von Strecken (Abständen) zwischen vorgegebenen Standorten („destinations“) und einem gesuchten „zentralen“ Standort zwischen diesen vorgegebenen Standorten erfasst werden. Das spezielle Problem, die Summe der (ungewichteten) Abstände zwischen lediglich drei vorgegebenen Standorten und einem gesuchten „zentralen“ Standort zu minimieren, wurde bereits von FERMAT im 17. Jahrhundert grundsätzlich formuliert („FERMAT’s problem“). Später wurde es von WEBER (und Dritten) auf die Suche nach einem „zentralen“ Standort in Bezug auf beliebig viele vorgegebene Standorte sowie auf Gewichte für die Abstände zwischen diesen Standorten und dem gesuchten „zentralen“ Standorten verallgemeinert („generalized FERMAT location problem“). Außerdem wurde die Modellinterpretation im Hinblick auf die Auswahl eines transportkostenminimalen „zentralen“ Standorts erweitert, indem vor allem ein zusätzlicher Transportkostensatz als konstanter Faktor in der Zielfunktion ergänzt und die Gewichte als Transportmengen aufgefasst wurden. Des Weiteren ist auch von einem FERMAT-TORRICELLI-Problem die Rede, weil das ursprünglich von FERMAT formulierte spezielle Problem erstmals von TORRICELLI gelöst wurde; vgl. Mordukhovich und Nam (2019), S. 205 f. u. 211.

92 Nur am Rande sei erwähnt, dass das klassische Transportkostenmodell trotz seiner „Banalität“ erhebliche Schwierigkeiten hinsichtlich seiner analytischen Lösbarkeit aufweist (für mehr als vier Beschaffungs- und Absatzorte lässt es sich nicht mehr mittels einer „analytischen“ Formel lösen).

93 Vgl. Hohmann (2022), S. 115; Corsten und Gössinger (2016), S. 418; Chopra und Meindl (2014), S. 166; Domschke und Scholl (2008), S. 174; Hansmann (2006), S. 114; Steiner (1993), S. 123; Lüder (1990), S. 41; Melzer (1979), S. 142.

Diese Zielfunktion drückt aus, dass die Transportkosten zwischen allen Beschaffungs- und Absatzorten O_n einerseits sowie dem gesuchten optimalen Standort O_S so gering wie möglich ausfallen sollen. Die Transportkosten werden anhand der zu transportierenden Gütermengen m_n und der zurückzulegenden Transportstrecken s_n [Kilometer] ermittelt. Die Transportstrecken s_n werden mittels der euklidischen Distanz

$$s_n = \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}$$

zwischen einem Beschaffungs- oder Absatzort O_n und dem gesuchten optimalen Standort O_S in einem „idealisierten“ zweidimensionalen Koordinatensystem ermittelt. Zur Veranschaulichung der Zielfunktion und der euklidischen Distanz dient die nachfolgende Abbildung 1.1 mit zwei Beschaffungs- und zwei Absatzorten O_n ($n = 1, \dots, 4$), dem gesuchten optimalen Standort O_S und einer exemplarischen Transportstrecke s_2 zwischen dem Absatzort O_2 und dem gesuchten optimalen Standort O_S.

In den meisten „lehrbuchartigen“ Beiträgen, die sich mit dem klassischen Transportkostenmodell befassen, bleibt es bei der o. a. Zielfunktion, die Transportkosten zu minimieren.[94] Sie stellt die „subjektive“[95] Komponente des Optimierungsmodells dar. Eine solche Modellformulierung erweist sich jedoch als unvollständig, weil die „quasi-objektive“[96] Komponente mit den Restriktionen fehlt, welche den Entscheidungsspielraum des Entscheidungsträgers begrenzen. Im einfachsten Fall sind mit $\mathbb{R}$ als Menge der reellen Zahlen die Definitionsbereiche der beiden Entscheidungsvariablen x_S und y_S sowie die üblichen Nichtnegativitätsbedingungen (wegen der Beschränkung auf den ersten Quadranten des zugrunde liegenden zweidimensionalen kartesischen Koordinatensystems) als Restriktionen zu ergänzen:

$$x_S \in \mathbb{R} \wedge x_S \geq 0 \tag{1.2a}$$

$$y_S \in \mathbb{R} \wedge y_S \geq 0 \tag{1.2b}$$

Zusätzlich kann das analytische Wissen („a priori“) genutzt werden, dass der gesuchte optimale Standort O_S nur innerhalb des „Lösungspolygons“ oder auf dessen Grenzen liegen kann.[97] In der o. a. Abbildung 1.1[98] wird das Lösungspolygon durch die vier vorgegebenen Beschaffungs- und Absatzorte O_1 bis O_4 aufgespannt. Dieses Wissen lässt sich mittels folgender Ungleichungen mit $a_n, b_n, c_n \in \mathbb{R}$ und $c_n \geq 0$ für die (höchstens) N Grenzen des Lösungspolygons ausdrücken:[99]

$$(x_S, y_S) \in \{(x, y) \,|\, a_n \cdot x + b_n \cdot y \leq / \geq c_n \text{ für } n = 1, \dots, N\} \tag{1.3}$$

94 Vgl. z. B. Domschke und Scholl (2008), S. 174; Zahn und Schmid (1996), S. 307; Kappler und Rehkugler (1991), S. 227; Melzer (1979), S. 142; Hansmann (1974), S. 27.

95 Vgl. Zelewski u. a. (2008), S. 270 ff. u. 332.

96 Vgl. Zelewski u. a. (2008), S. 273 ff. u. 332.

97 Dieser Sachverhalt lässt sich leicht auf indirekte Art beweisen: Hypothetisch sei angenommen, der „optimale“ Standort läge außerhalb des Lösungspolygons. Dann lässt sich immer ein alternativer Standort auf der nächstgelagerten Grenze des Lösungspolygons finden (z. B. durch Fällen eines Lots zwischen dem hypothetischen Standort und der Lösungspolygongrenze), der bezüglich aller Beschaffungs- und Absatzorte geringere Distanzen und somit auch geringere Transportkosten aufweist. Folglich kann der hypothetisch angenommene Standort nicht transportkostenminimal und somit auch nicht optimal sein (q.e.d.).

98 Vgl. zu ähnlichen grafischen Darstellungen Hohmann (2022), S. 115 (stark vereinfacht); Corsten und Gössinger (2016), S. 419; Zahn und Schmid (1996), S. 307; Steiner (1993), S. 123.

99 Im nachfolgenden Ungleichungssystem lassen sich die Ungleichungsoperatoren „≤“ versus „≥“ nicht allgemeingültig festlegen, weil sie davon abhängen, ob eine Grenze des Lösungspolygons dieses „nach oben“ oder „nach rechts“ (≤) bzw. „nach unten“ oder „nach links“ (≥) begrenzt.

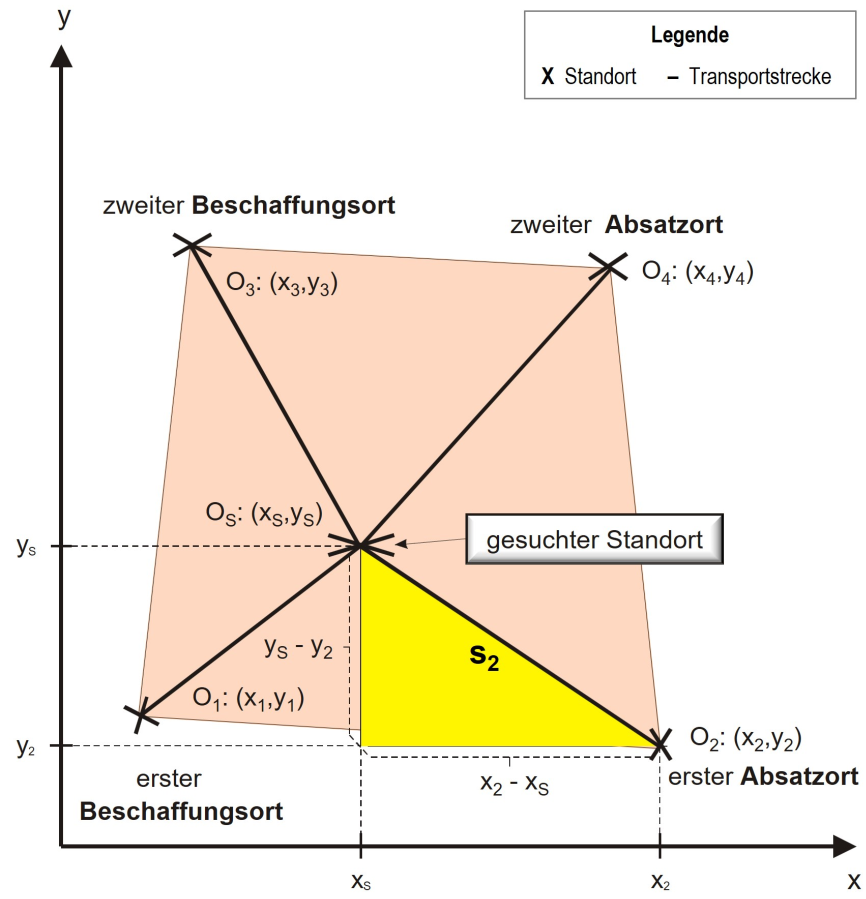


Abbildung 1.1: Skizze zur Ermittlung eines optimalen – transportkostenminimalen – Standorts

Am Rande sei angemerkt, dass das Ungleichungssystem der Formel 1.3 die Formeln 1.2a und 1.2b für die Definitionsbereiche und Nichtnegativitätsbedingungen der beiden Entscheidungsvariablen x_S und y_S impliziert, sodass auf die Formeln 1.2a und 1.2b verzichtet werden kann, falls die Formel 1.3 zur Bestimmung der „quasi-objektiven“ Komponente des klassischen Transportkostenmodells herangezogen wird.

Exkurs zur Komplexität und Relevanz des klassischen Transportkostenmodells

Das klassische Transportkostenmodell, dass durch die Formeln 1.1 sowie 1.2a und 1.2b oder 1.3 spezifiziert wird, erweist sich aufgrund seines Alters von mehr als 100 Jahren (!) nicht nur als „altehrwürdig“, sondern prima facie auch als sehr „schlicht“.[100] Daher könnte

100 Vgl. Domschke und Scholl (2008), S. 174. In ähnlicher Weise spricht Melzer (1979), S. 142, von einer „zwar einfachen, aber doch hinreichend interessanten und schönen Struktur“ des klassischen Transportkostenmodells (Steiner-Weber-Problems).

eingewandt werden, es würde sich kaum für einen wissenschaftlichen Beitrag zur Erörterung des wissenschaftlichen Theorieverständnisses eignen. Einen solchen Einwand erachten die Verfasser jedoch aus mehreren Gründen als nicht stichhaltig.

Zunächst sei angeführt, dass sich weithin bekannte und schlichte (Modell-)Beispiele in der Regel besonders gut dafür eignen, „anspruchsvollere“ Sachverhalte wie ein Theorieverständnis und eine entsprechende Theorierekonstruktion (für das klassische Transportkostenmodell) zu verdeutlichen. Denn die Bekanntheit und Schlichtheit eines Modells gestattet es, von detaillierten Modelldiskussionen abzusehen und sich stattdessen auf dessen theoretische Rekonstruktion sowie das zugrunde liegende Theorieverständnis zu fokussieren.

Darüber hinaus erweist sich das klassische Transportkostenmodell als keineswegs so schlicht, wie es angesichts seiner oben vorgestellten Spezifizierung auf den ersten Blick erscheinen mag. Stattdessen erweist sich dieses Modell trotz der wenigen und einfach anmutenden Formeln für die Modellspezifizierung als erstaunlich komplex, sobald seine Lösung zur Bestimmung eines optimalen (kostenminimalen) Standorts betrachtet wird. Ohne in der hier gebotenen Kürze auf die Details ihrer Herleitung einzugehen,[101] lauten die Formeln zur Ermittlung der Koordinaten (x_S, y_S) des auszuwählenden optimalen Standorts O_S:[102]

$$x_S = \frac{\sum_{n=1}^{N} \frac{m_n \cdot x_n}{\sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}}}{\sum_{n=1}^{N} \frac{m_n}{\sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}}} \qquad \text{sofern } \forall n = 1, \ldots, N : (x_S, y_S) \neq (x_n, y_n) \qquad (1.4a)$$

$$y_S = \frac{\sum_{n=1}^{N} \frac{m_n \cdot y_n}{\sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}}}{\sum_{n=1}^{N} \frac{m_n}{\sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}}} \qquad \text{sofern } \forall n = 1, \ldots, N : (x_S, y_S) \neq (x_n, y_n) \qquad (1.4b)$$

101 Da die Zielfunktion der zu minimierenden Transportkosten zwei Entscheidungsvariablen x_S und y_S umfasst, sind als notwendige Bedingungen für die Existenz eines lokalen Kostenminimums die zwei partiellen Ableitungen der Zielfunktion nach x_S und y_S jeweils gleich Null zu setzen. Dies liefert nach einigen Berechnungen (z. B. Anwendung der Kettenregel für Ableitungen einschließlich des „lästigen“ Wurzelterms in der Zielfunktion) bereits die beiden o. a. Formeln. Das klassische Transportkostenmodell erweist sich als „gutmütig“, weil die notwendigen Bedingungen für die Existenz eines lokalen Kostenminimums zugleich hinreichende Bedingungen hierfür darstellen und weil es sich beim lokalen Kostenminimum zugleich um ein globales Kostenminimum handelt. Allerdings muss bei den o. a. Formeln der Sonderfall ausgeschlossen werden, dass der gesuchte optimale Standort zufällig mit einem der Beschaffungs- oder Absatzorte zusammenfällt, weil in diesem Fall mindestens einer der Nenner der o. a. Quotiententerme den unzulässigen Wert Null annehmen würde. Für diesen Sonderfall ist eine separate, aber unkomplizierte „Nebenrechnung“ durchzuführen.

102 Vgl. Eiselt und Sandblom (2022), S. 284; Hohmann (2022), S. 117; Corsten und Gössinger (2016), S. 420; Hansmann (2006), S. 115; Domschke und Drexl (1996), S. 169; Zahn und Schmid (1996), S. 311; Lüder (1990), S. 42; Melzer (1979), S. 168. Die Bedingungen $(x_S, y_S) \neq (x_n, y_n)$, die in den vorgenannten Quellen nicht explizit angeführt werden, sind aus zwei Gründen außerordentlich wichtig. Erstens schließen sie in den Formeln 1.4a und 1.4b die mathematisch unzulässige Division durch Null für den Sonderfall $(x_S, y_S) = (x_n, y_n)$ aus. Zweitens – und dies ist viel wichtiger – stellt es sich für den nachfolgend vorgestellten Näherungsalgorithmus zur Lösung des klassischen Transportkostenproblems als eine gravierende Herausforderung dar, wie sich dieser Näherungsalgorithmus während einer seiner zahlreichen Iterationen verhält, wenn die Zwischenwerte der Koordinaten für den gesuchten optimalen Standort „zufällig“ mit den Koordinaten (x_n, y_n) für einen der vorgegebenen Beschaffungs- oder Absatzorte (in der Fachliteratur oftmals als „vertices“, „destinations“ oder „anchors“ bezeichnet) O_n mit $n = 1, \ldots, N$ übereinstimmen. Denn im Falle einer solchen Übereinstimmung kann es dazu kommen, dass der Näherungsalgorithmus mit einem der Koordinaten (x_n, y_n) für einen der vorgegebenen Beschaffungs- oder Absatzorte abbricht, obwohl es sich nicht um den gesuchten kostenminimalen Standort handelt.

Wegen der Interdependenzen der beiden Entscheidungsvariablen x_S und y_S, die in beiden Formeln sowohl auf der linken als auch auf der rechten Gleichungsseite „unauflösbar“ vorkommen, und wegen der Wurzelterme in den Nennern der beiden Quotiententerme lässt sich das Gleichungssystem der Formeln 1.4a und 1.4b für mehr als vier Beschaffungs- und Absatzorte (also $N \geq 5$) nicht mehr analytisch lösen.[103]

Stattdessen lässt sich im allgemeinen Fall nur ein heuristischer Näherungsalgorithmus[104] zur Ermittlung des optimalen Standorts mit den Koordinaten (x_S, y_S) anwenden, der auf Arbeiten von Weiszfeld[105] und Miehle[106] zurückreicht[107]. Er stützt sich auf die beiden Bedingungen 1.4a und 1.4a. Dieser Näherungsalgorithmus erweist sich jedoch als „bösartig“, weil er bei „beliebig langer“ Ausführungszeit *nicht* – wie es bei „gutartigen“ Näherungsalgorithmen der Fall ist – gegen das exakte („theoretische“, weil bei der Algorithmusausführung unbekannte) Optimum konvergiert.[108] Vielmehr existieren abzählbar („denumerable“) unendlich viele „pathologische“ Fälle[109] für das klassische Transportkostenmodell, in denen sich der Näherungsalgorithmus für einen (in Abhängigkeit von der numerischen Konstellation des jeweils betrachteten Transportkostenmodells) „ungünstig“ ausgewählten Startpunkt nicht konvergent verhält, sondern in einem der vorgegebenen Beschaffungs- und Absatzorte als einer suboptimalen Lösung verharrt.

103 Vgl. Melzer (1979), S. 143.

104 Vgl. Eiselt und Sandblom (2022), S. 284 f.; Hohmann (2022), S. 117 ff.; Mordukhovich und Nam (2019), S. 211 ff.; Corsten und Gössinger (2016), S. 420 ff.; Chopra und Meindl (2014), S. 168; Katz und Vogl (2010), S. 400 ff.; Domschke und Scholl (2008), S. 175; Hansmann (2006), S. 115 f.; Cánovas u. a. (2002), S. 327; Domschke und Drexl (1996), S. 169 ff.; Zahn und Schmid (1996), S. 311 f.; Brimberg (1995), S. 72 ff.; Steiner (1993), S. 124; Lüder (1990), S. 40 u. 42 f.; Chandrasekaran und Tamir (1989), S. 293 f.; Melzer (1979), S. 168; Katz (1974), S. 90; Kuhn (1973), S. 99 ff.

105 Vgl. Weiszfeld (1937), S. 355 ff. (damals erst 16-jährig). Vgl. auch Beck und Sabach (2015), S. 2 f. u. 4 f., insbesondere S. 6 ff.

106 Vgl. Miehle (1958), S. 237 ff.

107 Vgl. Eiselt und Sandblom (2022), S. 284; Mordukhovich und Nam (2019), S. 205; Beck und Sabach (2015), S. 1, 2 f., 8 u. 31; Domschke und Scholl (2008), S. 175; Domschke und Drexl (1996), S. 169; Brimberg (1995), S. 72; Lüder (1990), S. 40; Melzer (1979), S. 168; Kuhn (1973), S. 99 u. 101.

108 Dennoch wird des Öfteren die – angebliche – Konvergenz des Näherungsalgorithmus gegen das „theoretische“ Optimum ohne Einschränkungen behauptet; vgl. Hohmann (2022), S. 120 („dass die Approximation an das Optimum mit jeder weiteren Iteration größer wird“); Hansmann (2006), S. 116; Zahn und Schmid (1996), S. 311; Steiner (1993), S. 124 („Ein Beweis für die Konvergenz der Iteration gegen den optimalen Standort liegt vor“), und zwar unter Berufung auf Kuhn (1973); Lüder (1990), S. 40 („Konvergenzbeweis“) u. 43 („Die Konvergenz des Verfahrens ist gesichert [...].“). Ein genaueres Studium der Fachliteratur zeigt jedoch, dass der Näherungsalgorithmus im strengen Sinne nicht konvergiert, obwohl diese Konvergenz – teilweise unter Verweis auf „Beweise“ – zu Unrecht behauptet wird. Darauf wird unmittelbar nachfolgend eingegangen.

109 Vgl. zur „eingeschränkten“ Konvergenz des Näherungsalgorithmus, weil er für abzählbar („denumerable“) unendlich viele, von den Verfassern als „pathologisch“ bezeichnete Ausnahmefälle (Startpunkte) nicht konvergiert, beispielsweise Mordukhovich und Nam (2019), S. 205; Beck und Sabach (2015), S. 9 u. 10 (allerdings mit einem Vorbehalt gegenüber dem „Beweis“ in Kuhn (1973), die Anzahl der Ausnahmefälle von der Konvergenz sei „denumerable“) sowie S. 14 ff.; Katz und Vogl (2010), S. 403 u. 408; Brimberg (2003), S. 199 f., 201 u. 206; Cánovas u. a. (2002), S. 328; Brimberg (1995), S. 72 (kritisch), 73 u. 75 f.; Chandrasekaran und Tamir (1989), S. 294; Katz (1974), S. 89 f. u. 96; Kuhn (1973), S. 101 ff. (Konvergenzbeweis), insbesondere S. 104 f. (Konvergenztheorem) und S. 106 f. (Ausnahmefälle); Hansmann (1974), S. 27. Von Domschke und Drexl (1996), S. 169, wird immerhin in recht allgemeiner Weise auf „Konvergenzprobleme“ hingewiesen; vgl. zu einer solchen Andeutung auch Domschke und Scholl (2008), S. 175.

Diese knappen Erläuterungen zur Lösung des klassischen Transportkostenmodells sollten hinreichend verdeutlicht haben, dass es sich bei diesem Transportkostenmodell nur prima facie um ein „schlichtes“ Modell handelt. Wenn der Aspekt der Modelllösung einbezogen wird, dann handelt es sich ganz im Gegenteil um ein Modell mit nicht zu unterschätzender Komplexität. Daher stellt es keineswegs ein triviales Beispiel dar, um die Rekonstruktion der Theorie, die diesem Modell zugrunde liegt, sowie das zugehörige Theorieverständnis zu verdeutlichen. Schließlich ist vor dem Hintergrund der Debatte um „rigor versus relevance“ dem Einwand entgegenzutreten, das klassische Transportkostenmodell besitze keine Relevanz, weil es von zu vielen Determinanten der betrieblichen Realität abstrahiere. Zwar trifft es zu, dass das klassische Transportkostenmodell auf zahlreichen vereinfachenden Prämissen beruht. Dies betrifft beispielsweise die Verwendung von nur einem und konstanten Transportkostensatz (Abstrahierung von unterschiedlichen Transportmitteln und von Kosteneinflüssen der Art der Transportstrecken, wie etwa Transporten im Gebirge versus im Flachland) und die Voraussetzung eines „flachen“ zweidimensionalen Koordinatensystems für die Lokalisierung der Standorte. Aber trotz dieser Prämissen lässt sich das klassische Transportkostenmodell auf zahlreiche betriebswirtschaftlich relevante Standortprobleme „direkt“ oder zumindest in „angepasster“ Form anwenden.[110] Hierfür seien nur einige wenige Beispiele genannt.

Für die Auswahl eines Zentrallagers im EU-Raum wurden nach der EU-weiten Liberalisierung des grenzenüberschreitenden Güterverkehrs infolge des Cecchini-Reports[111] Standortplanungsmodelle eingesetzt, die der Struktur des klassischen Transportkostenmodells grundsätzlich entsprechen (allenfalls um weitere Formalziele und zusätzliche Restriktionen ergänzt wurden). Für die Lokalisierung von „Verteilerkästen“ zur Anbindung von Wohnvierteln an die Telekommunikationsinfrastruktur lässt sich das klassische Transportkostenmodell sogar in „vereinfachter“ Form anwenden, indem lediglich die Summe der „Transportstrecken“ für den Datentransfer (also die Summe der Kabellängen) minimiert wird, sodass von den Datentransportmengen abgesehen werden kann. Ein spezielles, aber sehr anschauliches Beispiel für die Relevanz des klassischen Transportkostenmodells stellt die Errichtung eines „Heliports“ am Standort HP_0 dar. Er dient der Stationierung eines Rettungshubschraubers, um Krankentransporte von z. B. drei Unfallorten UO_1, UO_2 und UO_3 (als „Beschaffungsorten“) zu einem Universitätsklinikum UK_4 (als „Absatzort“) durchzuführen. In diesem Fall erweist sich die Verwendung der „abstrakten“ euklidischen Distanz zur Entfernungsmessung sogar als realitätsadäquat, weil sich die Flugstrecken des Rettungshubschraubers als „Luftlinien“ zwischen jeweils zwei Standorten messen lassen. Ohne auf weitere Details einzugehen,[112] veranschaulicht Abbildung 1.2 die Struktur des Standortproblems für die Stationierung eines Rettungshubschraubers.

110 Vgl. Melzer (1979), S. 142 f.

111 Vgl. Cecchini u. a. (1988a), insbesondere S. 3 ff., 8 ff., 31 ff., 71 ff. u. 92 f. (Abbau von Grenzformalitäten); Cecchini u. a. (1988b), insbesondere S. 15 ff., 54 ff., 100 ff. u. 125 f. (Abbau von Grenzformalitäten). Vgl. darüber hinaus auch die „Folgeanalysen“ in Hafner u. a. (2014), insbesondere S. 7 ff., 18 ff. u. 25 ff.; Ząbkowicz (2015), S. 504 ff., insbesondere S. 513.

112 Beispielsweise kann sich die Zielfunktion auf die Transportstrecken, auf die Transportzeiten oder die Transportkosten beziehen. Die Berücksichtigung von Transportstrecken entspricht dem o. a. Standortproblem für die Lokalisierung von „Verteilerkästen“. In diesem Fall kann die Zielfunktion – je nach Erkenntnisinteresse des Entscheidungsträgers – nur die Hinflugstrecken vom Heliport zu den Unfallorten, sowohl die Hinflugstrecken vom Heliport zu den Unfallorten als auch die Weiterflugstrecken von den Unfallorten zum Universitätsklinikum, sowohl die Hinflugstrecken vom Heliport zu den Unfallorten als auch die Rückflugstrecken vom Universitätsklinikum zum Heliport oder die Summe aus den vorgenannten Hin-, Weiter- und Rückflugstrecken umfassen.

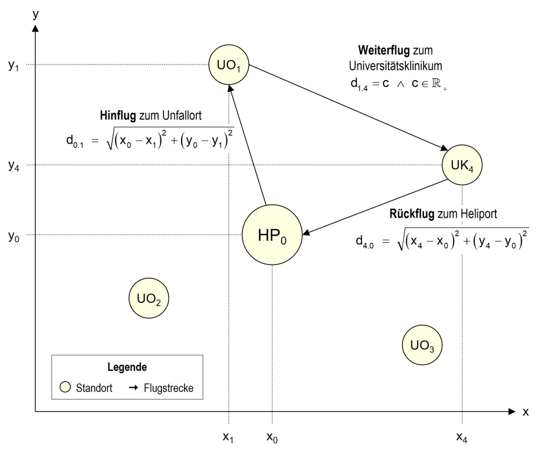

y
y_1
y_4
y_0
UO_1
Weiterflug zum Universitätsklinikum
$d_{1,4} = c \wedge c \in \mathbb{R}_+$
Hinflug zum Unfallort
$d_{0,1} = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$
UK_4
HP_0
Rückflug zum Heliport
$d_{4,0} = \sqrt{(x_4 - x_0)^2 + (y_4 - y_0)^2}$
UO_2
UO_3
Legende
○ Standort
→ Flugstrecke
x
x_1
x_0
x_4

Abbildung 1.2: Stationierung eines Rettungshubschraubers mit der Struktur des klassischen Transportkostenmodells

Als letztes relevanzverdeutlichendes Beispiel dient eine Erweiterung des klassischen Transportkostenmodells mit nur einem gesuchten optimalen Standort auf ein Multi-Standortproblem. Es betrifft die Festlegung von mehreren Standorten für das Laden der Batterien von Bussen mit Elektroantrieb im Gebiet eines Öffentlichen Nahverkehrsnetzes.[113] Hierbei gilt es hinsichtlich der Busfahrzeiten die Verträglichkeit der Batterieladezeiten an den Ladestationen (vornehmlich an den Start- oder Endhaltestellen der Buslinien) mit dem Busfahrplan zu beachten. Außerdem müssen die Fahrstrecken der Busse zwischen zwei Ladestationen die Restriktion einhalten, dass die Kapazitäten ihrer Batterien für diese Fahrstrecken ausreichen (in der Regel zuzüglich eines Sicherheitszuschlags). Darüber hinaus kann z. B. die Mehrbelastung der Batterien, die im Winterbetrieb für die Busheizung entsteht, berücksichtigt werden. Noch komplexer fällt das kombinierte Fahrzeiten- und Fahrstreckenmodell aus, wenn die Anzahl der Ladestationen nicht fest vorgegeben ist, sondern im Hinblick auf das Ziel, die Gesamtkosten des Busbetriebs zu minimieren, selbst Gegenstand der Ermittlung einer optimalen Modelllösung sein kann. Die voranstehenden Beispiele haben hoffentlich „hinreichend“ verdeutlicht, dass das klassische Transportkostenmodell mit seiner Grundstruktur so leistungsfähig ist, dass es sich – entweder direkt oder nach entsprechenden Anpassungen – auf eine Vielzahl von betriebswirtschaftlich relevanten Problemen anwenden lässt. Es sollte daher trotz seiner „Altehrwürdigkeit“ und scheinbaren „Schlichtheit“ nicht unterschätzt werden.

Durch die Berücksichtigung von Transportzeiten lassen sich reale Nebenbedingungen erfassen, wie etwa die Einhaltung einer maximalen Transportzeit zwischen dem Start des Rettungshubschraubers am Heliport und dem Landen des Rettungshubschraubers mit einem transportierten Unfallopfer am Universitätsklinikum. Für diesen Fall sind als Zielfunktion die Hinflugstrecken vom Heliport zu den Unfallorten und die Weiterflugstrecken von den Unfallorten zum Universitätsklinikum zu erfassen und jeweils mit der Fluggeschwindigkeit des Rettungshubschraubers zu multiplizieren, die an die Stelle des Transportkostensatzes im klassischen Transportkostenmodell tritt. Zur Ermittlung der Transportkosten kann auf die zuvor erwähnten Transportzeiten zurückgegriffen werden (jetzt allerdings für Hin-, Weiter- und Rückflugstrecken), die mit den Betriebskosten des Rettungshubschraubers z. B. je Transportminute (als modifiziertem Transportkostensatz, vor allem im Hinblick auf den Treibstoffverbrauch) zu multiplizieren sind.

113 Vgl. Kunith u. a. (2016), S. 1 f., 4 f. u. 7 ff., insbesondere Figure 3 auf S. 11.

1.3.2 Rekonstruktion einer „Miniaturtheorie“ für das klassische Transportkostenmodell

Die Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, lässt sich anhand des starken nomisch-formalsprachlichen Theorieverständnisses rekonstruieren.[114] Zu diesem Zweck wird zunächst das klassische Transportkostenmodell so aufbereitet, dass es sich als ein *systematischer Aussagenzusammenhang* darstellen lässt. Mithilfe der bereits eingeführten aussagen- und prädikatenlogischen Notationen für Junktoren bzw. Quantoren sowie mit $OPT(x_S, y_S)$ als Prädikat für die Optimalität eines Standorts O_S mit den Koordinaten (x_S, y_S) resultiert die nachfolgende formallogische Rekonstruktion des klassischen Transportkostenmodells.

$$\begin{aligned}&\forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R} :\\ &OPT(x_S, y_S) \leftrightarrow \neg \left(\exists\, \widehat{x_S} \in \mathbb{R}\, \exists\, \widehat{y_S} \in \mathbb{R} : K_T(\widehat{x_S}, \widehat{y_S}) < K_T(x_S, y_S)\right)\end{aligned} \tag{1.5}$$

$$\begin{aligned}&\wedge \forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R} :\\ &K_T(x_S, y_S) = k_T \cdot \Big(\sum_{n=1}^{N} m_n \cdot s\big((x_S, y_S), (x_n, y_n)\big)\Big)\end{aligned} \tag{1.6}$$

$$\begin{aligned}&\wedge \forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R}\, \forall x_n \in \mathbb{R}\, \forall y_n \in \mathbb{R} :\\ &s\big((x_S, y_S), (x_n, y_n)\big) = \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2}\end{aligned} \tag{1.7}$$

$$\wedge \forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R} : x_S \geq 0 \wedge y_S \geq 0 \tag{1.8}$$

$$\begin{aligned}&\wedge \forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R} : OPT(x_S, y_S) \rightarrow \ldots\\ &(x_S, y_S) \in \big\{\{(x, y) \,|\, a_n \cdot x + b_n \cdot y \leq / \geq c_n \text{ für } n = 1, \ldots, N\}\big\}\end{aligned} \tag{1.9}$$

Die Formel 1.5 drückt die Optimalitätsbedingung aus, dass ein Standort O_S mit den Koordinaten (x_S, y_S) genau dann kostenminimal ist, wenn kein alternativer Standort $O_{\widehat{S}}$ mit den Koordinaten $(\widehat{x_S}, \widehat{y_S})$ existiert, für den die Transportkosten K_T echt kleiner sind als für den Standort O_S.[115] Die Formel 1.6 definiert die Transportkosten K_T in Abhängigkeit von den transportierten Mengen m_n und den Transportstrecken $s((x_S, y_S), (x_n, y_n))$, die ihrerseits von den Koordinaten (x_S, y_S) und (x_n, y_n) des gesuchten Standorts O_S bzw. eines Beschaffungs- oder Absatzorts O_n mit $n = 1, \ldots, N$ abhängen. Die Formel 1.7 spezifiziert als Messvorschrift für die Transportstrecken die euklidische Distanz zwischen zwei Standorten in einem zweidimensionalen kartesischen Koordinatensystem. Die Formel 1.8 expliziert die Nichtnegativitätsbedingungen für die Koordinaten x_S und y_S des gesuchten optimalen

114 Die Theorierekonstruktion ist auch anhand der anderen eingangs angesprochenen Theorieverständnisse möglich. Im Folgenden wird aber von vornherein das anspruchsvollste Theorieverständnis zugrunde gelegt.

115 Diese Formulierung, die auf den ersten Blick ungewöhnlich erscheinen mag, ist im Rahmen der Prädikatenlogik erforderlich, weil dieses logische Kalkül keinen mathematischen Minimierungsoperator wie „→ *min*!“ umfasst.

Standorts O_S.[116] Die Formel 1.9 drückt das zusätzliche Wissen aus, dass ein Standort O_S nur dann optimal sein kann, wenn er innerhalb oder auf den Grenzen des Lösungspolygons liegt. Wie bereits an früherer Stelle erläutert, ist es für die Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, nicht erforderlich (aber auch nicht schädlich), die beiden Formeln 1.8 und 1.9 gemeinsam zu verwenden. Stattdessen reicht es aus, entweder nur die Formel 1.8 oder nur die Formel 1.9 – die ihrerseits die Formel 1.8 impliziert – zu verwenden.

Die formallogisch rekonstruierte Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, umfasst also fünf (oder, wie bereits erläutert, vier) Aussagen im Sinne der o. a. allquantifizierten prädikatenlogischen Formeln 1.5 bis 1.9, die mittels des Junktors „$\wedge$“ für das aussagenlogische Konjugat („und“) miteinander verknüpft sind. Folglich liegt ein *systematischer Aussagenzusammenhang* vor, der die Konstituente „formallogisch“ des starken nomisch-formallogischen Theorieverständnisses erfüllt. Aus der Perspektive dieses Theorieverständnisses verbleibt also „nur“ noch die Aufgabe nachzuweisen, dass das System der Formeln 1.5 bis 1.9 auch mindestens eine *nicht-triviale nomische Hypothese* umfasst. Dieser Nachweis stellt, wie nachfolgend erläutert, eine nicht zu unterschätzende Herausforderung für die Theorierekonstruktion dar.

Um die Übersichtlichkeit der prädikatenlogischen Formeln zu fördern, wird im Folgenden auf die Ergänzung der Definitionsbereiche für die variablen Terme hinter den Allquantoren verzichtet. Da mindestens eine nicht-triviale nomische Hypothese in dem systematischen Aussagenzusammenhang der Formeln 1.5 bis 1.9 gesucht ist, reicht es aus, sich auf allquantifizierte Formeln zu beschränken. Diese Fokussierung hilft hier jedoch nicht weiter, weil für jede der vorgenannten fünf Formeln ihre variablen Terme jeweils durch einen Allquantor gebunden wird. Daher muss jede dieser Formeln hinsichtlich ihrer Eignung als nicht-triviale nomische Hypothese untersucht werden. Im Folgenden wird von der o. a. Reihenfolge der Formeln 1.5 bis 1.9 etwas abgewichen, um von „einfachen“ zu „komplexeren“ Kandidaten für eine nicht-triviale nomische Hypothese (kurz: NTNH) voranzuschreiten.

Als erster NTNH-Kandidat kommt die Formel 1.6 in Betracht. In vereinfachter, von den Definitionsbereichen abstrahierender Notation lautet sie „der Erinnerung halber“:

$$\forall x_S \forall y_S \,:\, K_T(x_S, y_S) = k_T \cdot \Big(\sum_{n=1}^{N} m_n \cdot s\big((x_S, y_S), (x_n, y_n)\big) \Big) \tag{1.10}$$

Diese Formel scheidet als nicht-triviale nomische Hypothese trotz ihrer Allquantifizierung aus, weil es sich um eine rein *definitorische Festlegung* handelt, wie die Transportkosten K_T für einen Standort O_S mit den Koordinaten (x_S, y_S) berechnet werden. Eine solche Begriffsdefinition ist begriffsanalytisch wahr; sie gilt „a priori“ vor jeder möglichen Erfahrung. Daher besitzt sie aus logisch-erkenntnistheoretischer Perspektive die Eigenschaft der definitorischen Allgemeingültigkeit – oder mit anderen Worten der „Trivialität“.[117]

116 Die reellzahligen Definitionsbereiche der Koordinaten x_S und y_S werden durch die „Reichweiten“ der beiden in Formel 1.8 vorangestellten Allquantoren spezifiziert. Gleiches gilt hinsichtlich dieser Definitionsbereiche für die übrigen Formeln 1.5 bis 1.7 sowie 1.9.

117 Um Missverständnissen vorzubeugen, wird darauf hingewiesen, dass mit dieser „Allgemeingültigkeit“ keineswegs ein empirisch ausgerichteter Wahrheitsanspruch wie in einem essenzialistischen Begriffskonzept gemeint ist. Vielmehr vertreten die Verfasser ein nominalistisches Begriffskonzept, dem zufolge Begriffsdefinitionen unabhängig von „der“ Realität erfolgen und nur innerhalb eines sprachlichen Aussagenzusammenhangs (letztlich willkürlich, allenfalls zweckmäßig) als „definitorisch gültig“ oder – hier synonym verstanden: als

Den zweiten NTNH-Kandidaten stellt die Formel 1.9 in abermals vereinfachter Notation dar:

$$\begin{aligned} &\forall x_S \in \mathbb{R}\, \forall y_S \in \mathbb{R} \,:\, OPT(x_S, y_S) \rightarrow \ldots \\ &(x_S, y_S) \in \Big\{\{(x, y) \,|\, a_n \cdot x + b_n \cdot y \leq / \geq c_n \text{ für } n = 1, \ldots, N\}\Big\} \end{aligned} \tag{1.11}$$

Auch diese Formel kommt trotz ihrer Allquantifizierung nicht als nicht-triviale nomische Hypothese in Betracht, weil sich – wie früher kurz erläutert wurde – mittels eines einfachen mathematischen Beweises aufzeigen lässt, dass es zur „Bedingung der Möglichkeit eines optimalen Standorts“ gehört, dass dieser Standort innerhalb oder auf den Grenzen des Lösungspolygons liegt. Es handelt sich in diesem Fall also um eine deduktiv-analytisch wahre mathematische Aussage, die abermals „a priori“ vor jeder möglichen Erfahrung gilt. Daher besitzt sie aus logisch-erkenntnistheoretischer Perspektivität die Eigenschaft der deduktiv beweisbaren Allgemeingültigkeit und erweist sich wiederum als „trivial“.

Als dritter NTNH-Kandidat wird die Formel 1.9 betrachtet, für die in wiederum vereinfachter Notation gilt:

$$\forall x_S\, \forall y_S \,:\, x_S \geq 0 \wedge y_S \geq 0 \tag{1.12}$$

In diesem Fall handelt es sich lediglich um eine Randbedingung für den intendierten Anwendungsbereich[118] der Theorie, die dem klassischen Transportkostenmodell zugrunde liegt. Diese Randbedingung lässt sich nicht unmittelbar in die dichotome erkenntnistheoretische Unterscheidung „trivial“ versus „nicht-trivial“ einordnen. Vielmehr handelt es sich um eine „kontingente“ Aussage, die weder notwendig noch unmöglich ist. Sie betrifft die kontingente Voraussetzung, für die Ermittlung eines optimalen Standortes den ersten Quadranten eines zweidimensionalen kartesischen Koordinatensystems mit nicht-negativen Koordinaten für den gesuchten optimalen Standort als „Rahmen“ (oder „Frame“) für alle weiteren Überlegungen vorauszusetzen.[119] Diese Voraussetzung ist zulässig, aber keineswegs zwingend. Beispielsweise lässt sich auch ein dreidimensionales Koordinatensystem (für die „Erdkugel“) mit sowohl positiven als auch negativen Koordinaten (sowie Null-Koordinaten) für die Unterscheidung zwischen Längengraden „westlich versus östlich von Greenwich“ (einschließlich

„allgemeingültig“ – vorausgesetzt werden. Zur Verdeutlichung sei darauf hingewiesen, dass in einem solchen sprachlichen Aussagenzusammenhang auch abweichende Definitionen des Transportkostenbegriffs als „allgemeingültig“ vorausgesetzt werden können. Dies lässt sich daran erkennen, dass in der o. a. Definition des Transportkostenbegriffs nur die variablen Transportkosten berücksichtigt werden, und zwar nur insoweit, wie sie in Abhängigkeit von Transportmengen oder Transportstrecken linear variieren. Diese definitorische Setzung ist streng genommen willkürlich. Sie kann allenfalls durch ihre „Anschlussfähigkeit“ an etablierte Fachliteratur und hinsichtlich ihrer „Zweckmäßigkeit“ für die Lösung realer, von den Transportkosten dominierter Standortplanungsprobleme gerechtfertigt werden. Aber auch andere Transportkostendefinitionen ließen sich mit gleichem Anspruch auf „definitorische Gültigkeit“ voraussetzen. Dies betrifft beispielsweise Transportkostendefinitionen, die auch fixe Kostenbestandteile einbeziehen, wie z. B. für den Fuhrpark sowie für die – eventuell regional variierenden – Grundstücks- und Gebäudekosten am gesuchten Standort. Ebenso lassen sich Transportkostendefinitionen vorstellen, die in nicht-linearer Weise von Transportmengen oder Transportstrecken abhängen oder sogar weitere Kosteneinflussgrößen einbeziehen. In allen Fällen handelt es sich um „realitätsunabhängige“, a priori wahre und somit erkenntnistheoretisch triviale definitorische Festlegungen des Modell-Designers, die aufgrund ihrer Realitätsunabhängigkeit für eine nicht-triviale nomische Hypothese nicht in Betracht kommen.

118 Auf das spezielle Konstrukt eines „intendierten Anwendungsbereichs“ für eine Theorie wird später etwas ausführlicher zurückgekommen.

119 Die Formel 1.12 lässt sich auf die Koordinaten (x_n, y_n) aller Beschaffungs- oder Absatzorte O_n mit $n = 1, \ldots, N$ erweitern, die ebenfalls mit Allquantoren zu binden sind. Auf diese Erweiterung wird hier im Interesse der Übersichtlichkeit verzichtet.

der Lage exakt auf dem Längengrad von Greenwich als „Nullmeridian“) sowie zwischen Breitengraden „nördlich und südlich des Äquators“ (einschließlich der Lage exakt auf dem Äquator) vorstellen. Solche konzeptionellen Vorentscheidungen hinsichtlich des „Rahmens“ einer Theorieformulierung sind zwar wichtig für das Modell- oder Theorie-Design, entziehen sich jedoch der Unterscheidung zwischen empirisch gehaltvollen (nicht-trivialen) und begrifflich oder mathematisch „a priori“ wahren (trivialen) Aussagen. Auf jeden Fall stellen sie keine nicht-trivialen nomischen Hypothesen im Sinne des hier betrachteten starken nomisch-formallogischen Theorieverständnisses dar.[120]

Einen interessanteren NTNH-Kandidaten stellt die Formel 1.7 dar. Sie wird im Folgenden im Hinblick auf den zeitlich und räumlich ubiquitären Geltungsanspruch realwissenschaftlicher Theorien um Zeitpunkte t und Raumpunkte r[121] der Distanzmessung für die Transportstrecke s zwischen einem Beschaffungs- oder Absatzort O_n und dem gesuchten optimalen Standort O_S erweitert, weil solche Distanzmessungen in „der“ Realität durchaus zeit- und raumabhängig sein können:[122]

$$\forall x_S \forall y_S \forall x_n \forall y_n \forall t \forall r \; : \; s\big((x_S, y_S), (x_n, y_n), t, r\big) = \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2} \tag{1.13}$$

In diesem Fall handelt es sich erstmals um eine nicht-triviale nomische Hypothese (Distanzmessungshypothese), weil sich in „der“ Realität überprüfen lässt, ob die Transportstrecke s zwischen einem Beschaffungs- oder Absatzort O_n und dem gesuchten optimalen Standort O_S tatsächlich mit der Distanz in einem zweidimensionalen kartesischen Koordinatensystem übereinstimmt. Daher wird das starke nomisch-formalsprachliche Theorieverständnis bereits an dieser Stelle erfüllt.

Allerdings erweist sich diese Distanzmessungshypothese aus betriebswirtschaftlicher Perspektive als weitgehend „uninteressant“. Zwar lässt diese nicht-triviale nomische Hypothese

120 Hinzu kommt, dass die Formel 1.12 zwar Allquantoren aufweist, aber ein weiteres charakteristisches Merkmal von nicht-trivialen nomischen Hypothesen nicht aufweist. Es handelt sich um den Sachverhalt, dass sich nicht-triviale nomische Hypothesen im Rahmen einer prädikatenlogischen Rekonstruktion immer als (allquantifizierte) Subjugate darstellen lassen. Diese charakteristische Subjugatform fehlt der o. a. Formel 1.12 als Randbedingung der betroffenen Theorie.

121 Raumpunkte spielen – im Gegensatz zu Zeitpunkten (wie später vor allem im Zusammenhang mit Lernprozessen erläutert wird) – in Theorien der Wirtschaftsinformatik und der Betriebswirtschaftslehre in der Regel keine wesentliche Rolle. Dennoch werden sie hier „der Vollständigkeit halber“ als Komplement zu den interessanteren Zeitpunkten stets angeführt. In wirtschaftswissenschaftlichen Diskursen lassen sich Raumpunkte „in abstrakter Weise“ vor allem als „Branchen“ oder als regionale oder nationale „Märkte“ interpretieren, für die branchen- bzw. marktspezifische Besonderheiten gelten.

122 Für die denkmögliche Zeit- oder Raumabhängigkeit von Distanzmessungen in „der“ Realität muss jedoch „weit ausgeholt“ werden, sodass über die betriebswirtschaftliche Relevanz der nachfolgend angeführten, exemplarischen Argumente trefflich gestritten werden kann. Für die Zeitabhängigkeit von Distanzmessungen lässt sich vor allem auf die physikalisch-kosmologische Erkenntnis einer – sogar beschleunigten – Expansion des Universums verweisen, die dazu führt, dass Distanzmessungen zwischen zwei Objekten unterschiedlich ausfallen können je nachdem, zu welchem Zeitpunkt (auf einer „kosmologischen“ Zeitskala) sie erfolgen. Die Raumabhängigkeit lässt sich in analoger, „weit hergeholter“ Weise auf die Entfernungsmessung in Gegenwart unterschiedlicher Massen zurückführen, die gemäß der Allgemeinen Relativitätstheorie von Einstein zu unterschiedlichen „Verzerrungen“ der Raumzeit führen können. Aber es muss nicht zu derart extremen Gedankenexperimenten gegriffen werden. Stattdessen lässt sich die Raumabhängigkeit von Distanzmessungen beispielsweise auf die relevanten „Transporträume“ beziehen. Wenn sich diese Transporträume auf Lufträume (für Flugzeuge oder auch – wie im o. a. Heliport-Beispiel – für Hubschrauber) oder auf Meeresoberflächen (für Frachtschiffe) erstrecken, kann z. B. in einer „guten Annäherung“ auf zweidimensionale euklidische Distanzmessungen zurückgegriffen werden, solange sich von der Erd(kugel)krümmung in der dritten (Raum-)Dimension im Rahmen der messtechnischen Toleranzen (Näheres dazu später) absehen lässt.

anhand realer Distanzmessungen mannigfach widerlegen, z. B. mithilfe eines Geografischen Informationssystems oder mittels GPS-Messungen im Rahmen einer „üblichen“ geografischen Internet-Software, wie etwa „Google Maps“. Daher müsste die hier rekonstruierte Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, als vielfach *empirisch gescheitert* aus allen betriebswirtschaftlichen Lehrbüchern verbannt werden. Aber so „schlicht“ funktionieren Realwissenschaften wie die Betriebswirtschaftslehre nicht.[123] Stattdessen wird von den Abweichungen zwischen den „theoretisch“ veranschlagten Transportstrecken im Sinne der o. a. Formel einerseits und den tatsächlich gemessenen Transportstrecken als „theoretisch uninteressanten“ Details andererseits in der Regel abgesehen, solange die Abweichungen als „unwesentlich“ eingestuft werden.[124] Sollten diese Abweichungen als „wesentlich“ empfunden werden, lässt sich die Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, entsprechend „reparieren“, indem anstelle der euklidischen Distanzmessung auf eine andere nicht-triviale nomische Distanzmessungshypothese zurückgegriffen wird, die sich beispielsweise auf Distanzen bezieht, die von einem Geografischen Informationssystem zur Verfügung gestellt werden. Wegen dieser „einfachen Reparaturmöglichkeit“ werden die mannigfaltigen empirischen Widerlegungen der nicht-trivialen nomischen Distanzmessungshypothese aus der o. a. Formel 1.13 im Allgemeinen als betriebswirtschaftlich „uninteressant“ (!) eingestuft.

Einen weiteren, dieses Mal auch aus betriebswirtschaftlicher Perspektive hoch interessanten NTNH-Kandidaten stellt die Formel 1.5 dar. Sie wird wiederum um Zeit- und Raumpunkte für den ubiquitären Geltungsansprach einer Theorie erweitert, während von den Definitionsbereichen für die variablen Terme weiterhin der Übersichtlichkeit halber abgesehen wird:

$$\forall x_S \, \forall y_S \, \forall x_n \, \forall y_n \, \forall t \, \forall r \; : \; OPT(x_S, y_S, t, r) \leftrightarrow \neg \left(\exists \widehat{x_S} \, \exists \widehat{y_S} \; : \; K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r) \right) \tag{1.14}$$

Dieser NTNH-Kandidat bedarf jedoch einer „verhaltenswissenschaftlichen“ Erweiterung und Präzisierung, weil sich die Formel 1.14 inhaltlich auf eine rationale Verhaltensweise eines beliebigen Entscheidungsträgers *et* bei der Auswahl eines optimalen Standorts erstreckt. Dieser Inhalt bedarf einer formalsprachlichen Explikation. Hierfür wird zunächst auf das Rationalitätsprädikat $RAT(et, t, r)$ zurückgegriffen. Es drückt die Rationalität[125] des Entscheidungsträgers *et* bei seiner Standortauswahl in einem Zeitpunkt *t* und einem Raumpunkt *r* aus.[126] Hinzu kommt die prädikatenlogische Formulierung $ZS(et, t, r)$ dafür, dass der

123 Polemisch überspitzt könnte die Ansicht vertreten werden, dass Realwissenschaften wie die Betriebswirtschaftslehre gar nicht auf ihre Widerlegung abzielen (in bemerkenswertem Kontrast zu vielfachen „Verbeugungen“ gegenüber einer falsifikationistischen Erkenntnisposition im Anschluss an Popper, der vor allem auch in betriebswirtschaftlichen Diskursen oftmals als erkenntnistheoretischer „Ankerpunkt“ ins „Schaufenster“ erkenntnistheoretischer „Präliminarien“ gestellt wird). Vielmehr geht es oftmals um den Nachweis der „Bewährung“ (siehe dazu das eingangs skizzierte schwache Theorieverständnis) – oder überspitzt formuliert die „Verifizierung“ – von Theorien, an denen im wissenschaftlichen Diskurs so lange wie „möglich“ festgehalten wird.

124 Dies lässt sich unter das Abstrahierungsmerkmal des schwachen Theorieverständnisses subsumieren.

125 Über die Definition der Rationalität eines Subjekts, wie z. B. eines Entscheidungsträgers, lässt sich ausführlich diskutieren. Hier wird ein Rationalitätsverständnis zugrunde gelegt, das sich an Gethmann orientiert. Gethmann (2004c), S. 468, fasst Rationalität auf als die „Bezeichnung für die Fähigkeit, Verfahren diskursiver Einlösung von Geltungsansprüchen ... zu entwickeln, ihnen zu folgen und über sie zu verfügen“. Vgl. auch Frank (2006b), S. 18.

126 Die Relativierung der Formel 1.14 auf Zeitpunkte spielt spätestens an dieser Stelle eine betriebswirtschaftlich bedeutsame Rolle, weil die einschlägigen standortabhängigen Kosten zeitlich variieren können, wie z. B. im

Entscheidungsträger *et* bei seiner Standortauswahl in einem Zeitpunkt *t* und einem Raumpunkt *r* in seinem Zielsystem *ZS* ausschließlich die Transportkosten (K_T) berücksichtigt (degenerierte Artenpräferenz hinsichtlich der Relevanz ausschließlich dieser einen Zielart), diese Transportkosten minimieren (*min*) möchte (Höhenpräferenz), eine deterministische (*det*) Entscheidungssituation ohne jeden Aspekt der Unsicherheit unterstellt (Risikopräferenz), aufgrund eines „statischen“ Modells nicht zwischen unterschiedlichen Zeitpunkten des Kostenanfalls unterscheidet (*stat*), wie es z. B. in einem investitionstheoretischen Modell für eine Standortauswahl zu erwarten wäre (Zeitpräferenz), und ausschließlich auf seine eigenen (Transport-)Kosten achtet (*nsoz*), also von allen Auswirkungen auf Dritte[127] absieht (Irrelevanz von Sozialpräferenzen).

Hiermit ist die verhaltenswissenschaftlich motivierte Rekonstruktionsarbeit noch nicht abgeschlossen. Denn das Optimalitätsprädikat $OPT(x_S, y_S, t, r)$ drückt nur einen „optimierungstechnischen Sachverhalt“, aber keine Verhaltensweise eines Entscheidungsträgers aus. Daher wird es durch das Prädikat $AUS_{opt}\,(et, t, r, (x_S, y_S))$ ersetzt, das die Auswahlhandlung eines Entscheidungsträgers *et* hinsichtlich eines Standorts mit den Koordinaten (x_S, y_S) in einem Zeitpunkt *t* und einem Raumpunkt *r* als problembezogene Verhaltensweise ausdrückt. Nach erforderlicher Anpassung des Bijugats, das sich in den Formeln 1.5 und 1.14 auf das Optimalitätsprädikat $OPT(x_S, y_S, [t, r])$ bezog, in ein „schwächeres“ Subjugat[128] resultiert als entsprechend „verfeinerter“ NTNH-Kandidat folgende Formel:

$$\begin{aligned}\forall x_S\,\forall y_S\,\forall t\,\forall r\,\forall et\ :\ \big(RAT(et,t,r) \wedge ZS(et,t,r) = (K_T, min, det, stat, nsoz)\big) \rightarrow \ldots\\ \Big(AUS_{opt}(et,t,r,x_S,y_S) \rightarrow \neg\big(\exists\widehat{x_S}\,\exists\widehat{y_S}\ :\ K_T(\widehat{x_S},\widehat{y_S},t,r) < K_T(x_S,y_S,t,r)\big)\Big)\end{aligned} \tag{1.15}$$

Diese prädikatenlogische Formel 1.15 stellt nicht nur eine aus betriebswirtschaftlicher Perspektive (im Gegensatz zur o. a. Distanzmessungshypothese) interessante nicht-triviale

Hinblick auf inflationär oder auch konjunkturell bedingte Kostensteigerungen. Die Raumpunkte treten hingegen weiterhin in den Hintergrund, weil die Raumkomponente – wie z. B. eine Standortauswahl in Deutschland, China oder Iran – bereits durch die Standortkoordinaten in der Formel 1.14 berücksichtigt wird.

127 Als solche Auswirkungen auf Dritte könnten bei einer Standortwahl beispielsweise die „externen Kosten“ berücksichtigt werden, die für das kommunale Umfeld eines Standorts anfallen. Dafür kommen beispielsweise bei einer Großinvestition – wie es kürzlich bei der Auswahl eines Standorts für einen zweiten zentralen Verwaltungsstandort („Headquarter“) von Amazon der Fall war – in Betracht: die Erhöhung des lokalen Mietpreisniveaus aufgrund des Zuzugs von zahlreichen Mitarbeitern (im Falle von Amazon immerhin ca. 50.000 Personen erwartet), die „Verstopfung“ des kommunalen Öffentlichen Personennahverkehrs und die „üblichen“ Subventionen für die standortnahe Verbesserung der öffentlichen Infrastruktur, die bei einem als konstant unterstellten kommunalen Ausgabenbudget für andere öffentliche Aufgaben, wie z. B. Obdachlosenhilfe, Kindertagesstätten und Bildungseinrichtungen, fehlen.

128 Diese Anpassung ist erforderlich, weil das Optimalitätsprädikat $OPT(x_S, y_S, t, r)$ einen „theoretischen“ Begriff darstellt, der unabhängig vom realen Verhalten eines Entscheidungsträgers dadurch definiert ist, dass er *genau dann* erfüllt ist, *wenn* die Optimierungsbedingung $\exists\widehat{x_S}\,\exists\widehat{y_S}\ :\ K_T(\widehat{x_S},\widehat{y_S},t,r) < K_T(x_S,y_S,t,r)$ zutrifft. Das nachfolgende Prädikat $AUS_{opt}\,(et,t,r,(x_S,y_S))$ für eine Auswahlhandlung eines Entscheidungsträgers *et* besitzt jedoch eine andere Qualität. Dies liegt daran, dass einerseits die Optimierungsbedingung „zufällig“ von *mehreren* Standorten erfüllt sein kann, jedoch andererseits die Auswahlhandlung eines Entscheidungsträgers immer nur höchstens (in der Regel: genau) *einen* Standort umfasst. Im Falle eines inkonsistent spezifizierten Standortauswahlproblems kann auch der Fall eintreten, dass sich überhaupt kein „modellerfüllender“ Standort auswählen lässt (hierauf weist die „verallgemeinerte“ Formulierung „höchstens“ hin). Während dieser verhaltensbezogenen Reduzierung des „Auswahlraums“ auf einen Standort kann es zwar mehrere im Sinne der Formeln 1.5 und 1.14 optimale Standorte geben (Formulierung als Bijugat: Ein Standort ist genau dann optimal, wenn er die o. a. Optimierungsbedingung erfüllt). Aber ein Entscheidungsträger wählt immer nur (höchstens oder genau) einen Standort aus. Darauf weist die Formulierung als Subjugat hin: *Wenn* ein Standort als optimal ausgewählt wird, *dann* muss er die o. a. Optimierungsbedingung erfüllen. Oder mit anderen Worten: Ein Standort wird nur dann ausgewählt, wenn es sich um einen aus den möglicherweise mehreren optimalen Standorten handelt.

nomische Hypothese dar, sondern birgt bei näherem Hinsehen auch gravierende theoretische Probleme. Daher wird sie im Folgenden etwas ausführlicher erörtert.

Zunächst wird die nicht-triviale nomische Hypothese der Formel 1.15 in natürlichsprachlicher Weise paraphrasiert,[129] um ihren Inhalt zu verdeutlichen: *Wenn* der *situative Kontext* vorliegt, dass der Entscheidungsträger *et* im Zeitpunkt *t* und im Raumpunkt *r* rational handelt (ausgedrückt durch das Prädikat $RAT(et, t, r)$), und wenn das *Zielsystem* $ZS(et, t, r)$ des Entscheidungsträgers *et* im Zeitpunkt *t* und im Raumpunkt *r* so beschaffen ist, dass er nur die Transportkosten (K_T) minimieren (*min*) möchte, an eine deterministische Umwelt (*det*) glaubt, sich statisch (*stat*) verhält und keine sozialen Aspekte berücksichtigt (*nsoz*), dann besteht die *optimale Auswahlhandlung* des Entscheidungsträgers daraus, einen transportkostenminimalen Standort mit den Koordinaten (x_S, y_S) auszuwählen (ausgedrückt durch das Prädikat $AUS_{opt}(et, t, r, x_S, y_S)$ mit der anschließenden Optimalitätsbedingung $\neg\big(\exists\widehat{x_S}\,\exists\widehat{y_S} : K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r)\big)$.

Es handelt sich um eine nicht-triviale nomische Hypothese (q.e.d.), weil keine begriffs- oder deduktiv-analytischen Wahrheiten „a priori" (vor jeder Erfahrung) ausgesprochen werden, sondern eine Behauptung aufgestellt wird, wie sich ein Entscheidungsträger hinsichtlich der Auswahl eines Standorts unter den Voraussetzungen von Rationalität und eines konkret spezifizierten Zielsystems „jederzeit" und „überall" (also ubiquitär) verhalten wird. Diese nomische Verhaltenshypothese ist empirisch gehaltvoll, weil sie sich durch die Beobachtung des tatsächlichen Auswahlverhaltens von Entscheidungsträgern bei Standortauswahlproblemen – zumindest prima facie (darauf wird hinsichtlich des Rationalitätsprädikats kritisch zurückgekommen) – empirisch überprüfen und somit auch widerlegen („falsifizieren") lässt.

Außerdem weist die Formel 1.15 die charakteristische Form eines Subjugats für eine nomische Hypothese auf. In der *Wenn-Komponente*, dem Antezedens $RAT(et, t, r) \wedge ZS(et, t, r) = (K_T, min, det, stat, nsoz)$, werden die *Anwendungsbedingungen* der nomischen Hypothese (und auch zum Teil der zugehörigen Theorie) expliziert. Sie erstrecken sich auf den situativen Kontext der unterstellten *Rationalität* der Auswahlhandlung des Entscheidungsträgers und auf das angenommene *Zielsystem* des Entscheidungsträgers. In der *Dann-Komponente*, der Konklusion $AUS_{opt}(et, t, r, x_S, y_S) \rightarrow \neg\big(\exists\widehat{x_S}\,\exists\widehat{y_S} : K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r)\big)$, wird der „nomische Gehalt im engeren Sinn" ausgedrückt, der festlegt, wie sich ein Entscheidungsträger unter den vorausgesetzten Anwendungsbedingungen hinsichtlich der *Auswahl eines optimalen Standorts* O_S mit den Koordinaten (x_S, y_S) *verhalten* wird. Die nicht-triviale nomische Hypothese der Formel 1.15, die sich auch als „nomischer Kern" der Theorie auffassen lässt, die dem klassischen Transportkostenmodell zugrunde liegt, weist aber auch einige „Abgründe" auf.

Zunächst fällt auf, dass die nicht-triviale nomische Hypothese der Formel 1.15 in ihrer Wenn-Komponente (Antezedens) zwei Anwendungsbedingungen für den Geltungsanspruch dieser Hypothese umfasst. Erstens wird mithilfe des Rationalitätsprädikats $RAT(et, t, r)$ vorausgesetzt, dass sich der Entscheidungsträger *et* im Zeitpunkt *t* und im Raumpunkt *r* rational verhält. Zweitens wird ein Zielsystem *ZS* für den Entscheidungsträger *et* im Zeitpunkt *t* und im Raumpunkt *r* vorausgesetzt, das sich als sehr „schlicht" erweist. Es umfasst die

129 An dieser Stelle zeigt sich in exemplarischer Weise, dass das „starke" nomisch-formalsprachliche Theorieverständnis auch das „schwache" Theorieverständnis umfassen kann (wenn zunächst von weiteren konstituierenden Merkmalen des „schwachen" Theorieverständnisses, wie z. B. Allgemeinheit und Abstrahierung abgesehen wird, die sich jedoch ebenso formalsprachlich erfassen lassen, wie etwa durch Allquantoren bzw. durch Präzisionspräferenzen, auf die noch zurückgekommen wird), das auf eine formalsprachliche Theoriespezifizierung zugunsten einer „intuitiv" anmutenden natürlichsprachlichen Formulierungsweise verzichtet.

Transportkosten als einzigen entscheidungsrelevanten Zielinhalt, betrachtet ausschließlich eine „konventionelle“ Kostenminimierung, abstrahiert von allen Unsicherheiten sowie dynamischen (mehrperiodigen) Einflussmöglichkeiten auf das reale Standortauswahlproblem und lässt Rücksichten auf die Betroffenheit weiterer Akteure im Zusammenhang mit Sozialpräferenzen vollkommen außer Acht.

Eine solche weitreichende Beschränkung des klassischen Transportkostenmodells und seiner zugrunde liegenden („rekonstruierten“) Theorie auf rational handelnde Entscheidungsträger und ein „schlichtes“ Zielsystem ist keineswegs grundsätzlich abzulehnen.[130] Stattdessen stellt eine solche Beschränkung ein Merkmal der meisten betriebswirtschaftlichen (Basis-)Modelle dar. Sie führt dazu, dass die zugrunde liegende Theorie dem Anspruch der Allgemeinheit, der im Rahmen des „schwachen“ Theorieverständnisses oftmals ohne tiefere Reflexion erhoben wird, nur ansatzweise gerecht wird. Daher wird im Fall der meisten betriebswirtschaftlichen Modelle, zu denen auch das hier vorgestellte klassische Transportkostenmodell gehört, zuweilen von „Miniaturtheorien“[131] gesprochen. Diese semantisch zutreffende Bezeichnung hebt darauf ab, dass mittels der Anwendungsbedingungen der (nicht-trivialen) nomischen Hypothesen einer Theorie der Geltungsanspruch dieser Theorie auf einen sehr kleinen Realitätsausschnitt eingeengt wird.

Dies ist nicht zu bestreiten. Aber es ist ebenso hervorzuheben, dass mittels dieser Einengung der ubiquitäre Geltungsanspruch einer nicht-trivialen nomischen Hypothese keineswegs aufgegeben wird. Stattdessen besitzt die nicht-triviale nomische Hypothese einer „Miniaturtheorie“ weiterhin einen ubiquitären Geltungsanspruch, der sich in den Allquantifizierungen ihrer Zeit- und Raumpunkte t bzw. r manifestiert. Dieser ubiquitäre

130 An dieser Stelle ist einem möglichen Missverständnis vorzubeugen. Die Beschränkung des klassischen Transportkostenmodells und seiner zugrunde liegenden („rekonstruierten“) Theorie auf rational handelnde Entscheidungsträger und ein „schlichtes“ Zielsystem stellt keine deskriptive Aussage darüber dar, wie sich Entscheidungsträger in der betrieblichen Realität tatsächlich verhalten (Aussagen dieser Art bleiben der deskriptiven Entscheidungstheorie oder dem Forschungsgebiet der „behavioral economics“ vorbehalten). Insbesondere wird nicht behauptet, dass sich Entscheidungsträger in der betrieblichen Realität „immer“ rational verhalten und „immer“ auf ein Zielsystem ZS mit den Transportkosten als einzig relevantem Zielinhalt beschränken. Eine solche Behauptung würde die Allquantoren zu Beginn der Formel 1.15 gründlich missverstehen. Stattdessen wird mittels der beiden Anwendungsbedingungen im Antezedens der Formel 1.15, d. h. in ihrer ersten Zeile, lediglich eine *konditionale* Aussage aufgestellt: *Falls* sich ein Entscheidungsträger rational verhält und sich auf ein Zielsystem ZS mit den Transportkosten als einzig relevantem Zielinhalt beschränkt, dann wird er hinsichtlich des gesuchten optimalen Standorts eine Auswahlhandlung realisieren, die der Dann-Komponente (Konklusion) in der zweiten Zeile der Formel 1.15 entspricht. *Ob* der in den beiden Anwendungsbedingungen spezifizierte „Fall“ in der betrieblichen Realität überhaupt eintritt und – wenn ja – *wie häufig* dies der Fall ist, darüber sagt die nicht-triviale nomische Hypothese der Formel 1.15 *nichts* aus. Daher gehören das hier vorgestellte klassische Transportkostenmodell und die ihm zugrunde liegende Theorie zum Bereich der *normativen* Entscheidungstheorie. Dieser Zweig der Entscheidungstheorie „beschreibt“ nicht, wie sich Entscheidungsträger in der betrieblichen Realität tatsächlich verhalten, sondern postuliert in „normativer“ (wertender) Weise, wie sich ein Entscheidungsträger in einer *idealisierten* Situation verhalten *sollte*, die einige (idealisierende) Grundannahmen der normativen Entscheidungstheorie erfüllt. Zu diesen Grundannahmen zählt im Allgemeinen die Prämisse der Rationalität des Entscheidungsträgers. Hinzu kommt im hier betrachteten Fall die spezielle Grundannahme, dass der Entscheidungsträger ein Zielsystem ZS mit den Transportkosten als einzig relevantem Zielinhalt verfolgt.

131 Die Bezeichnung „Miniaturtheorien“ ist nicht etabliert. Stattdessen wird häufiger von „Theorien mittlerer Reichweite“ oder „middle-range theories“ gesprochen; vgl. Lee (2021), S. 515 ff.; Boudon (1991), S. 519 ff.; Merton (1968), S. 39 ff. Wegen der weitreichenden Unbestimmtheit dessen, was unter einer „mittleren“ Reichweite zu verstehen ist, bevorzugen die Verfasser die „treffendere“ Bezeichnung einer „Miniaturtheorie“. Nur am Rande wird erwähnt, dass die hier angesprochenen „Theorien mittlerer Reichweite“ mit den „Quasi-Theorien“, die mit Verweis auf Albert (1957), S. 67 ff. u. 75 f., bereits an früherer Stelle kurz vorgestellt wurden, konzeptionell eng verwandt sind.

Geltungsanspruch wird mittels der Anwendungsbedingungen der nicht-trivialen nomischen Hypothese jedoch auf diejenigen sehr kleinen oder – salopp gesprochen „minimalistischen" – Realitätsausschnitte fokussiert, in denen diese Anwendungsbedingungen erfüllt werden. Dies bedeutet konkret: Sofern die Anwendungsbedingungen erfüllt sind, lässt sich die nicht-triviale nomische Hypothese einer Theorie hinsichtlich ihres *Geltungsanspruchs* überprüfen und in diesem Rahmen auch *empirisch widerlegen.* Außerhalb des Realitätsausschnitts, der durch diese Anwendungsbedingungen präzise eingegrenzt wird, kann die nicht-triviale nomische Hypothese einer Theorie wegen ihrer *Nichtanwendbarkeit* grundsätzlich nicht empirisch widerlegt werden. In diesem „Außenbereich" lässt sich lediglich die Nichtanwendbarkeit der nicht-trivialen nomischen Hypothese einer Theorie – und somit der betroffenen Theorie insgesamt – feststellen. Leider wird in zahlreichen Diskursen, die sich mit betriebswirtschaftlichen Theorien auseinandersetzten, diese *fundamentale* kategorische Differenzierung zwischen Geltungsanspruch und Nichtanwendbarkeit einer Theorie im Hinblick auf ihre nicht-trivialen nomischen Hypothesen übersehen. Daher sind „wohlfeile" Vorhaltungen, eine Theorie sei angesichts empirischer Fakten[132] „gescheitert", oftmals deplatziert, weil sie die Anwendungsbedingungen der jeweils betroffenen Theorie – fahrlässig oder sogar vorsätzlich – ausblenden. Dies mag auch daran liegen, dass die Anwendungsbedingungen einer Theorie, die sich zumeist in den Wenn-Komponenten ihrer nicht-trivialen nomischen Hypothesen manifestieren, oftmals nicht mit der gebührenden Sorgfalt gewürdigt werden.

Darüber hinaus erweist sich das Rationalitätsprädikat $RAT(et, t, r)$ als „vertrackt". Auf den ersten Blick mag es „unscheinbar daherkommen". Es stellt aber aus der Perspektive des „statement view" einen wesentlichen *Schwachpunkt* der hier vorgestellten Theorierekonstruktion dar. Denn es handelt sich um ein *Imputationsprädikat.* Es kann dem Verhalten eines Entscheidungsträgers *et* seitens eines Dritten (z. B. eines Beobachters) lediglich *zugeschrieben* werden. Seine Erfüllung lässt sich aber *nicht* durch eine (subjekt-)unabhängige Beobachtung *empirisch* zweifelsfrei („objektiv") *überprüfen* oder gar feststellen. Aufgrund dieses grundsätzlichen Überprüfungsdefekts kann streng genommen nicht empirisch entschieden werden, ob die Anwendungsbedingung der nicht-trivialen nomischen Hypothese der Formel 1.15 erfüllt ist – oder nicht. Dies stellt einen gravierenden *Defekt* der hier vorgeschlagenen Theorierekonstruktion dar.

Die grundsätzliche Problematik des Imputationscharakters des Rationalitätsprädikats $RAT(et, t, r)$ wird im Folgenden kurz, aber bei Weitem nicht erschöpfend erläutert. Die Rationalität eines Entscheidungsträgers *et* in einem Zeitpunkt *t* und in einem Raumpunkt *r* lässt sich leider nicht beobachten, sondern nur unterstellen („zuschreiben"). Denn es existieren keine empirisch überprüfbaren Kriterien, anhand derer sich die Rationalität eines Entscheidungsträgers „direkt" überprüfen lässt. Stattdessen wird auf die Rationalität eines Entscheidungsträgers in der Regel indirekt geschlossen, wenn er sich so verhält, wie es aus der Perspektive von näher zu spezifizierenden Rationalitätsannahmen zu erwarten

132 Es wird hier nicht näher diskutiert, ob sich aus erkenntnistheoretischer Perspektive tatsächlich unzweifelhafte empirische Fakten feststellen lassen, wie sie z. B. angesichts des aktuellen Mainstreams einer „faktenbasierten", „datengetriebenen" oder „evidenzbasierten" Forschung oftmals unkritisch präsupponiert werden. Zahlreiche sowohl empirische (!) als auch konzeptionelle Forschungsansätze lassen erhebliche Zweifel an einer solchen „naiven" Fakten-, Daten- oder Evidenzpräsupposition aufkommen. Sie lassen sich in der hier gebotenen Kürze nicht im Detail diskutieren. Lediglich in exemplarischer Weise wird auf einige wenige Schlagworte des einschlägigen wissenschaftlichen Diskurses verwiesen, wie etwa die „soziale Konstruktion" von Beobachtungs- oder Befragungsergebnissen, das Phänomen der „als sozial erwünscht" empfundenen Antworten im Rahmen empirischer Befragungen sowie die erkenntnistheoretische Position eines gemäßigten oder sogar radikalen Konstruktivismus.

ist. Diese Rationalitätsannahmen betreffen hier eine optimale Standortauswahl, die das Zielsystem ZS „bestmöglich“ erfüllt. Aber diese Rationalitätsvorstellung erweist sich als „zirkulär“. Denn ein Entscheidungsträger wird zunächst als rational handelnd im Sinne einer optimalen Alternativenauswahl vorausgesetzt und die Rationalität seines Handelns wird nachträglich im Hinblick darauf beurteilt, ob er eine im Sinne seines Zielsystem ZS optimale Alternative ausgewählt hat. Die optimale Alternativenauswahl wird also sowohl als Inbegriff der Rationalität vorausgesetzt als auch hinsichtlich der „gesetzesartigen“ Prognose des Verhaltens eines Entscheidungsträgers formuliert. Folglich wird der Begriff der optimalen Alternativenauswahl sowohl im Antezedens der Formel 1.15 mittels des Rationalitätsprädikats vorausgesetzt als auch in ihrer Konklusion mittels der Optimalitätsbedingung $\neg\left(\exists \widehat{x_S}\, \exists \widehat{y_S} : K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r)\right)$ angenommen. Aufgrund dieser „zirkulären“ Beziehung zwischen Antezedenz und Konklusion der Formel 1.15 gilt: Die nomische Hypothese, die mittels der Formel 1.15 ausgedrückt wird, erweist sich zwar im rein formallogischen Sinne als nicht-trivial, aber dennoch als nicht empirisch überprüfbar. Denn die nomische Hypothese der Formel 1.15 stellt sich als „widerlegungsresistent“ oder (synonym) als „immun“ gegenüber beliebigen empirischen Widerlegungsversuchen voraus.[133] Diese „*theoretische (Selbst-)Immunisierung*“[134] kann leicht verdeutlicht werden: Falls ein Entscheidungsträger in der betrieblichen Realität einen Standort auswählt, der die in Formel 1.15 enthaltene Optimalitätsbedingung $\neg\left(\exists \widehat{x_S}\, \exists \widehat{y_S} : K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r)\right)$ nicht erfüllt, so lässt sich darauf verweisen, dass der Entscheidungsträger „eben nicht rational“ gehandelt habe, dass also das Rationalitätsprädikat in der ersten Anwendungsbedingung der (nicht-trivialen) nomischen Hypothese der Formel 1.15 nicht erfüllt sei. Auf diese Weise lassen sich beliebige Handlungsweisen von Entscheidungsträgern in der betrieblichen Realität „wegerklären“, indem auf ihren „offensichtlichen“ Verstoß gegen das Rationalitätsprädikat verwiesen wird.

Diese „Rationalitätsfalle“ lässt sich für das konventionelle Theorienkonzept des hier vorgestellten „statement view“ des Öfteren feststellen, sofern sich die betroffenen Theorien auf das Verhalten rationaler Entscheidungsträger beziehen. Diese schwerwiegenden Probleme eines „zirkulären“ Bezugs zentraler Begriffe einer Theorie auf sich selbst lassen sich im konkurrierenden Theorienkonzept des „non statement view“ im Kontext der analytischen Wissenschaftstheorie (strukturalistisches Theorienkonzept)[135] präzise untersuchen und auch auflösen. Dies betrifft das Problem der sogenannten T-theoretischen Begriffe, die in einer Theorie T von Anfang an als gültig vorausgesetzt werden (sich also nicht zur empirischen Widerlegung einer Theorie eignen). Das Problem dieser T-theoretischen Begriffe kann zwar im strukturalistischen Theorienkonzept formalsprachlich zufriedenstellend gelöst werden, wie z. B. mittels der sogenannten Ramsey-Eliminierung.[136] Aber das strukturalistische

133 Dies stellt aus der Perspektive des „etablierten“ falsifikationistischen Wissenschaftsverständnisses eine „kleine Katastrophe“ dar.

134 Vgl. zur (Selbst-)Immunisierung von Theorien gegenüber empirischen Widerlegungsversuchen Popper (2005), S. 58 (mit Verweis auf Albert); Albert (1991), passim, wie z. B. S. 36, 41, 58, 64 115 ff. u. 126 ff.; Eichhorn (1979), S. 100.

135 Vgl. zu Überblicken über das Theorienkonzept des „non statement view“ (strukturalistisches Theorienkonzept) beispielsweise Kornmesser und Büttemeyer (2020), S. 144 ff.; Balzer und Brendel (2019), S. 17, 19 ff. u. 155 ff.; Schurz (2011), S. 184 ff. (ansatzweise, teilweise distanziert auf S. 188) u. 213 f.; Balzer (2009), S. 48 ff. u. 78 ff.; Moulines (2008), S. 159 ff.; Zelewski (2007), S. 452 ff.; Teichert und Rott (2004), S. 111 f.; Zelewski (2004), S. 10 ff.; Zelewski (1993), S. 94 ff., 132 ff., 151 ff. u. 225 ff.; Stegmüller (1987), S. 468 ff.; Sneed (1979), S. 165 ff. u. 259 ff., insbesondere S. 171 u. 183 f.

136 Vgl. Schurz (2011), S. 213 f.; Moulines (2008), S. 79 ff., insbesondere S. 81 f.; Rott (2004), S. 460 f.; Zelewski (1993), S. 118 ff.

Theorienkonzept erweist sich vor allem in formalsprachlicher Hinsicht als derart komplex, dass es in diesem Überblicksbeitrag nicht näher behandelt wird.

Um der vorgenannten „Rationalitätsfalle“ zu entkommen, bieten sich mehrere Vermeidungsstrategien an. Erstens könnte daran gedacht werden, auf das Rationalitätsprädikat $RAT(et, t, r)$ im Antezedenz der Formel 1.15 vollständig zu verzichten. Dies wäre aber keine überzeugende Antwort auf die „Rationalitätsfalle“. Denn bei einem Verzicht auf das Rationalitätsprädikat $RAT(et, t, r)$ im Antezedenz der Formel 1.15 ließe sich die nicht-triviale nomische Hypothese dieser Formel in mannigfaltigen, empirischen beobachtbaren Situationen widerlegen (was von den Vertretern des klassischen Transportkostenmodells und seiner theoretischen Fundierung vermutlich nicht intendiert wird). Denn in zahlreichen Situationen wird ein Entscheidungsträger nicht („vollkommen“) rational handeln, sondern seine Auswahlhandlungen auch von Einflüssen leiten lassen, die im Sinne der betriebswirtschaftlichen Entscheidungstheorie als nicht-rational gelten, aber vielleicht dennoch als „vernünftig“ klassifiziert werden. Dies betrifft beispielsweise Ad-hoc-Entscheidungen aufgrund mangelnder zeitlicher und sonstiger Ressourcen für den Entscheidungsprozess sowie „habitualisierte“ Entscheidungen, die nicht aufgrund von analytischen Optimierungsüberlegungen in Bezug auf das jeweils aktuelle Standortauswahlproblem, sondern in Anlehnung an frühere, als „erfolgreich“ empfundene und „generalisierte“ Entscheidungen getroffen werden. Das Spektrum der vorstellbaren Eigenschaften von zwar „plausiblen“ oder „realistisch“ anmutenden Entscheidungssituationen lässt sich nahezu beliebig erweitern. Es mag an dieser Stelle ausreichen, um zu belegen, dass die Präsupposition des Rationalitätsprädikats $RAT(et, t, r)$, es würde jeweils eine „optimale“ Standortalternative ausgewählt, in empirisch beobachtbaren Realproblemen einer Standortauswahl keineswegs erfüllt sein muss.

Aus den vorgenannten Gründen wäre es wissenschaftlich erstrebenswert, das hoch problematische Rationalitätsprädikat $RAT(et, t, r)$ im Antezedenz der Formel 1.15 durch andere Anwendungsbedingungen zu ersetzen, welche die Rationalität der Standortauswahl nicht in zirkulärer Weise voraussetzen, sondern diese Rationalität auf Rationalitätsanforderungen zurückführen, die unabhängig von der zirkulären Auswahl eines – im Sinne des Zielsystems ZS – transportkostenminimalen Standorts sind. Es sollten zumindest notwendige Rationalitätsanforderungen aufgestellt werden. Darüber hinaus wären hinreichende Rationalitätsanforderungen wünschenswert, liegen aber nach Einschätzung der Verfasser dieses Beitrags weit außerhalb des aktuellen betriebswirtschaftlichen Diskurses. Lediglich in exemplarischer Weise wird auf einige notwendige Bedingungen für die Rationalität der Standortauswahl hingewiesen: Der Entscheidungsträger sollte über „ausreichend“ Zeit und (Computer-)Ressourcen verfügen, um die Berechnungen für eine Standortauswahl durchzuführen (Kapazitätsprämisse). Außerdem sollten „ausreichende“ Ressourcen bereitstehen, damit der Entscheidungsträger (oder seine Mitarbeiter) die für das Modell der „optimalen“ Standortauswahl erforderlichen Informationen ermitteln kann (Informationsprämisse). Die Formulierungen „ausreichend“ lassen erkennen, dass es sich um keine präzise formulierten, sondern allenfalls um „tentative“ Anforderungen handelt. Diese Anforderungen müssten für eine betriebswirtschaftlich „präzise“ Theorie konkretisiert werden. Sie stellen im hier vorgelegten Beitrag lediglich Desiderate dar, zu deren Einlösung nur auf zukünftig wünschenswerte Forschungsarbeiten „vertröstet“ werden kann.

Aus den voranstehenden Ausführungen wird deutlich, dass die nicht-triviale nomische Hypothese der Formel 1.15 für die rekonstruierte Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, nicht nur als Ganzes eine herausragende Rolle spielt, weil es sich um die einzige – aus betriebswirtschaftlicher Sicht – „interessante“ nicht-triviale nomische Hypothese handelt. Vielmehr besitzen ihre zwei Anwendungsbedingungen im Antezedens (der ersten Zeile) der Formel 1.15 auch eine große Bedeutung für die Einschränkung des Geltungsanspruchs der rekonstruierten „Miniaturtheorie“ auf einen sehr kleinen Realitätsausschnitt, der durch die Rationalität des Entscheidungsträgers *et* und ein „schlichtes“ Zielsystem *ZS* mit den Transportkosten als einzigem für relevant erachteten Zielinhalt charakterisiert wird.

Um diese Einschränkung des Geltungsanspruchs einer Theorie mittels der Anwendungsbedingungen in ihrer nicht-trivialen nomischen Hypothese noch stärker herauszuarbeiten, kann auf das Konstrukt des *intendierten Anwendungsbereichs* IAB_T einer Theorie *T* zurückgegriffen werden. Dieses Konstrukt stammt zwar aus dem bereits kurz angesprochenen Theorienkonzept des „non statement view“ (oder synonym: strukturalistisches Theorienkonzept).[137] Das Konstrukt des intendierten Anwendungsbereichs lässt sich jedoch ohne Schwierigkeiten auf den „statement view“ übertragen, der diesem Beitrag als Theorienkonzept des „Mainstreams“ betriebswirtschaftlicher Theoriediskussionen zugrunde liegt.

Zunächst könnte erwogen werden, den intendierten Anwendungsbereich IAB_{TKM} der Theorie T_{TKM} für das klassische Transportkostenmodell (*TKM*) durch eine sechste Formel zu ergänzen, die zu den eingangs vorgestellten fünf Formeln 1.5 bis 1.9 – einschließlich der später erfolgten Weiterentwicklungen dieser fünf Formeln – hinzugefügt wird:

$$\forall et \forall t \forall r \, : \, RAT(et,t,r) \wedge ZS(et,t,r) = (K_T, min, det, stat, nsoz) \tag{1.16}$$

Diese Formel ist aber zurückzuweisen. Sie würde ausdrücken, dass alle Entscheidungsträger *et* „immer“ (in jedem Zeitpunkt *t*) und „überall“ (in jedem Raumpunkt *r*) rational handeln und das Zielsystem *ZS* verfolgen. Dies ist jedoch in der betrieblichen Realität nicht der Fall. Daher kann die Formel 1.16 nicht dazu dienen, den intendierten Anwendungsbereich der Theorie für das klassische Transportkostenmodell zu spezifizieren.

Stattdessen lässt sich der intendierte Anwendungsbereich der Theorie für das klassische Transportkostenmodell zunächst mithilfe eines Prädikats $IAB_{TKM}(et,t,r)$ spezifizieren, das sich auf beliebige Entscheidungsträger *et* sowie Zeitpunkte *t* und Raumpunkte *r* erstreckt. Mithilfe eines Subjugats wird ausgedrückt, dass eine (erste) notwendige Bedingung[138] für das Vorliegen dieses intendierten Anwendungsbereichs – also für die Erfüllung des Prädikats $IAB_{TKM}(et,t,r)$ – darin besteht, dass die Entscheidungsträger *et* rational handeln und das Zielsystem *ZS* verfolgen:

$$\begin{aligned} &\forall et \forall t \forall r \, : \, IAB_{TKM}(et,t,r) \rightarrow \\ &\quad RAT(et,t,r) \wedge ZS(et,t,r) = (K_T, min, det, stat, nsoz) \end{aligned} \tag{1.17}$$

137 Vgl. Balzer (2009), S. 48 ff.

138 Für die Spezifizierung des intendierten Anwendungsbereichs ist es wichtig, *notwendige* Bedingungen zu verwenden. Sie ermöglichen es, den intendierten Anwendungsbereich einer Theorie „offen“ zu formulieren, sodass der intendierte Anwendungsbereich durch Ergänzung weiterer notwendiger Bedingungen sukzessiv präzisiert werden kann, ohne ihn jemals vollständig festzulegen. Diese – prima facie eventuell „merkwürdig“ anmutende – Vorgehensweise entspricht der Verwendung „paradigmatischer Beispiele“, um im strukturalistischen Theorienkonzept den intendierten Anwendungsbereich einer Theorie „einzukreisen“, aber nicht vollständig zu definieren (was aus der Perspektive des strukturalistischen Theorienkonzepts in der Regel für nicht leistbar gehalten wird). Würde hingegen der intendierte Anwendungsbereich der Theorie für das klassische Transportkostenmodell mithilfe eines Bijugats in der Formel 1.16 spezifiziert werden, wären die Rationalität des Entscheidungsträgers *et* und sein Zielsystem *ZS* sowohl *notwendig* als auch *hinreichend* für das Vorliegen des intendierten Anwendungsbereichs dieser Theorie. Hierdurch wäre der intendierte Anwendungsbereich bereits „abschließend“ (also vollständig) definiert, sodass keine Möglichkeit bestünde, den intendierten Anwendungsbereich – wie es im Folgenden geschehen wird – weiterführend zu präzisieren.

Mittels einer zweiten notwendigen Bedingung lässt sich der intendierte Anwendungsbereich der Theorie für das klassische Transportkostenmodell auf interessante Weise präzisieren.[139] Hierzu wird das ursprünglich eingeführte Prädikat $IAB_{TKM}(et, t, r)$ um die Koordinaten (x_S, y_S) des gesuchten optimalen Standorts O_S sowie die Koordinaten (x_n, y_n) eines beliebigen Beschaffungs- oder Absatzortes O_n erweitert.[140] Darüber hinaus wird die Funktion s für die Transportstrecke zwischen den Standorten O_S und O_n, die auf der euklidischen Distanz in einem zweidimensionalen kartesischen Koordinatensystem beruhte, als theoretische Transportstrecke s_{theo} präzisiert. Ihr wird die Funktion s_{real} für die reale Transportstrecke gegenübergestellt, die sich beispielsweise mithilfe eines Geografischen Informationssystems ermitteln lässt. Schließlich wird ein Parameter $tol(et, t, r)$ ergänzt, der im Sinne eines Schwellenwerts die Toleranz des Entscheidungsträgers *et* im Zeitpunkt *t* und im Raumpunkt *r* hinsichtlich der Abweichungen zwischen der theoretischen, im klassischen Transportkostenmodell verwendeten Transportstrecke s_{theo} und der realen Transportstrecke s_{real} ausdrückt. Dieser Schwellenwert für die vom Entscheidungsträger maximal tolerierte Abweichung zwischen theoretischer und realer Streckenlänge lässt sich neben den o. a. fünf Präferenzarten für ein „gut strukturiertes“ Entscheidungsmodell als eine neuartige, sechste Präferenzart für die „Präzisionspräferenz“[141] des Entscheidungsträgers auffassen.

Mithilfe dieser Erläuterungen lässt sich die weiterführende Präzisierung des intendierten Anwendungsbereichs der Theorie für das klassische Transportkostenmodell durch folgende Formel festlegen:

$$\begin{aligned} &\forall et \, \forall x_S \, \forall y_S \, \forall x_n \, \forall y_n \, \forall t \, \forall r \; : \; IAB_{TKM}(et, x_S, y_S, x_n, y_n, t, r) \rightarrow \ldots \\ &\quad s_{theo}\big((x_S, y_S), (x_n, y_n), t, r\big) - s_{real}\big((x_S, y_S), (x_n, y_n), t, r\big) \leq tol(et, t, r) \\ &\quad mit \; : \\ &\quad s_{theo}\big((x_S, y_S), (x_n, y_n), t, r\big) = \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2} \end{aligned} \tag{1.18}$$

Schließlich wird die Optimalitätsbedingung $\neg\big(\exists \widehat{x_S} \, \exists \widehat{y_S} \; : \; K_T(\widehat{x_S}, \widehat{y_S}, t, r) < K_T(x_S, y_S, t, r)\big)$, die in der Formel 1.15 als prädikatenlogisches Äquivalent der angestrebten Kostenminimierung enthalten ist, kritisch betrachtet. Dieser Optimalitätsbedingung lässt sich eine „utopische“ Anmaßung von Wissen vorhalten. Diese Vorhaltung ist berechtigt, weil die Existenzquantoren in der Optimalitätsbedingung ein „Allwissen“ des Entscheidungsträgers ausdrücken: Demzufolge *weiß* der Entscheidungsträger, dass *kein* alternativer Standort $O_{\hat{S}}$ mit den Koordinaten $(\widehat{x_S}, \widehat{y_S})$ existiert, der geringere Transportkosten K_T als der optimale Standort O_S mit den Koordinaten (x_S, y_S) aufweist. Dieser Wissensanspruch hinsichtlich der objektiven *Nichtexistenz* eines alternativen Standorts mit geringeren Transportkosten lässt sich bei näherer Betrachtung nicht ernsthaft aufrechterhalten. Denn kein Entscheidungsträger vermag in der betrieblichen Realität die – potenziell unendliche – Mannigfaltigkeit alternativer Standorte vollständig zu überblicken, unter denen auch ein Standort mit geringeren Transportkosten als im – dann nicht mehr – „optimalen“ Standort O_S verborgen sein könnte.

139 Weiterführende Präzisierungen des intendierten Anwendungsbereichs lassen sich grundsätzlich mittels zusätzlicher notwendiger Bedingungen für die Erfüllung des Prädikats $IAB_{TKM}(et, t, r)$ einführen. Daher besitzen die hier vorgestellten Formeln 1.17 und 1.18 lediglich exemplarischen Charakter, wie sich der intendierte Anwendungsbereich der Theorie für das klassische Transportkostenmodell sukzessive präzisieren lässt.

140 Der Einfachheit halber wird darauf verzichtet, zwischen zwei Prädikaten $IAB_{TKM.1}(et, t, r)$ und $IAB_{TKM.2}(et, x_S, y_S, x_n, y_n, t, r)$ mit jeweils unterschiedlicher „Stelligkeit“ des Prädikatsarguments zu differenzieren, obwohl dies strenggenommen erforderlich wäre.

141 Diese „Präzisionspräferenz“ entspricht dem „Approximationsapparat“, den Balzer (2009), S. 50 u. 240 ff., für Theorien aus der Perspektive des „non statement view“ anführt.

Um diese Präsupposition eines „allwissenden“ Entscheidungsträgers zu vermeiden, muss die Optimalitätsbedingung in der Formel 1.15 mit der Suggestion der *objektiven Nichtexistenz* eines alternativen Standorts mit geringeren Transportkosten durch das *subjektive Nichtwissen* des Entscheidungsträgers hinsichtlich alternativer Standorte mit geringeren Transportkosten ersetzt werden. Hierdurch wird das Optimierungskalkül des klassischen Transportkostenmodells und der zugrunde liegenden, hier rekonstruierten Theorie in Bezug auf das *Wissen relativiert*, über das ein Entscheidungsträger *et* aktuell (im Zeitpunkt *t* und im Raumpunkt *r*) verfügt. Dieses „Verfügen über Wissen“ wird durch das neu eingeführte Prädikat $W(et,t,r)$ ausgedrückt.[142] Mit seiner Hilfe lässt sich die frühere Formel 1.15 wie folgt „wissensrelativiert“ wiedergeben:

$$\begin{aligned}&\forall x_S \forall y_S \forall t \forall r \forall et \, : \, \big(RAT(et,t,r) \wedge ZS(et,t,r) = (K_T, min, det, stat, nsoz)\big) \rightarrow \ldots \\ &\quad \Big(AUS_{opt}\,(et,t,r,x_S,y_S) \rightarrow \ldots \\ &\qquad \neg\big(\exists \widehat{x_S}\, \exists \widehat{x_S} \, : \, K_T(W(et,t,r), \widehat{x_S}, \widehat{y_S}) < K_T(W(et,t,r), x_S, y_S)\big)\Big)\end{aligned} \tag{1.19}$$

Die Formel 1.19 fordert als „epistemisch modifizierte“ Optimalitätsbedingung nur noch, dass der Entscheidungsträger *et* gemäß seinem aktuellen Wissenstand im Zeitpunkt *t* und im Raumpunkt *r* keinen alternativen Standort $O_{\widehat{S}}$ mit den Koordinaten $(\widehat{x_S}, \widehat{y_S})$ kennt, der geringere Transportkosten K_T als der optimale Standort O_S mit den Koordinaten (x_S, y_S) aufweist. Diese Relativierung hinsichtlich des aktuellen Wissensstands des Entscheidungsträgers[143] hat gravierende Konsequenzen in Bezug auf die „logische Qualität“ der Theorie, die dem klassischen Transportkostenmodell zugrunde liegt. Denn für die Berücksichtigung des aktuellen Wissensstands des Entscheidungsträgers ist es erforderlich, von der üblichen deduktiven (Prädikaten-)Logik zu einer *non-monotonen*[144] „epistemischen Logik“[145] überzugehen. Eine solche non-monotone Logik zeichnet sich dadurch aus, dass logisch-mathematische Beweise (wie z. B. hinsichtlich der Optimalität eines Standorts im klassischen Transportkostenmodell), die einmal korrekt durchgeführt wurden, nicht mehr – wie es in der deduktiven Logik der Fall ist – ihre Gültigkeit „für immer“ behalten. Vielmehr kann sich ein logisch-mathematischer Beweis, der angesichts eines aktuellen Wissensstands korrekt erfolgte, in der Zukunft vor dem Hintergrund eines veränderten – entweder verringerten (Verlernen) oder auch, vor allem, erweiterten (Lernen) – Wissensstands als falsch herausstellen. Diese „temporale Instabilität“ der Gültigkeit logisch-mathematischer Beweise stellt

142 In einer weiter ausgebauten Theorievariante sollte näher untersucht werden, ob sich das „Verfügen über Wissen“ durch ein Prädikat wie $W(et,t,r)$ ausgedrückt werden soll oder ob anspruchsvollere logische Konstruktionen zu bevorzugen sind, wie z. B. „logische Operatoren“ (einer epistemisch erweiterten Modallogik?), welche die Wissensverfügbarkeit symbolisieren.

143 Eine solche Relativierung hinsichtlich des aktuellen Wissensstands des Entscheidungsträgers sollte nach Ansicht der Verfasser in allen „guten“ Entscheidungsmodellen, zumindest in den ihnen zugrunde liegenden Theorien, auch im Rahmen der normativen Entscheidungstheorie erfolgen, um die Anmaßung von „Allwissenheit“ hinsichtlich des Entscheidungsträgers auszuschließen. Allerdings verwundert es nicht, dass diese wissensbezogene Relativierung von Entscheidungsmodellen und Entscheidungstheorie weithin unterbleibt, weil die nachfolgend skizzierten Komplikationen einer resultierenden non-monotonen epistemischen Logik vermutlich gescheut werden (sofern sie überhaupt bekannt sind). Vgl. zur Relativierung von KI-Systemen auf den jeweils aktuellen Wissensstand eines solchen KI-Systems oder seiner Benutzer Beckstein (1996), S. 61 ff. u. 70 ff. (im Kontext von begründungsverwaltenden KI-Systemen).

144 Vgl. zur Non-Monotonie „logischer“ Inferenzsysteme, wie z. B. begründungsverwaltender Systeme, Beierle und Kern-Isberner (2006), S. 202 ff.; Beckstein (1996), S. 79 ff.

145 Vgl. zu Beispielen für eine epistemische Logik Gethmann (2004a), S. 646 ff.; Spies (2004), S. 202 ff.; Stegmüller (1987), S. 175 ff.

eine große Herausforderung an Modelle und Theorien dar, in denen die Veränderung des Wissens eines Entscheidungsträgers im Zeitverlauf explizit zugelassen wird.[146]

Diese Herausforderung beruht darauf, dass sich das Wissen $W(et, t, r)$ eines Entscheidungsträgers *et* vor allem im Zeitverlauf[147] verändern kann. Daher ist es möglich, dass sich *früher richtige* Urteile hinsichtlich der (auf das aktuell verfügbare Wissen relativierten) „Nichtexistenz“ von alternativen Standorten mit geringeren Kosten *nachträglich* als *falsch* herausstellen, weil neues Wissen über alternative Standorte erlangt wird, die tatsächlich geringere Transportkosten aufweisen als der bislang für optimal gehaltene Standort. Folglich kann im klassischen Transportkostenmodell kein Standort „für immer“ als optimal erkannt werden, sondern seine Optimalität hängt stets vom jeweils aktuell verfügbaren Wissen des Entscheidungsträgers *et* ab. Auf diese Weise wird es möglich, in einem „epistemisch überarbeiteten“ Transportkostenmodell mithilfe einer non-monotonen Logik auch *Lernprozesse* des Entscheidungsträgers *et* modellendogen und explizit zu erfassen.[148] Dies stellt nach Einschätzung der Verfasser dieses Beitrags einen wesentlichen Fortschritt der „Realitätsadäquanz“ von Entscheidungsmodellen und Theorien dar, wie hier in exemplarischer Weise anhand des klassischen Transportkostenmodells und der ihm zugrunde liegenden Theorie demonstriert wurde.

Zusammenfassend wird „die“[149] Theorie präsentiert, die zuvor als theoretisches Fundament des klassischen Transportkostenmodells schrittweise rekonstruiert wurde. Auf der Grundlage der jeweils überarbeiteten Formeln resultiert anstelle des ursprünglich vorgelegten Formelsystems mit den Formeln 1.5 bis 1.9 das nachfolgende Formelsystem mit den Formeln 1.20 bis 1.26:

$$\begin{aligned}
&\forall x_S \forall y_S \forall t \forall r \forall et \;:\; (RAT(et,t,r) \wedge ZS(et,t,r) = (K_T, min, det, stat, nsoz)) \rightarrow \ldots \\
&\quad \Big(AUS_{opt}(et,t,r,x_S,y_S) \rightarrow \ldots \\
&\qquad \neg\big(\exists \widehat{x}_S \exists \widehat{x}_S \;:\; K_T\big(W(et,t,r), \widehat{x}_S, \widehat{y}_S\big) < K_T\big(W(et,t,r), x_S, y_S\big)\big)\Big)
\end{aligned} \tag{1.20}$$

146 Diese Herausforderung spielt beispielsweise im Rahmen der Erforschung der Künstlichen Intelligenz (KI) eine große Rolle im Zusammenhang mit sogenannten begründungsverwaltenden Systemen, wie etwa den „assumption-based truth maintenance systems“ (ATMS). In solchen begründungsverwaltenden Systemen werden aus einer zunächst vorgegebenen Menge von Prämissen von Computern mittels deduktiver Inferenzregeln (Schlussfolgerungsregeln) „intelligente“ Schlüsse gezogen, wie z. B. in betriebswirtschaftlichen Kontexten hinsichtlich der Lösung eines Planungs- oder eines Diagnoseproblems. Falls sich mindestens eine der ursprünglich vorliegenden Prämissen im Zeitverlauf – beispielsweise durch „Erkenntnisgewinn“ – ändert, sind KI-Systeme mit ATMS-Fähigkeit in der Lage, erstens zu überprüfen, ob die ursprünglich hergeleiteten Pläne bzw. Diagnosen weiterhin Bestand haben, und zweitens – falls nicht – eigenständig nur diejenigen Planungs- bzw. Diagnoseschritte zurückzunehmen, die von den Prämissenänderungen betroffen sind („Net-Change-Prinzip“), und durch die Herleitung neuer Planungs- bzw. Diagnoseergebnisse auf der Grundlage der veränderten Prämissen zu ersetzen. Vgl. zu solchen KI-Systemen mit ATMS-Fähigkeit z. B. Beierle und Kern-Isberner (2006), S. 206 ff. (begründungsverwaltende Systeme im Allgemeinen) sowie S. 228 ff. (ATMS-Fähigkeit im Besonderen); Beckstein (1996), S. 13 ff., 31 ff., 79 ff., 115 ff. u. 295 ff. (begründungsverwaltende Systeme im Allgemeinen mit mehreren speziellen Varianten) sowie S. 169 ff., 229 ff. u. 259 ff. (ATMS-Fähigkeit im Besonderen).

147 Wissensveränderungen des Entscheidungsträgers in Abhängigkeit vom Raumpunkt *r*, wie z. B. in betriebswirtschaftlichen Kontexten der Branche, spielen nach Einschätzung der Verfasser für betriebswirtschaftliche Modelle und Theorien – bis zum Nachweis des Gegenteils – keine Rolle.

148 Gleiches gilt „mutatis mutandis“ für analoge Verlern- oder Vergessensprozesse, die jedoch aus betriebswirtschaftlicher Perspektive weniger interessant sind.

149 Wegen der „Freiheitsgrade“ der Theorierekonstruktion liegt streng genommen nicht genau eine rekonstruierte Theorie vor, sondern eine Mannigfaltigkeit von alternativen Theorierekonstruktionen, über deren „Angemessenheit“ sich trefflich streiten lässt.

$$\wedge \quad \forall x_S \forall y_S \; : \; K_T(x_S, y_S) = k_T \cdot \left(\sum_{n=1}^{N} m_n \cdot s\big((x_S, y_S), (x_n, y_n)\big) \right) \tag{1.21}$$

$$\begin{aligned} \wedge \quad & \forall x_S \forall y_S \forall x_n \forall y_n \forall t \forall r \; : \\ & s_{theo}((x_S, y_S), (x_n, y_n), t, r) = \sqrt{(x_S - x_n)^2 + (y_S - y_n)^2} \end{aligned} \tag{1.22}$$

$$\wedge \quad \forall x_S \forall y_S \; : \; x_S \geq 0 \wedge y_S \geq 0 \tag{1.23}$$

$$\begin{aligned} \wedge \quad & \forall x_S \forall y_S \; : \; OPT(x_S, y_S) \rightarrow \ldots \\ & (x_S, y_S) \in \big\{\{(x, y) \,|\, a_n \cdot x + b_n \cdot y \leq / \geq c_n \text{ für } n = 1, \ldots, N\}\big\} \end{aligned} \tag{1.24}$$

$$\begin{aligned} \wedge \quad & \forall et \forall t \forall r \; : \; IAB_{\,TKM} \rightarrow \ldots \\ & RAT(et, t, r) \wedge ZS(et, t, r) = (K_T, min, det, stat, nsoz) \end{aligned} \tag{1.25}$$

$$\begin{aligned} \wedge \quad & \forall et \forall x_S \forall y_S \forall x_n \forall y_n \forall t \forall r \; : \; IAB_{\,TKM}\,(et, x_S, y_S, x_n, y_n, t, r) \rightarrow \ldots \\ & s_{theo}\big((x_S, y_S), (x_n, y_n), t, r\big) - s_{real}\big((x_S, y_S), (x_n, y_n), t, r\big) \leq tol(et, t, r) \end{aligned} \tag{1.26}$$

Am Rande sei erwähnt, dass die Rekonstruktion der Theorie, die dem klassischen Transportkostenmodell nach Maßgabe des voranstehenden Formelsystems zugrunde liegt, nicht eindeutig ist. Wie die vorangehenden Erörterungen, die vom Formelsystem der Formeln 1.5 bis 1.9 zum Formelsystem der Formeln 1.20 bis 1.26 führten, erkennen lassen, erweist sich die Rekonstruktionsarbeit als ein „kreativer Prozess“, dessen Ergebnis nicht durch einen „Algorithmus“ eindeutig vorgegeben ist, sondern zahlreiche subjektive Konstruktionsentscheidungen der Verfasser umfasst. Daher könnten andere Autoren zu einer anderen Theorie gelangen, die dem klassischen Transportkostenmodell zugrunde liegt. Die Verfasser würden es begrüßen, wenn solche alternativen Rekonstruktionsvorschläge die theoretische Diskussion um das klassische Transportkostenmodell befruchten würden.

1.4 Reflexion der Theorierekonstruktion für das klassische Transportkostenmodell

1.4.1 Zusammenfassung wesentlicher Ergebnisse zum „Thesenstreit“

Die Ausführungen in Kapitel 1.3.2 zur Rekonstruktion einer Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, bieten wesentliche Erkenntnisse zum „Thesenstreit“, der zu Beginn dieses Beitrags anhand von fünf Thesen entfaltet wurde.

Die (erste) *These der Theorielosigkeit* wird widerlegt, weil zumindest für den Bereich der Betriebswirtschaftslehre in exemplarischer Weise anhand der theoretischen Rekonstruktion des klassischen Transportkostenmodells aufgezeigt wurde, dass sie – die Betriebswirtschaftslehre – in ihrem Basisbereich über eine „wissenschaftlich akzeptable“ theoretische Fundierung verfügt. Diese Erkenntnis lässt sich auf den Bereich der Wirtschaftsinformatik übertragen, weil sich Wirtschaftsinformatik und Betriebswirtschaftslehre hinsichtlich ihrer theoretischen Fundierung nach Einschätzung der Verfasser nicht grundsätzlich unterscheiden.

A fortiori wird auch die (zweite) *These der Theorielosigkeit* widerlegt, weil sich die behauptete prinzipielle Unmöglichkeit einer theoretischen Fundierung von Wirtschaftsinformatik und Betriebswirtschaftslehre durch den exemplarischen Beweis des Gegenteils nicht aufrechterhalten lässt.

Die (dritte) *These der schwachen Theoriefundierung* entzieht sich hingegen der Argumentation des vorliegenden Beitrags. Denn diese These geht von der Voraussetzung aus, dass Wirtschaftsinformatik und Betriebswirtschaftslehre in ihrem Basisbereich über eine „wissenschaftlich akzeptable“ theoretische Fundierung verfügen, der hierbei zugrunde gelegte wissenschaftliche Theoriebegriff aber nicht streng definiert wird („schwaches Theorieverständnis“), sondern ohne Bezug auf nomische Hypothesen und überwiegend natürlichsprachlich umschrieben wird. Die *Voraussetzung* dieser dritten These wird im vorliegenden Beitrag nicht erfüllt, weil er einen streng definierten wissenschaftlichen Theoriebegriff zugrunde legt und sich explizit auf (nicht-triviale) nomische Hypothesen bezieht.

Die (fünfte) *These der starken nomisch-formalsprachlichen Theoriefundierung* wird unmittelbar bestätigt, weil eine formalsprachlich verfasste Theorie rekonstruiert wurde, die dem klassischen Transportkostenmodell zugrunde liegt und auch mindestens eine nicht-triviale nomische Hypothese umfasst.[150]

Hierdurch wird auch die (vierte) *These der starken nomischen Theoriefundierung* bestätigt, weil sich die formalsprachliche Formulierung der rekonstruierten Theorie ohne Schwierigkeiten in eine natürlichsprachliche Theoriebeschreibung „übersetzen“ lässt. Die natürlichsprachlichen Erläuterungen zu den o. a. prädikatenlogischen Formeln, insbesondere ihre natürlichsprachlichen Paraphrasierungen, verdeutlichen diese Übersetzungsmöglichkeit.

Die Verfasser hoffen, sie konnten mit ihrer Rekonstruktion einer Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, und ihren darauf beruhenden Antworten zum eingangs vorgestellten „Thesenstreit“ einen Beitrag leisten, der die vielfältigen und tiefgründigen Publikationen von Frank zu den erkenntnis- und wissenschaftstheoretischen Grundlagen der Wirtschaftsinformatik – hier aus der eng verwandten betriebswirtschaftlichen Perspektive – im Hinblick auf eine mögliche Präzisierung des jeweils (implizit oder explizit) vorausgesetzten Theorieverständnisses ergänzt. Sollten die hier vorgetragenen Ansichten, vor allem theoretischen Rekonstruktionen, auf Widerspruch stoßen, wäre dies im Sinne einer Befruchtung des wissenschaftlichen Diskurses, dessen Bedeutung für die disziplinäre „Diskussionskultur“ von Frank immer wieder betont wurde, ausdrücklich zu begrüßen.

1.4.2 Limitationen der vorgestellten Theorierekonstruktion

Hinsichtlich der hier vorgestellten Rekonstruktion einer Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, sind mehrere Einschränkungen („Limitationen“) zu beachten. Sie werden im Folgenden nur kurz angeführt, weil sie zumeist aus den voranstehenden Ausführungen unmittelbar hervorgehen.

Zunächst ist grundsätzlich anzumerken, dass die Theorierekonstruktion keineswegs „notwendig“ oder „eindeutig“ ist. Stattdessen floss in die Rekonstruktionsarbeit eine Vielzahl

150 Die rekonstruierte Theorie umfasst zwar zwei nicht-triviale nomische Hypothesen. Aber das erste Exemplar, das sich auf die Distanzmessung zwischen zwei Standorten bezieht, erweist sich aus betriebswirtschaftlicher Perspektive als „uninteressant“. Daher verbleibt auf jeden Fall eine nicht-triviale, aus betriebswirtschaftlicher Perspektive interessante nomische Hypothese.

von – oftmals impliziten, zuweilen auch expliziten – Gestaltungsentscheidungen ein, die auch in anderer Weise hätten getroffen werden können. Folglich existieren vermutlich mehrere alternative Theorien, die mit gleichem Anspruch auf „Rekonstruktionserfolg" dem klassischen Transportkostenmodell zugrunde gelegt werden können. Solche alternativen Rekonstruktionsangebote sollten von Dritten präsentiert werden, um sich über die Güte der rekonstruierten Theorien im kritisch-konstruktiven Diskurs miteinander austauschen zu können. Hierfür werden operationale Kriterien zur Beurteilung der Rekonstruktionsgüte benötigt, die in diesem Beitrag noch nicht thematisiert wurden, sondern weiterführender Forschungsarbeiten bedürfen.

Die Theorierekonstruktion beruhte vor allem auf der These der starken nomisch-formalsprachlichen Theoriefundierung, die mit einem „starken" Theorieverständnis kombiniert wurde. Dieses Theorieverständnis setzt voraus, dass eine zu rekonstruierende Theorie einen systematischen Aussagenzusammenhang im Sinne des Theorienkonzepts der „statement view" darstellt und mindestens eine nicht-triviale nomische Hypothese umfasst. In formalsprachlicher Hinsicht wurde auf die Ausdruckskraft der Prädikatenlogik (erster Stufe) gesetzt, die schließlich im Sinne einer „epistemischen" Logik erweitert wurde. Es wurde also in die Theorierekonstruktion eine *Vielzahl* von (hier nur exemplarisch angeführten) *Prämissen* „investiert", die als solche keineswegs „selbstverständlich" sind, sondern sich mit jeweils guten Gründen in Zweifel ziehen lassen. Es ist möglich, jeden solchen Zweifel als eine Limitation der hier vorgestellten Theorierekonstruktion aufzufassen. Beispielsweise können die starke formalsprachliche Fokussierung sowie die „Engführung" des Erkenntnisinteresses entlang der Leitlinie von nicht-trivialen nomischen Hypothesen als Limitationen betrachtet werden, die den Blick von anderen – nicht-formalsprachlichen bzw. nicht-nomischen – Theorieaspekten abgelenkt haben.[151] Insbesondere die formalsprachliche Fokussierung auf die Prädikatenlogik könnte als nicht-notwendige Einschränkung der Erkenntnismöglichkeiten anlässlich der Theorierekonstruktion gesehen werden.[152]

Darüber hinaus ist selbstkritisch einzuwenden, dass die Voraussetzung des „starken" Theorieverständnisses eventuell „weichere" Erkenntnisse ausgegrenzt hat, die nur im Rahmen des „schwachen" Theorieverständnisses möglich erscheinen.[153] Diese Limitation zeigt sich besonders deutlich hinsichtlich des o. a. Eingeständnisses, dass im vorliegenden Beitrag zur These der schwachen Theoriefundierung aufgrund des „radikal" unterschiedlichen Theorieverständnisses keine substanzielle Argumentation erfolgte. Es wäre begrüßenswert, wenn die Anhänger des „schwachen" Theorieverständnisses eine „alternative" Theorie rekonstruieren würden, die aus ihrer Sicht dem klassischen Transportkostenmodell zugrunde gelegt werden kann und hierbei nur auf die Konstituenten des „schwachen" Theorieverständnisses zurückgreift, also vor allem auf Formalsprachlichkeit und Nomizität verzichtet.

Eine noch gravierendere Limitation ergibt sich aus der Beschränkung auf das „etablierte" Theorienkonzept des „statement view". Durch diesen Mainstream-Ansatz werden Erkenntnispotenziale ausgeblendet, die sich erst im wesentlich reichhaltigeren, aber auch

151 Wer diese Ansicht teilt, sei herzlich dazu einladen, derart übersehene Theorieaspekte – die über das klassische Transportkostenmodell nicht hinausgehen – konkret zu benennen.

152 Die Verfasser nehmen diese Limitation sehr ernst. Sie haben selbst anlässlich der Erweiterung der Theorierekonstruktion in Richtung auf eine „epistemische" Logik skizziert, dass die „klassische" Prädikatenlogik mit ihrer Monotonieprämisse nicht ausreicht, um die wissens-, insbesondere lernbedingte Non-Monotonie von Schlussfolgerungen „angemessen" zu berücksichtigen, die sich u. a. auch auf das kontingente Wissen der Entscheidungsträger beziehen.

153 Wer diese Ansicht teilt, sei abermals „ermuntert", derart übersehene Erkenntnisse in Bezug auf das klassische Transportkostenmodell konkret zu beschreiben.

anspruchsvolleren Theorienkonzept des „non statement view“ aus dem Bereich der Analytischen Wissenschaftstheorie eröffnen würden. Diese zusätzlichen Erkenntnispotenziale betreffen vor allem den Umgang mit und die Eliminierung von T-theoretischen Begriffen, die dem Theorienkonzept des „statement view“ grundsätzlich fremd sind. Wie wünschenswert die systematische Behandlung von T-theoretischen Begriffen grundsätzlich wäre, zeigte sich angesichts der Erörterung der „zirkulären“ Charakteristik des – aus wissenschaftstheoretischer Perspektive – äußerst problematischen Rationalitätsprädikats. Hierauf konnte jedoch nicht näher eingegangen werden, weil der vorliegende Beitrag auf das Theorienkonzept des „statement view“ beschränkt bleibt. Gründe wie „Anschlussfähigkeit“ an den wissenschaftstheoretischen Mainstream von Wirtschaftsinformatik und Betriebswirtschaftslehre lassen sich zwar zur Rechtfertigung dieses Vorgehens anführen, können aber die hierdurch bedingte Limitation der Erkenntnismöglichkeiten in Bezug auf „vertrackte“ Begriffe wie das Rationalitätsprädikat nicht „wegerklären“.

Weitere Limitationen ergeben sich aus Details der Theorierekonstruktion, die sich nicht unmittelbar in Bezug auf Eigenarten des zugrunde liegenden klassischen Transportkostenmodells rechtfertigen lassen.[154] Ein Beispiel mag als „pars pro toto“ zur Verdeutlichung ausreichen. So wurde in der hier vorgestellten Theorierekonstruktion stets nur jeweils ein beliebiger Entscheidungsträger *et* betrachtet. Dies stellt eine nicht-notwendige Limitation der rekonstruierten Theorie im Hinblick auf Individualentscheidungen bei der Standortauswahl dar. Stattdessen hätten auch Gruppenentscheidungen in die Theorierekonstruktion einbezogen werden können. Dies hätte zwar hinsichtlich der theoretisch darzustellenden Meinungsbildungsprozesse bei Gruppenentscheidungen – unter Beachtung heterogener (Sozial-) Präferenzen und personell heterogener Wissensstände – zu erheblichem formalsprachlichen Rekonstruktionsaufwand geführt, wäre aber seitens des klassischen Transportkostenmodells, das sich hinsichtlich der standortbezogenen Entscheidungsfindungs- und Personalstruktur nicht festlegt, auch nicht ausgeschlossen gewesen.[155]

1.4.3 Ausblick auf weiterführende Forschungsperspektiven

Weiterführende Forschungsperspektiven ergeben sich unmittelbar aus den Limitationen, die voranstehend skizziert wurden, und aus Problemen der Theoriekonstruktion für das klassische Transportkostenmodell, die im Rahmen des Kapitels 1.3.2 geleistet wurde. Unter Verweis auf diese bereits erfolgten Ausführungen sollte es ausreichen, weiterführende Forschungsperspektiven kurz aufzulisten:

1. Alternative Rekonstruktionsangebote für Theorien, die dem klassischen Transportkostenmodell zugrunde liegen, sollten von Dritten präsentiert werden, um einen kritisch-konstruktiven Diskurs über – konzeptionell unterschiedlich aufgestellte – Theorierekonstruktionen zur Diskussion zu stellen.

154 Dagegen werden Details der Theorierekonstruktion, die lediglich Eigenarten des zugrunde liegenden klassischen Transportkostenmodells reflektieren, nicht als Limitationen der Theorierekonstruktion betrachtet, weil sie zu einer „modellgetreuen“ Theorierekonstruktion gehören. Beispielsweise stellen die Einschränkungen der rekonstruierten Theorie auf nur einen Zielinhalt (die Transportkosten), die Außerachtlassung von mehreren Perioden und die Ausblendung von Unsicherheit – insbesondere hinsichtlich des Transportkostensatzes sowie der zu transportierenden Gütermengen – keine Limitationen der Theorierekonstruktion dar, sondern ergeben sich unmittelbar aus der Struktur des klassischen Transportkostenmodells.

155 Das hier angeführte Beispiel verdeutlicht zugleich in exemplarischer Weise das weiter oben vorgetragene Argument, dass in die Rekonstruktionsarbeit eine Vielzahl von Gestaltungsentscheidungen eingeflossen ist, die auch in anderer Weise hätten getroffen werden können.

2. Kriterien für die Beurteilung der Güte alternativer Theorierekonstruktionen sollten „lagerübergreifend“[156] entwickelt werden.

3. Das „vertrackte“ Rationalitätsprädikat $RAT(et, t, r)$ sollte durch weniger kritikanfällige Konstrukte ersetzt werden. Dazu zählen im Rahmen des „statement view“ notwendige oder hinreichende „Bedingungen für die Möglichkeit“ rationalen Handelns, wie sie früher in exemplarischer Weise angesprochen wurden. Ebenso kommt die Konzeptualisierung des Rationalitätsprädikats als ein T-theoretischer Begriff im Rahmen des „non statement view“ in Betracht, der alsdann z. B. mittels der RAMSEY-Eliminierung aus den „partiellen potenziellen Modellen“ einer strukturalistisch formulierten Theorie T für das klassische Transportkostenmodell zu entfernen wäre.[157]

4. Die Rekonstruktion einer Theorie, die dem klassischen Transportkostenmodell zugrunde liegt, mittels des Theorienkonzepts des „non statement view“ würde nicht nur den systematischen Umgang mit T-theoretischen Begriffen wie dem Rationalitätsprädikat erleichtern, sondern böte auch Hilfen, wie sich der intendierte Anwendungsbereich einer solchen Theorie präziser spezifizieren lässt, als es hier im Rahmen des „statement view“ angedeutet wurde.

5. Die Relativierung der Optimalitätsbedingung des klassischen Transportkostenmodells auf den jeweils aktuellen Wissensstand eines Entscheidungsträgers sollte im Kontext einer epistemischen Logik weiterentwickelt werden. Der hier vorgestellte Ansatz, zur Repräsentation dieses Wissensstands das Prädikat $W(et, t, r)$ zu verwenden, erscheint den Verfassern noch nicht ausgereift, weil es sich nur um eine wissensrelativierende „Reparatur“ innerhalb der konventionellen Prädikatenlogik (erster) Stufe handelt. Stattdessen wäre der Übergang zu einer „echten“ epistemischen Logik wünschenswert, in der – über konventionelle Prädikate hinaus – auch eigenständige „Operatoren“ zur Kennzeichnung des jeweils aktuellen Wissensstands eines Entscheidungsträgers verwendet werden. Außerdem wären die monotonen Schlussfolgerungsweisen (Inferenzregeln) der konventionellen deduktiven Prädikatenlogik durch komplexere Schlussfolgerungsweisen[158] zu ersetzen, die auch die Non-Monotonie von Schlussfolgerungen als Folge von Wissensveränderungen, insbesondere durch Lernprozesse der betroffenen Entscheidungsträger, zu bewältigen vermögen.

Wie sich anhand der voranstehenden Auflistung zeigt, bietet selbst ein derart „altehrwürdiges“ und „schlichtes“ Modell wie das klassische Transportkostenmodell hinsichtlich der Rekonstruktion einer Theorie, die ihm zugrunde liegt, noch eine respektable „Spielwiese“ für eine vertiefte und – vor allem auch – verbesserte Theorierekonstruktion. Die Verfasser verleihen ihrer Hoffnung Ausdruck, dass der hier vorgelegte Beitrag ein wenig dazu beizutragen vermag, solche weiterführenden Rekonstruktionsarbeiten anzustoßen. Sie könnten

156 Mit „Lagern“ sind hier Wissenschaftler-Gemeinschaften gemeint, die beispielsweise unterschiedliche Theorienkonzepte („statement view“ versus „non statement view“) und verschiedene Theorieverständnisse („starkes“ versus „schwaches“ Theorieverständnis) bevorzugen.

157 Vgl. Zelewski (1993), S. 123 f.

158 In dieser Hinsicht wird auf die bereits erwähnte ATMS-Fähigkeit von KI-Systemen verwiesen. Als begründungsverwaltende Systeme – hier im Besonderen vom Typ der „assumption-based truth maintenance systems“ (ATMS) – sind sie in der Lage, mit der Non-Monotonie von Schlussfolgerungen systematisch umzugehen, wenn sich Wissen (vor allem Prämissen) im Zeitverlauf ändert, auf dem die früher gezogenen Schlussfolgerungen beruhten.

sich auch auf die Theorierekonstruktion für wesentlich anspruchsvollere, vor allem realitätsnähere Modelle der Wirtschaftsinformatik und der Betriebswirtschaftslehre erstrecken als das hier untersuchte klassische Transportkostenmodell.

Literatur

Albert, H. (1957). „Wissenschaftstheorie“. In: *Swiss Journal of Economics and Statistics* 93. Wiederabdruck in: Topitsch, E. (Hrsg.): Logik der Sozialwissenschaften. 10. Aufl., Königstein 1980, S. 60–76.

Albert, H. (1976). „Theorie und Prognose in den Sozialwissenschaften“. In: *Handwörterbuch der Betriebswirtschaft, Band 3.* Hrsg. von E. Grochla und W. Wittmann. 4. Aufl. Stuttgart, S. 4674–4692.

Albert, H. (1991). *Traktat über kritische Vernunft.* 5. Aufl. Tübingen.

Anderson, C. (2008). „The End of Theory: The Data Deluge Makes the Scientific Method Obsolete“. In: *Wired* (23.06.2008). URL: `https://www.wired.com/2008/06/pb-theory/`.

Avison, D. und Malaurent, J. (2014). „Is theory king?: questioning the theory fetish in information systems“. In: *Journal of Information Technology* 29.4, S. 327–336.

Baldridge, D., Floyd, S. und Markóczy, L. (2004). „Are managers from Mars and academicians from Venus? Toward an understanding of the relationship between academic quality and practical relevance“. In: *Strategic Management Journal* 25.11, S. 1063–1074.

Balzer, W. (2009). *Die Wissenschaft und ihre Methoden – Grundsätze der Wissenschaftstheorie.* 2. Aufl. Freiburg, München.

Balzer, W. und Brendel, K. (2019). *Theorie der Wissenschaften.* Wiesbaden.

Beck, A. und Sabach, S. (2015). „Weiszfeld's Method: Old and New Results“. In: *Journal of Optimization Theory and Applications* 164.1, S. 1–40.

Beckstein, C. (1996). *Begründungsverwaltung – Grundlagen, Systeme und Algorithmen.* Stuttgart, Leipzig.

Beierle, C. und Kern-Isberner, G. (2006). *Methoden wissensbasierter Systeme: Grundlagen – Algorithmen – Anwendungen.* 3. Aufl. Wiesbaden.

Bichler, M., Frank, U., Avison, D., Malaurent, J., Fettke, P., Hovorka, D., Krämer, J., Schnurr, D., Müller, B., Suhl, L. und Thalheim, B. (2016). „Theories in Business and Information Systems Engineering“. In: *Business & Information Systems Engineering* 58.4, S. 291–319.

Boudon, R. (1991). „What Middle-Range Theories Are“. In: *Contemporary Sociology* 20.4, S. 519–522.

Brimberg, J. (1995). „The Fermat-Weber location problem revisited“. In: *Mathematical Programming* 71.1, S. 71–76.

Brimberg, J. (2003). „Further notes on convergence of the Weiszfeld algorithm“. In: *Yugoslav Journal of Operations Research* 13.2, S. 199–206.

Brühl, R. (2021). *Wie Wissenschaft Wissen schafft – Wissenschaftstheorie und -ethik für die Sozial- und Wirtschaftswissenschaften.* 3. Aufl. München.

Cánovas, L., Cañavate, R. und Marín, A. (2002). „On the convergence of the Weiszfeld algorithm“. In: *Mathematical Programming* 93.2, S. 327–330.

Cecchini, P., Catinat, M. und Jacquemin, A. (1988a). *The European Challenge 1992 – The Benefits of a Single Market.* (mit Robinson, J.) Aldershot.

Cecchini, P., Catinat, M., Jacquemin, A. und Stabenow, M. (1988b). *Europa '92 – Der Vorteil des Binnenmarktes.* (mit Robinson, J.) Baden-Baden.

Chandrasekaran, R. und Tamir, A. (1989). „Open questions concerning Weiszfeld's algorithm for the Fermat-Weber location problem“. In: *Mathematical Programming* 44.1–3, S. 293–295.

Chopra, S. und Meindl, P. (2014). *Supply Chain Management – Strategie, Planung und Umsetzung*. 5. Aufl. Hallbergmoos.

Chukwuere, J., Lubbe, S., Meyer, J. und Klopper, R. (2018). „Rigour versus Relevance in Information Systems Research in South Africa“. In: *Alternation* 25.1, S. 31–67.

Corsten, H. und Gössinger, R. (2016). *Produktionswirtschaft – Einführung in das industrielle Produktionsmanagement*. 14. Aufl. München.

Dodge, J., Ospina, S. und Foldy, E. (2005). „Integrating Rigor and Relevance in Public Administration Scholarship: The Contribution of Narrative Inquiry“. In: *Public Administration Review* 65.3, S. 286–300.

Domschke, W. (1996). „Standortplanung“. In: *Handwörterbuch der Produktionswirtschaft*. Hrsg. von W. Kern, H. Schröder und J. Weber. 2. Aufl. Stuttgart, S. 1912–1922.

Domschke, W. und Drexl, A. (1996). *Logistik: Standorte*. 4. Aufl. München.

Domschke, W. und Scholl, A. (2008). *Grundlagen der Betriebswirtschaftslehre – Eine Einführung aus entscheidungsorientierter Sicht*. 4. Aufl. Berlin, Heidelberg.

Duhem, P. (1998). *Ziel und Struktur der physikalischen Theorien*. Original: La Théorie physique: Son Objet, Sa Structure. Paris 1906 (2. Aufl., Paris 2014). Nachdruck der Übersetzung von Friedrich Adler (1908). Hamburg.

Eichhorn, W. (1979). „Die Begriffe Modell und Theorie in der Wirtschaftswissenschaft“. In: *Wissenschaftstheoretische Grundfragen der Wirtschaftswissenschaften*. Hrsg. von H. Raffée und B. Abel. München, S. 60–104.

Eiselt, H. und Sandblom, C.-L. (2022). *Operations Research – A Model-Based Approach*. 3. Aufl. Cham.

Eisenhardt, K. (1989). „Building Theories from Case Study Research“. In: *Academy of Management Review* 14.4, S. 535–550.

Flickinger, M., Tuschke, A., Gruber-Muecke, T. und Fiedler, M. (2014). „In search of rigor, relevance, and legitimacy: what drives the impact of publications?“ In: *Journal of Business Economics* 84.1, S. 99–128.

Frank, U. (1997). „Erfahrung, Erkenntnis und Wirklichkeitsgestaltung – Anmerkungen zur Rolle der Empirie in der Wirtschaftsinformatik“. In: *Wirtschaftsinformatik – Ergebnisse empirischer Forschung*. Hrsg. von O. Grün und L. Heinrich. Wien, New York, S. 21–35.

Frank, U. (1998a). „Die Evaluation von Artefakten: Eine zentrale Herausforderung der Wirtschaftsinformatik“. In: *Tagungsband des Workshops „Evaluation und Evaluationsforschung in der Wirtschaftsinformatik“, 05.06.1998 an der Johannes Kepler Universität Linz*. Hrsg. von L. Heinrich und I. Häntschel, S. 1–23 (eigene Paginierung).

Frank, U. (1998b). „Essential Research Strategies in the Information Systems Discipline: Reflections on Formalisation, Contingency and the Social Construction of Reality“. In: *The Systemist* 20.20, S. 98–113.

Frank, U. (1998c). „Wissenschaftstheoretische Herausforderungen der Wirtschaftsinformatik“. In: *Innovation in der Betriebswirtschaftslehre. Tagung der Kommission Wissenschaftstheorie, 02.-03.09.1997 in Marburg*. Hrsg. von E. Gerum. Wiesbaden, S. 91–118.

Frank, U. (1999). „Zur Verwendung formaler Sprachen in der Wirtschaftsinformatik: Notwendiges Merkmal eines wissenschaftlichen Anspruchs oder Ausdruck eines übertriebenen Szientismus?“ In: *Wirtschaftsinformatik und Wissenschaftstheorie – Bestandsaufnahme* *und Perspektiven.* Hrsg. von J. Becker, W. König, R. Schütte, O. Wendt und S. Zelewski. Wiesbaden, S. 129–160.

Frank, U. (2000). „Evaluation von Artefakten in der Wirtschaftsinformatik“. In: *Evaluation und Evaluationsforschung in der Wirtschaftsinformatik – Handbuch für Praxis, Lehre und Forschung.* Hrsg. von L. Heinrich und I. Häntschel. München, Wien, S. 35–48.

Frank, U. (2001). „Informatik und Wirtschaftsinformatik – Grenzziehungen und Ansätze zur gegenseitigen Befruchtung“. In: *Das ist Informatik.* Hrsg. von J. Desel. Berlin, Heidelberg, S. 47–66.

Frank, U. (2002). *Forschung in der Wirtschaftsinformatik: Profilierung durch Kontemplation – ein Plädoyer für den Elfenbeinturm.* Techn. Ber. 30. Koblenz: Universität Koblenz-Landau.

Frank, U. (2003a). „Einige Gründe für eine Wiederbelebung der Wissenschaftstheorie“. In: *Die Betriebswirtschaft* 63, S. 278–292.

Frank, U., Hrsg. (2003b). *Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik. Proceedings der Tagung, 05.-06.06.2003 in Koblenz.* Koblenz.

Frank, U., Hrsg. (2004a). *Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik – Theoriebildung und -bewertung, Ontologien, Wissensmanagement. Tagung „Wissenschaftstheorie in Ökonomie und Verwaltung“ der Kommission Wissenschaftstheorie, 05.-06.06.2003 in Koblenz.* Wiesbaden.

Frank, U. (2004b). „Zwischen Wettbewerbsorientierung und Qualitätssicherung – Universität und Wissenschaft in Zeiten des Wandels“. In: *Wissenschaftsmanagement* 10.1, S. 27–31.

Frank, U. (2006a). „Evaluation of Reference Models“. In: *Reference Modeling for Business Systems Analysis.* Hrsg. von P. Fettke und P. Loos. Hershey, S. 118–140.

Frank, U. (2006b). *Evaluation von Forschung und Lehre an Universitäten – Ein Diskussionsbeitrag.* Techn. Ber. 6. Essen: Institut für Informatik und Wirtschaftsinformatik (ICB), Universität-Duisburg Essen, Campus Essen.

Frank, U. (2006c). *Towards a Pluralistic Conception of Research Methods in Information Systems.* Techn. Ber. 7. revised version December 2006. Essen: Institut für Informatik und Wirtschaftsinformatik (ICB), Universität-Duisburg Essen, Campus Essen.

Frank, U. (2007a). „Ein Vorschlag zur Konfiguration von Forschungsmethoden in der Wirtschaftsinformatik“. In: *Wissenschaftstheoretische Fundierung und wissenschaftliche Orientierung der Wirtschaftsinformatik.* Hrsg. von F. Lehner und S. Zelewski. Berlin, S. 156–185.

Frank, U. (2007b). „Relevance of Research Implies Relevance to Researchers“. In: *Wirtschaftsinformatik* 49.5, S. 404–405.

Frank, U. (2008). „Herausforderungen der Wirtschaftsinformatik in Zeiten des Wandels“. In: *Quo vadis Wirtschaftsinformatik – Festschrift für Prof. Gerhard F. Knolmayer zum 60. Geburtstag.* Hrsg. von T. Myrach und R. Jung. Wiesbaden, S. 37–56.

Frank, U. (2009). „Die Konstruktion möglicher Welten als Chance und Herausforderung der Wirtschaftsinformatik“. In: *Wissenschaftstheorie und gestaltungsorientierte Wirtschaftsinformatik.* Hrsg. von J. Becker, H. Krcmar und B. Niehaves. Heidelberg, S. 161–173.

Frank, U. (2010). „Zur methodischen Fundierung der Forschung in der Wirtschaftsinformatik“. In: *Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz.* Hrsg. von H. Österle, R. Winter und W. Brenner. Nürnberg, S. 35–44.

Frank, U. (2011a). „An Ambivalent Concept“. In: *Business & Information Systems Engineering* 53.2. Als Abschnitt 5 zu: Loos, P.; Fettke, P.; Weißenberger, B.E.; Zelewski, S.; Heinzl, A.; Frank, U.; Iivari, J.: What in Fact Is the Role of Stylized Facts in Fundamental Research of Business and Information Systems Engineering? In: Business & Information Systems Engineering, Vol. 53 und Band 13 (2011), No. 2, S. 114–115 u. 117.

Frank, U. (2011b). „Ein ambivalentes Konzept“. In: *Wirtschaftsinformatik* 53.2. Als Abschnitt 5 zu: Loos, P.; Fettke, P.; Weißenberger, B.E.; Zelewski, S.; Heinzl, A.; Frank, U.; Iivari, J.: Welche Rolle spielen eigentlich stilisierte Fakten in der Grundlagenforschung der Wirtschaftsinformatik? In: Wirtschaftsinformatik, 53. Jg. und Band 13 (2011), Nr. 2, S. 117–119 u. 121.

Frank, U. (2013a). „Between Canonization and Differentiation“. In: *Business & Information Systems Engineering* 5.6. Als Abschnitt 4 zu: Loos, P.; Mettler, T.; Winter, R.; Goeken, M.; Frank, U.; Winter, A.: Methodological Pluralism in Business and Information Systems Engineering? In: Business & Information Systems Engineering, Vol. 55 und Band 15 (2013), No. 6, S. 456–458 u. 459.

Frank, U. (2013b). „Zwischen Kanonisierung und Differenzierung“. In: *Wirtschaftsinformatik* 55.6. Als Abschnitt 4 zu: Loos, P.; Mettler, T.; Winter, R.; Goeken, M.; Frank, U.; Winter, A.: Methodenpluralismus in der Wirtschaftsinformatik? In: Wirtschaftsinformatik, 55. Jg. und Band 15 (2013), Nr. 6, S. 461–462 u. 464.

Frank, U. (2014). „Higher Value of Research by Promoting Value for Researchers“. In: *Communications of the Association for Information Systems* 34. Article 43, S. 823–828.

Frank, U. (2015). „Modelle und die Hoffnung auf eine bessere Welt“. In: *Erwägen Wissen Ethik* 26.3, S. 372–375.

Frank, U. (2017a). *Mögliche zukünftige Welten und begründete Hoffnung als Ergänzung zu Wahrheitskonzepten: Skizze eines erweiterten Theoriebegriffs.* Abstract zu einem Vortrag im Rahmen der Kolloquiumsreihe „Wissenschaftstheoretische Betrachtungen in der Wirtschaftsinformatik“ am 24.04.2017 an der Universität Duisburg-Essen. Essen 2017 (eigene Paginierung). Ergänzende Präsentationsunterlagen im Internet, letzter Zugriff am 18.07.2023. URL: `https://www.wi-inf.uni-duisburg-essen.de/FGFrank/documents/Sonstige/Theoriebegriff-Vortrag-UF-2017-04-24.pdf`.

Frank, U. (2017b). „Theories in the Light of Contingency and Change: Possible Future Worlds and Well-Grounded Hope as a Supplement to Truth“. In: *Proceedings of the 50th Hawaii International Conference on System Sciences (HICSS-50), 03.-07.01.2017 in Big Island, Hawaii. Honolulu 2017.* Hrsg. von T. Bui und R. Sprague (Jr.), S. 5727–5736.

Frank, U. (2019). „Linguistic Structures in the Light of the Digital Transformation: Addressing the Conflict Between Reference and Change“. In: *The Art of Structuring: Bridging the Gap Between Information Systems Research and Practice.* Hrsg. von K. Bergener, M. Räckers und A. Stein. Cham, S. 41–54.

Frank, U. (2021a). „Die (Wirtschafts-)Informatik in Zeiten der digitalen Transformation – Einige Anmerkungen und Thesen“. In: *Der Weg in die „Digitalisierung“ der Gesellschaft – Was können wir aus der Geschichte der Informatik lernen?* Hrsg. von J. Pohle und K. Lenk. Marburg, S. 227–244.

Frank, U. (2021b). „Language, Change, and Possible Worlds – Philosophical Considerations of the Digital Transformation“. In: *Crisis and Critique: Philosophical Analysis and Current Events.* Hrsg. von A. Siegetsleitner, A. Oberprantacher, M.-L. Frick und U. Metschl. Bd. 28. Publications of the Austrian Ludwig Wittgenstein Society – New Series. Proceedings of the 42nd International Ludwig Wittgenstein Symposium. Berlin, Boston, S. 117–138.

Frank, U. und Bock, A. (2020). *Organisationsforschung und Wirtschaftsinformatik: Zeit für eine Annäherung?* ICB Research Reports – Forschungsberichte des ICB 67. Essen: Institut für Informatik und Wirtschaftsinformatik (ICB), Universität-Duisburg Essen, Campus Essen.

Frank, U., Klein, S., Krcmar, H. und Teubner, A. (1999). „Aktionsforschung in der WI – Einsatzpotentiale und Einsatzprobleme“. In: *Wirtschaftsinformatik und Wissenschaftstheorie – Grundpositionen und Theoriekerne*. Hrsg. von R. Schütte, J. Siedentopf und S. Zelewski. Essen, S. 71–90.

Frank, U., Schauer, C. und Wigand, R. (2008). „Different Paths of Development of Two Information Systems Communities: A Comparative Study Based on Peer Interviews“. In: *Communications of the Association for Information Systems* 22. Article 21, S. 389–412.

Frank, U. und Schauer, H. (2001). „Potentiale und Herausforderungen des Wissensmanagements aus der Sicht der Wirtschaftsinformatik“. In: *Wissen in Unternehmen: Konzepte – Maßnahmen – Methoden*. Hrsg. von G. Schreyögg. (eigene Paginierung gemäß PDF-Datei: S. 1-14). Berlin, S. 163–182.

Frank, U., Squazzoni, F. und Troitzsch, K. (2009). „EPOS-Epistemological Perspectives on Simulation: An Introduction“. In: *Epistemological Aspects of Computer Simulation in the Social Sciences. Second International Workshop EPOS 2006, 05.-06.10.2006 in Brescia, Revised Selected and Invited Papers*. Hrsg. von F. Squazzoni. Berlin, Heidelberg, S. 1–11.

Frank, U., Strecker, S., Fettke, P., Vom Brocke, J., Becker, J. und Sinz, E. (2014). „Das Forschungsfeld „Modellierung betrieblicher Informationssysteme““. In: *Wirtschaftsinformatik* 56.1, S. 49–54.

Frank, U., Winter, R., Mertens, P., König, W., Scheer, A.-W., Buhl, H., Buxmann, P., Legner, C. und Suhl, L. (2013). „„Impact Engineering“ or Social Responsibility? A Debate on the Responsibilities of Academics and Ways to Combine Scientific with Practice Impact“. In: *Business & Information Systems Engineering* 57.4, S. 279–292.

Freimann, J. (1994). „Das Theorie-Praxis-Dilemma der Betriebswirtschaftslehre – Wissenschaftssoziologische Überlegungen zu einem besonderen Verhältnis“. In: *Das Theorie-Praxis-Problem der Betriebswirtschaftslehre*. Hrsg. von W. Fischer-Winkelmann. Wiesbaden, S. 7–24.

Gethmann, C. (2004a). „Logik, epistemische“. In: *Enzyklopädie Philosophie und Wissenschaftstheorie*. Hrsg. von J. Mittelstraß. Bd. Band 2: H–O. Sonderausgabe Stuttgart 2004. Stuttgart, Weimar, S. 646–650.

Gethmann, C. (2004b). „Paradigma“. In: *Enzyklopädie Philosophie und Wissenschaftstheorie*. Hrsg. von J. Mittelstraß. Bd. Band 3: P–So. Sonderausgabe Stuttgart 2004. Stuttgart, Weimar, S. 33–37.

Gethmann, C. (2004c). „Rationalität“. In: *Enzyklopädie Philosophie und Wissenschaftstheorie*. Hrsg. von J. Mittelstraß. Bd. Band 3: P–So. Sonderausgabe Stuttgart 2004. Stuttgart, Weimar, S. 468–481.

Gulati, R. (2007). „Tent Poles, Tribalism, and Boundary Spanning: The Rigor- Relevance Debate in Management Research“. In: *Academy of Management Journal* 50.4, S. 775–782.

Habermas, J. (2020). „Erkenntnis und Interesse“. In: *Technik und Wissenschaft als ‹Ideologie›*. Hrsg. von J. Habermas. 22. Aufl. [Frankfurter Antrittsvorlesung vom 28.6.1965, zuerst publiziert in: Merkur, 19. Jg. (1965), Heft 213, S. 1139-1153.] Frankfurt, S. 146–168.

Hafner, M. ; Robin, E. und Hoorens, S. (2014). *The Cost of Non-Europe in the Single Market (‚Cecchini Revisited’). I – Free Movement of Goods*. Techn. Ber. Brussels: Study EPRS, European Parliamentary Research Service (PE 536.353), European Added Value Unit September 2014. URL: `https://www.europarl.europa.eu/EPRS/EPRS_STUDY_536353_CoNE_Single_Market_I.pdf`.

Hansmann, K.-W. (1974). *Entscheidungsmodelle zur Standortplanung der Industrieunternehmen.* Wiesbaden.

Hansmann, K.-W. (2006). *Industrielles Management.* 8. Aufl. München, Wien.

Heinrich, L. (2005). „Forschungsmethodik einer Integrationsdisziplin: Ein Beitrag zur Geschichte der Wirtschaftsinformatik". In: *NTM Zeitschrift für Geschichte der Wissenschaften, Technik und Medizin* 13.2, S. 104–117.

Heinrich, L. (2011). *Geschichte der Wirtschaftsinformatik – Entstehung und Entwicklung einer Wissenschaftsdisziplin.* (unter Mitarbeit von Ardelt, R.G.) Berlin, Heidelberg.

Hermes, H. (1991). *Einführung in die mathematische Logik – Klassische Prädikatenlogik.* 5. Aufl. Stuttgart.

Hodgkinson, G., Herriot, P. und Anderson, N. (2001). „Re-aligning Stakeholders in Management Research: Lessons from Industrial, Work and Organizational Psychology". In: *British Journal of Management* 12.S1. (Special Issue), S41–S48.

Hodgkinson, G. und Rousseau, D. (2009). „Bridging the Rigour–Relevance Gap in Management Research: It's Already Happening!" In: *Journal of Management Studies* 46.3, S. 534–546.

Hoffmann, D. (2013). *Die Gödel'schen Unvollständigkeitssätze – Eine geführte Reise durch Kurt Gödels historischen Beweis.* Berlin, Heidelberg.

Hohmann, S. (2022). *Logistik- und Supply Chain Management – Theorien und quantitative Aufgaben.* Wiesbaden.

Johansson, L.-G. (2016). *Philosophy of Science for Scientists.* Cham, Heidelberg, New York et al.

Kahneman, D. und Tversky, A. (1979). „Prospect Theory: An Analysis of Decision under Risk". In: *Econometrica* 47.2, S. 263–291.

Kappler, E. und Rehkugler, H. (1991). „Konstitutive Entscheidungen". In: *Industriebetriebslehre – Entscheidungen im Industriebetrieb.* Hrsg. von E. Heinen. 9. Aufl. Wiesbaden, S. 73–240.

Katz, I. (1974). „Local Convergence in Fermat's Problem". In: *Mathematical Programming* 6, S. 89–104.

Katz, I. und Vogl, S. (2010). „A Weiszfeld algorithm for the solution of an asymmetric extension of the generalized Fermat location problem". In: *Computers and Mathematics with Applications* 59.1, S. 399–410.

Khan, S., Zkik, K., Belhadi, A. und Kamble, S. (2021). „Evaluating barriers and solutions for social sustainability adoption in multi-tier supply chains". In: *International Journal of Production Research* 59.11, S. 3378–3397.

Kieser, A. und Leiner, L. (2009). „Why the Rigour–Relevance Gap in Management Research Is Unbridgeable". In: *Journal of Management Studies* 46.3, S. 516–533.

Kieser, A. und Nicolai, A. (2005). „Success Factor Research – Overcoming the Trade-Off Between Rigor and Relevance?" In: *Journal of Management Inquiry* 14.3, S. 275–279.

Kornmesser, S. und Büttemeyer, W. (2020). *Wissenschaftstheorie – Eine Einführung.* Berlin.

Kuckertz, A. (2012). „Evidence-based Management – Mittel zur Überbrückung der Kluft von akademischer Strenge und praktischer Relevanz?" In: *Schmalenbachs Zeitschrift für betriebswirtschaftliche Forschung* 64, S. 803–827.

Kuhn, H. (1973). „A Note on Fermat's Problem". In: *Mathematical Programming* 4, S. 98–107.

Kuhn, T. (2017). *Die Struktur wissenschaftlicher Revolutionen.* 25. Aufl. Frankfurt am Main.

Kühn, T. (2002). *Two Cultures, Universities and Intellectuals – Der englische Universitätsroman der 70er und 80er Jahre im Kontext des Hochschuldiskurses.* Tübingen.

Kunith, A., Mendelevitch, R. und Goehlich, D. (2016). *Electrification of a City Bus Network: An Optimization Model for Cost-Effective Placing of Charging Infrastructure and Battery Sizing of Fast Charging Electric Bus Systems.* DIW Discussion Papers 1577. Berlin: German Institute for Economic Research (DIW Berlin).

Lee, J. (2021). „The Legacy of Robert K. Merton: On Theories of the Middle Range". In: *Sociological Forum* 36.2, S. 515–519.

Lüder, K. (1990). „Standortwahl – Verfahren zur Planung betrieblicher Standorte und innerbetrieblicher Standorte". In: *Industriebetriebslehre – Handbuch für Studium und Praxis.* Hrsg. von H. Jacob. 4. Aufl. Wiesbaden, S. 25–100.

Macharzina, K. und Wolf, J. (2018). *Unternehmensführung: Das internationale Managementwissen, Konzepte – Methoden – Praxis.* 10. Aufl. Wiesbaden.

McCloskey, D. (1983). „The Rhetoric of Economics". In: *Journal of Economic Literature* 21, S. 481–517.

McCloskey, D. (1984). „The Literary Character of Economics". In: *Daedalus* 113.3, S. 97–119.

McCloskey, D. (1995). „Modern Epistemology Against Analytic Philosophy: A Reply to Mäki". In: *Journal of Economic Literature* 33, S. 1319–1323.

Melzer, D. (1979). „Das Steiner-Weber-Problem als eine Optimierungsaufgabe über der zulässigen Parametermenge einer konvexen Optimierungsaufgabe mit Parametern in den rechten Seiten der Restriktionen". In: *Anwendungen der linearen parametrischen Optimierung.* Hrsg. von K. Lommatzsch. Berlin, Lizenzausgabe Basel, Stuttgart 1979, S. 142–170.

Mertens, P. (2005). „Gefahren für die Wirtschaftsinformatik – Risikoanalyse eines Faches". In: *Wirtschaftsinformatik 2005 – eEconomy, eGovernment, eSociety.* Hrsg. von O. Ferstl, E. Sinz, S. Eckert und T. Isselhorst. Wiesbaden, S. 1733–1754.

Merton, R. (1968). *Social Theory and Social Structure.* 3. Aufl. New York.

Miehle, W. (1958). „Link-Length Minimization in Networks". In: *Operations Research* 6.2, S. 232–243.

Mordukhovich, B. und Nam, N. (2019). „The Fermat-Torricelli Problem and Weiszfeld's Algorithm in the Light of Convex Analysis". In: *Journal of Applied and Numerical Optimization* 1.3, S. 205–215.

Moulines, C. (2008). *Die Entwicklung der modernen Wissenschaftstheorie (1890-2000) – Eine historische Einführung.* Hamburg.

Nicolai, A. (2004). „Der „trade-off" zwischen „rigour" und „relevance" und seine Konsequenzen für die Managementwissenschaften". In: *Zeitschrift für Betriebswirtschaft* 74, S. 99–118.

Nicolai, A., Schulz, A.-C. und Göbel, M. (2011). „Between Sweet Harmony and a Clash of Cultures: Does a Joint Academic–Practitioner Review Reconcile Rigor and Relevance?" In: *The Journal of Applied Behavioral Science* 47.1, S. 53–75.

Olbrich, S., Frank, U., Gregor, S., Niederman, F. und Rowe, F. (2017). „On the Merits and Limits of Replication and Negation for IS Research". In: *AIS Transactions on Replication Research* 3. Paper 1, S. 1–19.

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A. und Sinz, E. (2010). „Memorandum zur gestaltungsorientierten Wirtschaftsinformatik". In: *Zeitschrift für betriebswirtschaftliche Forschung* 62.11. S. 664–672. Auch veröffentlicht in: Österle, H.; Winter, R.; Brenner, W. (Hrsg.): Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz. Nürnberg 2010, S. 1-11.

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A. und Sinz, E. (2011). „Memorandum on design-oriented information systems research“. In: *European Journal of Information Systems* 20.1, S. 7–10.

Popper, K. (2005). *Logik der Forschung*. 11. Aufl. Tübingen.

Reichertz, J. (2016). *Qualitative und interpretative Sozialforschung – Eine Einladung*. Wiesbaden.

Rescher, N. (1977). *Dialectics – A Controversy-Oriented Approach to the Theory of Knowledge*. Albany.

Rescher, N. und Brandom, R. (1980). *The Logic of Inconsistency – A Study in Non-Standard Possible-World Semantics and Ontology*. Oxford.

Römpp, G. (2018). *Philosophie der Wissenschaft – Eine Einführung*. Wien, Köln, Weimar.

Rorty, R. (2021). *Kontingenz, Ironie und Solidarität*. 13. Aufl. Frankfurt.

Rott, H. (2004). „Ramsey-Satz“. In: *Enzyklopädie Philosophie und Wissenschaftstheorie. Band 3: P–So*. Hrsg. von J. Mittelstraß. Sonderausgabe Stuttgart 2004. Stuttgart, Weimar, S. 460–461.

Sander, H. (2018). „Methodologische Adäquanz von Steuerwirkungstheorien“. Dissertation. Wiesbaden: Universität Duisburg-Essen.

Schmiel, U. (2016). „Evolutionary Analysis of Tax Law: A Methodological Approach“. In: *Modern Economy* 7.4, S. 377–390.

Schmiel, U. und Weitz, A. (2019). „Is a One-book-system Adequate? A Framework for Tax Law Analysis Under Genuine Uncertainty“. In: *Accounting, Economics, and Law: A Convivium* 9.2. Article 20160062, S. 1–18.

Schurz, G. (2011). *Einführung in die Wissenschaftstheorie*. 3. Aufl. Darmstadt.

Sneed, J. (1979). *The Logical Structure of Mathematical Physics*. 2. Aufl. Dordrecht, Boston, London.

Snow, C. (2012). *The Two Cultures and the Scientific Revolution*. 14. Druck (der 1. Aufl. 1959). Cambridge, New York, Melbourne.

Spies, M. (2004). *Einführung in die Logik – Werkzeuge für Wissensrepräsentation und Wissensmanagement*. Heidelberg, Berlin.

Stegmüller, W. (1987). *Hauptströmungen der Gegenwartsphilosophie – Eine kritische Einführung, Band II*. 8. Aufl. Stuttgart.

Steiner, M. (1993). „Konstitutive Entscheidungen“. In: *Vahlens Kompendium der Betriebswirtschaftslehre. Band 1*. Hrsg. von M. Bitz, K. Dellmann, M. Domsch und H. Egner. 3. Aufl. München, S. 115–169.

Straub, D. und Ang, S. (2008). „Readability and the Relevance versus Rigor Debate“. In: *MIS Quarterly* 32.4, S. III–XIII.

Teichert, D. und Rott, H. (2004). „Strukturalismus (philosophisch, wissenschaftstheoretisch)“. In: *Enzyklopädie Philosophie und Wissenschaftstheorie. Band 4: Sp–Z*. Hrsg. von J. Mittelstraß. Sonderausgabe Stuttgart 2004. Stuttgart, Weimar, S. 109–115.

Tversky, A. und Kahneman, D. (1991). „Loss Aversion in Riskless Choice: A Reference-Dependent Model“. In: *The Quarterly Journal of Economics* 106.4, S. 1039–1061.

Tversky, A. und Kahneman, D. (1992). „Advances in Prospect Theory: Cumulative Representation of Uncertainty“. In: *Journal of Risk and Uncertainty* 5, S. 297–323.

Weiszfeld, E. (1937). „Sur le point pour lequel la somme des distances de n points donnés est minimum“. In: *Tohoku Mathematical Journal – First Series* 43, S. 355–386.

Wiltsche, H. (2013). *Einführung in die Wissenschaftstheorie*. Göttingen.

Wittgenstein, L. (1996). „Tractatus logico-philosophicus“. In: *Wittgenstein – Ausgewählt und vorgestellt von Thomas H. Macho.* Hrsg. von T. Macho. S. 91–166. Siehe auch: a) die Original-Publikation: Logisch-Philosophische Abhandlung. In: Annalen der Naturphilosophie, 14. Jg. (1921), S. 185-262. b) Wiederabdruck: Tractatus Logico-Philosophicus. 6. Druck, London 1955 (S. 24-189). c) Wiederabdruck: Logisch-philosophische Abhandlung – Tractatus logico-philosophicus. Kritische Edition, hrsg. von McGuinness, B.; Schulte, J. Frankfurt 1989 (S. 1-179). München.

Wolf, J. (2020). *Organisation, Management, Unternehmensführung – Theorien, Praxisbeispiele und Kritik.* 6. Aufl. Wiesbaden.

Wolf, J. und Rosenberg, T. (2012). „How Individual Scholars Can Reduce the Rigor-Relevance Gap in Management Research“. In: *BuR – Business Research* 5.2, S. 178–196.

Ząbkowicz, J. (2015). „The single market and the “bicycle theory” of the European Union politics. Does it still work?“ In: *Ekonomia i Prawo – Economics and Law* 14.4, S. 503–516.

Zahn, E. und Schmid, U. (1996). *Produktionswirtschaft 1: Grundlagen und operatives Produktionsmanagement.* Stuttgart.

Zelewski, S. (1993). „Strukturalistische Produktionstheorie – Konstruktion und Analyse aus der Perspektive des „non statement view““. Geringfügig überarbeitete Fassung der Habilitationsschrift, Universität zu Köln, 1992. Wiesbaden.

Zelewski, S. (1999). „Strukturalistische Rekonstruktion einer theoretischen Begründung des Produktivitätsparadoxons der Informationstechnik“. In: *Wirtschaftsinformatik und Wissenschaftstheorie – Bestandsaufnahme und Perspektiven.* Hrsg. von J. Becker, W. König, R. Schütte, O. Wendt und S. Zelewski. Wiesbaden, S. 25–68.

Zelewski, S. (2004). „Epistemische Unterbestimmtheit ökonomischer Theorien – eine Analyse des konventionellen Theorienkonzepts aus der Perspektive des „non statement view““. In: *Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik – Theoriebildung und -bewertung, Ontologien, Wissensmanagement. Tagung der Kommission Wissenschaftstheorie des Verbands der Hochschullehrer für Betriebswirtschaft e.V., 05.-06.06.2003 in Koblenz.* Hrsg. von U. Frank. Wiesbaden, S. 1–30.

Zelewski, S. (2007). „Beurteilung betriebswirtschaftlichen Fortschritts – Ein metatheoretischer Ansatz auf Basis des „non statement view““. In: *Die Betriebswirtschaft* 67.4, S. 445–481.

Zelewski, S., Hohmann, S., Hügens, T. und unter Mitarbeit von Peters, M. (2008). *Produktionsplanungs- und -steuerungssysteme – Konzepte und exemplarische Implementierungen insbesondere mithilfe von SAP® R/3®*. München: Oldenbourg.

Chapter 2

Academic Publishing and Academic Ethos – Looking Back to the Future

Stefan Klein

Ulrich and I shared an office for five years at GMD's Cologne Research Center on the Information Economy (1987–1992),[1]which laid the foundation for a lifelong friendship. We both had studied management at the University of Cologne (1978–1983) at a time when seminar theses and punch cards for the COBOL course as part of the IS specialization were written with. We compiled collections of index cards for referencing and abstracting literature and collected folders with copied articles and books. A notorious examination question in the oral IS exam was 'how many punch cards fit into this room?'. The historic reflections are obviously subjective and stylized, yet they try to capture the changing sentiments over the past decades as well as persistent issues like the role and ethos of science that are revisited and rearticulated over time. The disciplinary focus is on management, IS and computer science, other disciplines have different research and publication cultures. I use Elsevier as an example to illustrate the impact and risks of digitalization of academic publishing house, or 'information analytics businesses' as Elsevier has rebranded itself.

2.1 Utopia (Late 1980ies–Mid 1990ies)

'Scientists and technologists, like all people who create, must dream, must put forth a vision of their future, or else they relegate their work to a kind of near-pointless incrementalism' (Feigenbaum 1989, p. 118).

2.1.1 Information Search and Access

While working as PostDocs at GMD, we were discussing visions of digital research infrastructures. One example for a customized bibliographic database including links to the digital files of articles and books, i. e., software that supports the capturing of bibliographic information of articles or books based on standardized identifiers (today primarily DOI or ISBN), the assignment of keywords, annotations, the inclusion of digital files, referencing within word processors and the creation of bibliographies, which for the user – over time – becomes a private library accessible with a notebook or tablet. Such reference managers

1 Forschungsstelle für Informationswirtschaft which integrated research streams and researchers from GID, the Research Center for Information and Documentation `https://www.deutsche-digitale-bibliothek.de/item/FGOQHVT44GNYQCABC6EBCZ6AXXYJ7TBQ`

function at the same time as research and authoring tools with a referencing app integrated in the word processor for in-line referencing and the automatic compilation of the list of references. What we imagined at the time, has become reality only a few years later: The first version of what is now Citavi was developed already in 1995, Mendeley's first version was released in 2008. Providing and managing digital bibliographic references – and soon after digital versions of articles and books – reflected the digital transformation of libraries at the time, which were complemented by thematically focused national digital archives, e. g., centralized technical information centers and document repositories ('Fachinformationszentren'). Many of those repositories used different search and retrieval languages and, thus, required specific training or the support of information brokers (Zelewski 1987).

2.1.2 Library of the Future

During the 1980ies, there was a first wave of research on AI, at the time referred to as expert systems. Feigenbaum has sketched a broad vision of the library of the future which provides novel forms of knowledge representation and knowledge access but also knowledge production:

> *'Now imagine the library as an active intelligent knowledge server. It stores knowledge of the disciplines in complex knowledge structures, perhaps in a knowledge representation formalism yet to be discovered or invented. It can reason with this knowledge to satisfy the needs of its users. These needs are expressed naturally with fluid discourse. The system can, of course, retrieve and exhibit. That is, it can act as an electronic textbook, but it can also collect relevant information, it can summarize, it can pursue relationships. It acts as a consultant on specific problems, offering advice on particular solutions, justifying those solutions with citations, or with a fabric of general reasoning. If the user can suggest a solution or a hypothesis, it can check it. It can even suggest extensions, or it can criticize the user's viewpoint with a detailed rationale of its agreement or disagreement. It pursues relational paths of associations, to suggest to the user previously unseen connections. Collaborating with the user, it uses its processes of association and analogizing to brainstorm for remote or novel concepts. With more autonomy, but with some guidance from the user, it uses criteria of "interestingness" to discover new concepts, new methods, new theories, new measurements.'* (Feigenbaum 1989, p. 122).

Feigenbaum's article demonstrates remarkable foresight and intriguing examples of algorithmic support of knowledge work, even though his anthropomorphic view of reasoning 'know-bots' seems untenable.

2.1.3 Open Science, Open Access and the Ethos of Science

In his foundational essay on the sociology of science, Merton (1942) has articulated four basic principles or norms which constitute the ethos of science, the ethical values guiding scientific work: universalism, communism (or communalism), disinterestedness, organized (or systematic) skepticism. Chubin (1985)'s essay 'Open Science and Closed Science: Tradeoffs in a Democracy', establishes the notion of open science based on Merton's norm of 'communality', i. e., 'the communal character of scientific knowledge ... open communication was an imperative for scientific integrity' (Chubin 1985, p. 73). Principles of open science are also reflected in Article 27 of the Universal Declaration of Human Rights adopted by the United Nations in 1948:

'1. Everyone has the right freely to participate in the cultural life of the community, to enjoy the arts and to share in scientific advancement and its benefits.'

'2. Everyone has the right to the protection of the moral and material interests resulting from any scientific, literary or artistic production of which he is the author.'

Science is seen as a subsystem of society with a distinctive logic as articulated by the Magna Charta Universitatum, a declaration and affirmation of the fundamental principles, was signed in 1988 on the occasion of the 900th anniversary of the University of Bologna and updated in 2020:

> *'... research and teaching must be intellectually and morally independent of all political influence and economic interests. ... To fulfil their potential, universities require a reliable social contract with civil society, one which supports pursuit of the highest possible quality of academic work, with full respect for institutional autonomy. As they create and disseminate knowledge, universities question dogmas and established doctrines and encourage critical thinking in all students and scholars. Academic freedom is their lifeblood; open enquiry and dialogue their nourishment. Universities embrace their duty to teach and undertake research ethically and with integrity, producing reliable, trustworthy and accessible results. Universities have a civic role and responsibility. They are part of global, collegial networks of scientific enquiry and scholarship, building on shared bodies of knowledge and contributing to their further development ...'* (Observatory Magna Charta Universitatum 2020).

2.2 Paradise Gained (Mid 1990ies–2010)

The 1990ies became a 'Gründerzeit' for IT driven innovation and provided large scale digitization, epitomized in 'Being Digital' by Negroponte (1996) who, at the time, was the director of MIT's Media Lab. The Internet (Campbell-Kelly and Garcia-Swartz 2013) had become a versatile, all-purpose technology, facilitating 'eEverything' and a 'smarter planet' (Lohr 2010); a hotbed for innovation and economic growth. In 1996, Barlow, one of the founders of the Electronic Frontier Foundation, formulated his 'Declaration of the Independence of Cyberspace.' The Declaration called for a self-regulated web combined with a rejection of all attempts at state control:

> *'Governments of the Industrial World, you weary giants of flesh and steel, I come from Cyberspace, the new home of Mind. On behalf of the future, I ask you of the past to leave us alone. ... We are creating a world that all may enter without privilege or prejudice accorded by race, economic power, military force, or station of birth. We are creating a world where anyone, anywhere may express his or her beliefs, no matter how singular, without fear of being coerced into silence or conformity'* (Barlow 1996).

Already in 1994, Dyson et al. (1996) had articulated a US-centric, utopian 'Magna Carta for the Knowledge Age', a new, libertarian social, economic and political order based on competition, individual freedom, redefining property rights in cyberspace with an emphasis on private ownership: *'To start with, Liberation –from Second Wave rules, regulations, taxes* *and laws laid in place to serve the smokestack barons and bureaucrats of the past. Next, of course, must come the creation of a new civilization, founded in the eternal truths of the American Idea'* (308). They understood the crucial role of antitrust law as key to *'prevent the acts and practices that can lead to the creation of new monopolies, or harm consumers ...'*, yet did not anticipate that only a few years later, tech giants had accumulated unprecedented market power, while at the same time claiming the libertarian ideas of overcoming rules, regulations, taxes and laws (308). *'This mixture of liberal and emancipatory visions of the web, neoliberal views of the market, and a strong technological determinism—comprising what then became known as the so-called Californian ideology—proved to be an extremely powerful narrative in the following decades. It was successful not least because it was able to bring together the world views of two quite different groups of actors: it fitted both the "freewheeling spirit of the hippies" and the "entrepreneurial zeal of the yuppies" (Barbrook and Cameron 1996: 45)'* (Dolata 2022, p. 458).

2.2.1 Promises: The Internet as Commons vs. the Dictate of Commercialization

Frischmann (2006) positions his paper by stating: 'the open access (commons) versus private control debate is raging'. At that point, the Internet had become a public and social infrastructure which has generated substantial social and economic value. Already in 1998, the Google founders published a paper that recognized the risks of advertisement driven search engines (and more broadly advertisement-driven business models): 'The anatomy of a large scale hypertextual Web search engine', in which they make a strong case for a search engine, one of the key infrastructure components of the Internet, crucial to navigate and access the growing body of information, that is not driven by advertising and biased towards advertisers, rather 'transparent and in the academic realm' (Brin and Page 1998):

> *'Currently, the predominant business model for commercial search engines is advertising. The goals of the advertising business model do not always correspond to providing quality search to users. [...], we expect that advertising funded search engines will be inherently biased towards the advertisers and away from the needs of the consumers. Since it is very difficult even for experts to evaluate search engines, search engine bias is particularly insidious. [...] But we believe the issue of advertising causes enough mixed incentives that it is crucial to have a competitive search engine that is transparent and in the academic realm'* (Brin and Page 1998, p. 18).

Also, their early motto, 'don't be evil', can be read as an awareness of the profound risks this technology, with unprecedented network externalities, could cause.

2.2.2 Digitalization of Research

At the time, digitization of academic articles and the subsequent digitalization of publishing, including the academic workflow, had and have profoundly changed academic work:

- Commercial and open access academic content platforms (e. g., Elsevier launched ScienceDirect.com in 1997, ResearchGate was launched in 2008) enabled search, recommendations, retrieval and immediate access to publications in digital format.
- Reference management tools became *'all-in-one reference and note-taking solution ... for individuals and teams'* (`https://www.citavi.com/en`) simplifying research and publications.

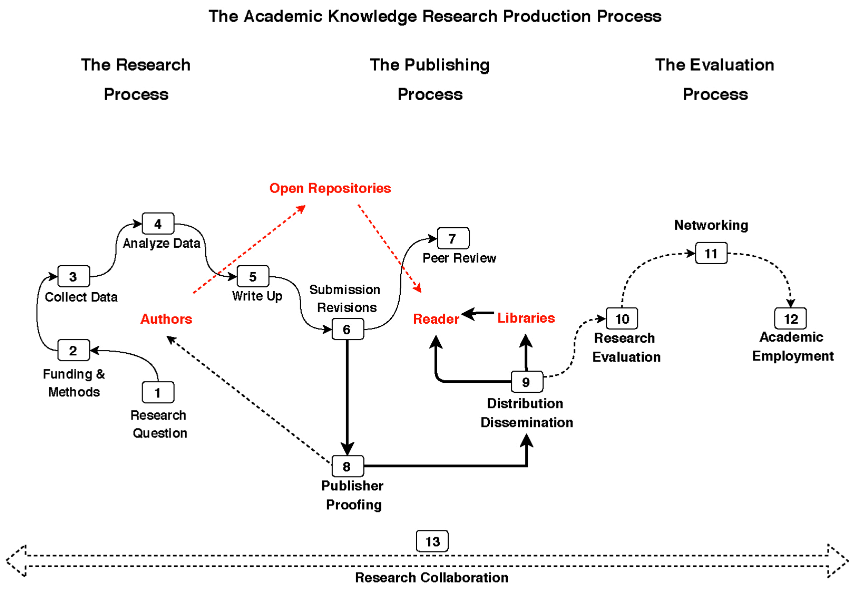


Figure 2.1: The Academic Knowledge Research & Production Process
(Source: `https://books.openedition.org/oep/docannexe/image/9068/img-4.jpg`)

- Datafication led to evidenced based, multi-metrics ranking systems of journals, articles and researchers.

Digitalization overall has increased academic productivity, with respect to ease and speed of search, access, management and analysis of content. It has also lowered the cost of publishing and distribution, authors are expected to submit digital, reproducible documents, distribution cost of digital articles or books is negligible. The reduction of publication costs, the non-rivalrous use of digital articles and the benefits of generating value through the production of further public goods (data, information, knowledge), suggests *'recognizing that basic research behaves economically as infrastructure – in the sense that it creates social value primarily when used as an input into the production of a wide variety of public good outputs – suggests that the social costs of restricting access to the resource can be significant'* (Frischmann 2006, p. 995).

2.2.3 Early Stages of Open Access

In 2001, the Budapest Open Access initiative extended the idea of open communication *'to make research articles in all academic fields freely available on the internet'* (Budapest Open Access Initiative – Make research publicly available). Extending the Budapest initiative, the Berlin Declaration in 2003 embraced the opportunities of the Internet and defines criteria of openness:

> *'[f]or the first time ever, the Internet now offers the chance to constitute a global and interactive representation of human knowledge, including cultural heritage and the guarantee of worldwide access. ... In order to realize the vision of a global and accessible representation of knowledge, the future Web has to*

> *be sustainable, interactive, and transparent. Content and software tools must be openly accessible and compatible'* (Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities, `https://doaj.org/`, 2003).

During this period, a number of open-access journals were founded (most are listed in the Directory of Open Access Journals – DOAJ), prominently PLOS (in 2000) in order to make articles available to all and for free, and the Creative Commons family of license models for open sharing (`https://creativecommons.org/20-years/`) was established (see also Lessig 2005).

2.3 Paradise Lost (Since 2010)

While long in the making, the full force of the transformation of publishing from a paper-based and library-based system towards digital platforms and research tools (digitization and digitalization) became visible since about 2010. As of 2020, digital publications accounted for 89 % of the global scholarly publication market (STM Association 2021, p. 4).

Digitization has led to the development of digital mega-bundles of academic content, mostly owned by commercial publishers. Digitization has profoundly changed the roles of libraries, who have become sidelined, disintermediated, and relegated to an administrative role of managing and executing the digital subscription contracts and access infrastructures.

While digitalization of publishing could have benefitted a more diverse group of publishing houses and also could have caused digital disruption of the incumbents, e. g., by the rise of open access and open science initiatives, the biggest academic publishers successfully avoided disruptive innovation (Ponte and Klein 2017). The outcome has been an increasing concentration and monopolization of the once fragmented academic publishing market, with a 48 % market share of the three largest publishers: Elsevier, SpringerNature and Wiley (Kim and Park 2020).

2.3.1 Access, Search and Surveillance

For many disciplines, relevant sources are distributed across a growing number of platforms, some open access, some subscription or pay-per-use, many mixed. The German Database Information System, a collaborative project co-funded by DFG currently lists 14.514 databases (`https://dbis.ur.de`).

As libraries play a more limited role in the retrieval of articles, most users would not be aware of the coverage and limitations of different platforms. ScienceDirect is often perceived as a neutral platform like the name suggests, even though it is the Elsevier platform. In case one's institution does not subscribe to the required databases, users search for accessible copies or versions of the article across grey sites, pre-publishing servers or self-archived versions on ResearchGate.

Unpaywall.org does not only have a programmatic name, but it is a useful tool to locate files of specific articles without access restrictions. These anecdotal observations suggest that even though access to literature has become much easier, publishers have created 'walled gardens' (Plantin et al. 2018, p. 303) which in turn has increased search cost. Few individual users pay the download fees for papers which can easily exceed 30 EUR.

Thus, the availability of free versions of an article determines, whether articles are actually read, thereby creating additional incentives for authors and their institutions to make papers accessible for free.
The platformatization of academic literature has shaped the retrieval and use of literature:

- Cross-platform search has become a skill to cover a fairly diverse ecosystem of platforms and sites.
- Search for articles typically focuses on a relatively small number of platforms. In particular students, doctoral candidates and junior faculty increasingly tend to conflate journal rankings, quality of individual articles and relevance of the articles and focus on highly ranked journals in their searches.
- Literature search is *often misunderstood* as putting a number of search terms into a query, which typically yields too many hits, which then are reduced to what is deemed a manageable number by applying simplistic (and dubious) criteria.

A second, less visible impact is that publishers have used the opportunity to monitor the behavior of the platform users. Just like Amazon, Google and Meta surveil the identities of their users, the clickstreams, the response to triggers and nudges, publishers can see the identities of those users who log-in in order to get access to articles, they provide ads to the users in form of recommendations of papers on their platform and keep lists of prior searches as a service to the readers, who might have forgotten what they looked for. The publishers' mostly opaque monitoring, tracking, and surveillance activities have become excessive and threatening the privacy of researchers as a report published by Deutsche Forschungsgemeinschaft has demonstrated and criticized (Deutsche Forschungsgemeinschaft (DFG) 2021)). The use of end-user tracking tools (e. g., ThreatMetrix by LexisNexis which is part of Elsevier's holding company RELX) and the fingerprinting of every PDF downloaded (Franceschi-Bicchierai 2022) – mostly unknown and opaque to the users – is justified by security and fraud protection or to *'... offer you customized content and other personalization to make the Service more efficient for you and more relevant to your interests and geography'* (Elsevier 2022a). Yet, the extent of user data collected, provides opportunities for de-anonymization and unauthorized use and re-use of user data (Carpenter 2020). Overall, Elsevier has acquired an unprecedented collection of data and behavioral insights, benefitting from and systematically exploiting data network effects (Gregory et al. 2022). While claiming to be transparent and acting in the users' interest, the privacy policy of Elsevier, e. g.,, is 3574 words long (about 11 pages) and details extensive, yet vague policies in terms of collecting, sharing, disclosing and retain user information:

> *'We **collect** information about you in three ways: directly from your input, from third-party sources, and through automated technologies. ...The Service may automatically **collect** information about how you and your device interact with the Service, ...We may also **share** your personal information with our group companies and with sponsors, joint venture partners and other third parties ...We also will **disclose** your personal information if we have a good faith belief that such disclosure is necessary to ...an acquisition by or merger with another company. ...We **retain** your personal information for as long as necessary to provide the Service and fulfill the transactions you have requested, or for other essential purposes ...Your personal information may be **stored and processed** in your region or another country ...(*`https://www.elsevier.com/legal/privacy-policy`*) [bold face added by the author]'*

Even if they do not actively intend to compromise their users' privacy, the extent of user behavioral monitoring, tracking and surveillance creates an unprecedented data trove controlled by numerous commercial companies that most users are not even aware of and that poses substantial security and privacy risks.

A third issue is the lack of transparency and independent control over the search results. Elsevier, which prioritizes its own revenue, controls the search and recommendation/advertising algorithms on ScienceDirect. To paraphrase Brin and Page (Brin and Page 1998, p. 18): Since it is very difficult even for experts to evaluate literature search engines, search engine bias is particularly insidious. We believe that journal competition and rankings cause enough mixed incentives that it is crucial to have an independent search engine that is transparent and in the academic realm.

2.3.2 Commercialization of Academic Research Infrastructures

The German Science and Humanities Council has provided an account of the transformation of academic publishing over the past 50 years, at the end of which an oligopoly of commercial academic publishers has emerged (Lariviere et al. 2015; Puehringer et al. 2021):

> *'While before World War II most scientific journals were still published by professional societies, in the mid-1990s, the share of commercial providers in the US already lay at 40 %. The sale of journal subscriptions to academic libraries in particular has enabled and continues to enable publishers to generate high and secure revenues ... A study examining the share of academic publications in major publishers' journals and its development between 1973 and 2013 shows that the natural and medical sciences as well as the social sciences and humanities were able to continuously increase their share. The highest degree of concentration occurred in the social sciences (with 70 % of articles with the top five publishers). Over the course of digitalisation, the share of the largest publishers has continued to rise globally, due both to the successful establishment of new journals and the takeover of existing journals by these publishers. Currently, more than 50 % of the total German academic publication volume is published in journals belonging to the three largest academic publishing houses: Springer Nature, Elsevier and Wiley.'* (Wissenschaftsrat 2022, p. 18)

The Problem

Despite promising and successful initiatives such as the Public Library of Science, arxiv.org or PLOS One, open-access publishing has not succeeded to disrupt the incumbent academic publishers (Monbiot 2011; Ponte and Klein 2017). Buranyi (2017) recounts the events in 2011, when there was substantial resistance against the substantial price increases of the largest academic publisher, which suggested that *'the tide was about to turn against the industry that Elsevier led'*. Yet, a year later, most libraries had signed the new contracts as a systemic and regulatory shift towards open access had not happened:

> *'In 2011, Claudio Aspesi, a senior investment analyst at Bernstein Research in London, made a bet that the dominant firm in one of the most lucrative industries in the world was headed for a crash. ... Aspesi, after talking to a network of more than 25 prominent scientists and activists, had come to believe the tide was about to turn against the industry that Elsevier led. More and more research libraries,*

> *which purchase journals for universities, were claiming that their budgets were exhausted by decades of price increases, and were threatening to cancel their multi-million-pound subscription packages unless Elsevier dropped its prices. State organisations such as the American NIH and the German Research Foundation (DFG) had recently committed to making their research available through free online journals, and Aspesi believed that governments might step in and ensure that all publicly funded research would be available for free, to anyone. Elsevier and its competitors would be caught in a perfect storm, with their customers revolting from below, and government regulation looming above. In March 2011, Aspesi published a report recommending that his clients sell Elsevier stock. A few months later, in a conference call between Elsevier management and investment firms, he pressed the CEO of Elsevier, Erik Engstrom, about the deteriorating relationship with the libraries. He asked what was wrong with the business if "your customers are so desperate". Engstrom dodged the question. Over the next two weeks, Elsevier stock tumbled by more than 20 %, losing GBP 1bn in value. The problems Aspesi saw were deep and structural, and he believed they would play out over the next half-decade—but things already seemed to be moving in the direction he had predicted. Over the next year, however, most libraries backed down and committed to Elsevier's contracts, and governments largely failed to push an alternative model for disseminating research. In 2012 and 2013, Elsevier posted profit margins of more than 40 %'* (Buranyi 2017).

Instead of sharing the reduced production cost of digital content, the large publishers used the transformation from journal subscription to the subscription of digital bundles of academic content to increase the prices. They justified the price increases by a growing number of articles as a result of ongoing publishing activities, acquisitions of journals and mergers[2]. The different modes of value extraction (Mazzucato 2018) extend the traditional publishing model, whereby the publisher acquires the article's copyright (for free) in return for its publication, by offering authors the retention of their copyright in exchange to article processing charges (APC, on average 2500 Euro per article) or publish and read fees (PAR) – euphemistically called 'Gold Open Access'. At the same time, the publishers rely on massive voluntary 'donations' of reviewers and editors:

> *'... the number of academic publications worldwide increased nearly fivefold from the 1980s to 2020, whereas the number of university researchers merely tripled. ... According to one historical review, the emergence of professional academics and the perception of peer review as a "voluntary duty" allowed commercial publishers to legitimize themselves and dominate the publishing market – "all while turning a profit. One conservative estimate put the cost of time "donated" to peer review for U.S. reviewers at USD 1.5 billion in 2020'* (Aczel et al. 2021; Goodman 2022).

If we apply the estimates to Elsevier, in 2021 reviewers 'donated' 186 million USD and research institutions 'donated' authors' time to write 1,95 million articles and subsequently revise many of those articles.

In 2020, the Dutch universities signed a new contract with Elsevier, whereby 'the institutions collectively pay 16,4 million Euro per year (from 2021 onwards), compared to 12,2 million Euro (reference point 2018) per year under the previous contract. With roughly 6000 articles published by researchers at Dutch institutes in 2019, this boils down

2 `http://www.nature.com/news/nature-owner-merges-with-publishing-giant-1.16731`

to an estimated 2700 Euro per article. ... both sides agreed to work towards the creation of infrastructure for research data and information, and to enter into "open science" projects' (Knecht 2020). Which suggests that Elsevier attempts to extend its revenue base towards service contracts and collaborative projects with universities, further legitimizing its role as research partner.

Platformatization

Platformatization has furthered datafication and algorithmization of content retrieval, recommendations, and marketing but also of rankings and evaluation of research. The transformation of the academic publishing market also led to an increasing vertical diversification of the leading publishing houses, accelerated and enabled by mergers and acquisitions. The pattern is familiar from the technology giants: evaluation and ranking services as well as research support services and a vertical integration into pre-print platforms are prominent areas of diversification, conveniently ignoring conflicts of interest between the roles of publisher and ranking organization.

> *'... the coordination, control and exploitation mechanisms characteristic of the platform architectures are characterized by a strong hierarchical orientation in which elements of co-optation and the orchestrated participation of users are embedded. In this hybrid constellation, the platform companies have a* ***high degree of structure-giving, rule-setting and controlling power—in addition to exclusive access to the raw data material generated there****. While this power may manifest, at times, as rigid control, direct coercion or enforceable accountability, for the majority of rule-obeying users it unfolds nearly imperceptibly and largely silently beneath the surface of a (supposed) openness that likewise characterizes the platforms as technically mediated spaces for social and economic exchange'* (Dolata and Schrape 2022) [bold face added by the author].
>
> *'... the major Internet platforms such as Facebook, Instagram and YouTube organize, observe and regulate significant parts of social exchange on the Internet today—via self-created social rules that can be read in their terms and conditions and community standards, and whose implementation is primarily algorithmic. The private-sector operators of the major platforms* ***have thus taken over essential social structuring and regulation functions on the Internet and have created a parallel social world that has so far neither been democratically legitimized nor democratically controlled'*** (Dolata 2021, p. 100) [bold face added by the author].

Platformatization has led to the development of complex infrastructures for large scale data collection and aggregation, which provide novel opportunities for diverse analytics operations on accumulated data, content and user profiles both for research as well as monetization purposes. The diversification has changed the role and identity of the publishing houses, which see themselves as research analytics companies (RAC). As such their relation with the academic community is changing, as the RACs are accumulating academic content, which provide unprecedented opportunities for research. Many RACs engage in research partnership with academics and develop their own research operations targeted at commercial customers.

Appropriation of Research Data and Content Usage Right

Nachtwey and Schaupp (2022) have analyzed the data driven value generation of digital platforms as a three step process of data commodification or assetization, which is subject to the dynamics of data network effects (Cusumano 2022):

> *'anchored in the platforms' terms and conditions, which stipulate that users share their data not only with each other but also with the company. This, in turn, enables a three-step process of commodification of data on the part of the platform. First, the data are appropriated as use values by the platforms to optimize their own services. Subsequently, a secondary commodification takes place, which, unlike other value creation processes, is decoupled from the use value of the data. Third, there is a process of cybernetization, in which the ability to influence users is sold to third parties as a commodity'* (Nachtwey and Schaupp 2022, p. 59).

The example of Elsevier is a fitting illustration of this process. RELX, Elsevier's parent company *'is a global provider of information-based analytics and decision tools for professional and business customers, enabling them to make better decisions, get better results and be more productive. ... We do this by leveraging a deep understanding of our customers to create innovative solutions which combine content and data with analytics and technology on global platforms ... leverage our skills, assets and resources across RELX, both to build solutions for our customers and to pursue cost efficiencies.'* (RELX 2021, p. 2), which suggests a clear focus on extending data collection, data acquisition, as well as sharing data and tools across RELX, with a clear emphasis on improving returns, beyond the already 38 % profit margin for Elsevier (RELX 2021, p. 23).[3] Enabled by the digitization of publishing and the development of the research infrastructure, Elsevier has enacted usage policies (terms and conditions, 3079 words) across their different business segments, which stipulate that they acquire the usage rights for all data and content a user is posting on any of the digital services, including submitted articles, reviews or any other communication: *'by posting, uploading, inputting, providing or submitting ("Posting") a Submission, ..., you grant us (and those we work with) a royalty-free, perpetual, irrevocable, worldwide, non-exclusive right and license to publish, post, reformat, index, archive, make available, link to, and otherwise use such Submission in all forms and media (whether now known or later developed), in connection with the operation of our respective businesses (including, without limitation, the Services) and to permit others to do so'* (Elsevier 2022b, p. 2). This is worse than a Faustian bargain, as most user would not even be aware of the data appropriation, nor does Elsevier explain the purpose of the data collection.

Analytics Operations, Tooling and Research Workflow

Elsevier has used the wealth of accumulated data, leveraging positive data network externalities (Gregory et al. 2022; Mitomo 2017) for the development of analytics solutions

3 *'Underlying revenue growth was +3 %, driven by continued good growth in electronic revenue, which represents 88 % of divisional revenue. ... In Primary Research growth was driven by broader content sets, increasing sophistication of analytics, and evolving technology platforms. Article submissions remained at last year's elevated levels. The number of articles published grew strongly, with continued growth in subscription articles and particularly strong growth in open access articles, leading to further market share gains in both payment models. In Databases & Tools and Electronic Reference, representing over a third of divisional revenue, strong growth was driven by content development and enhanced machine learning and natural language processing-based functionality'* (RELX 2021, p. 23).

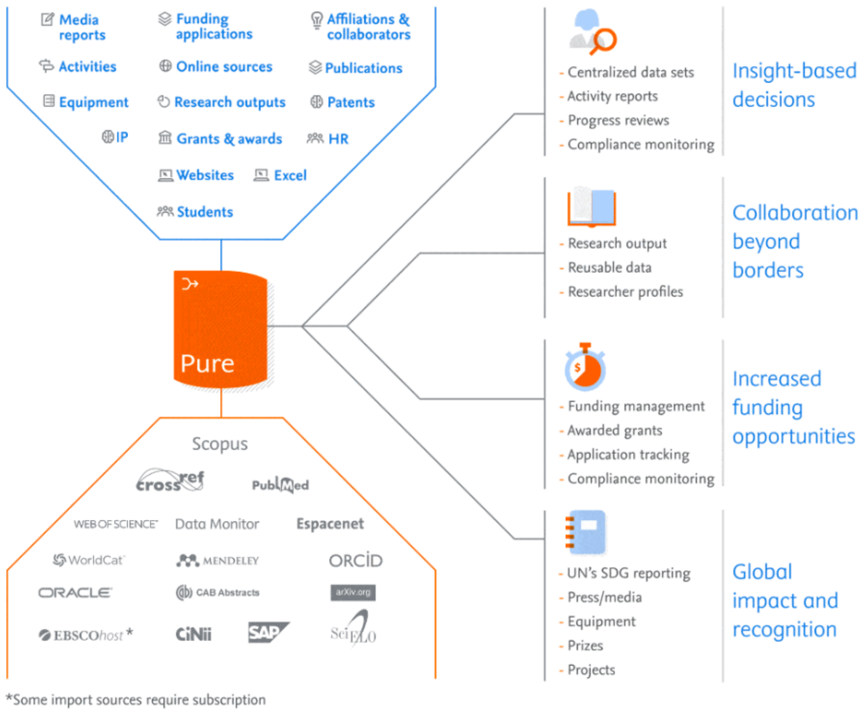


Figure 2.2: Pure data model (https://beta.elsevier.com/products/pure/how-it-works), see also Garcia Morgado (2020).

across all its business segments: content and research analytics, researcher and research market analytics. Figure 2.2 depicts the data model, sources and insights provided by Pure, Elsevier's research information system for universities. It documents how data are used across Elsevier's solutions for quite different purposes and notably the crucial role of Scopus. Elsevier has extended (https://www.elsevier.com/rd-solutions) its methodological repertoire covering machine learning and deep learning models, text analysis and extraction, ontologies, vocabularies and thesauri, knowledge graphs, semantic enrichment and annotation. Most of the underlying algorithms are proprietary and regarded as trade secret. Four examples of analytics applications give an indication about the profound impact analytics will have on research:

- Machine learning for article recommendation (Haak 2014) and predictive models in medicine: *'Over 1B scientific articles added by 2.5M researchers globally are analyzed using user-based collaborative filtering to generate over 250M article recommendations. Over 1000 predictive models have been trained on 1.5 billion electronic health care records using machine learning'*[4]
- Automated literature reviews through text and data mining[5]

4 https://freshservice.com/resources/case-study/elsevier
5 https://researchcollaborations.elsevier.com/en/projects/automated-literature-reviews-through-text-and-data-mining-meta-da

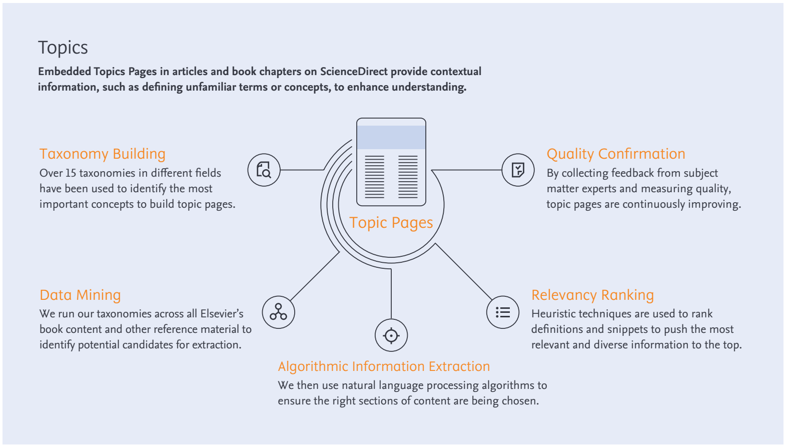

Topics
Embedded Topics Pages in articles and book chapters on ScienceDirect provide contextual information, such as defining unfamiliar terms or concepts, to enhance understanding.
Taxonomy Building
Over 15 taxonomies in different fields have been used to identify the most important concepts to build topic pages.
Topic Pages
Quality Confirmation
By collecting feedback from subject matter experts and measuring quality, topic pages are continuously improving.
Data Mining
We run our taxonomies across all Elsevier's book content and other reference material to identify potential candidates for extraction.
Relevancy Ranking
Heuristic techniques are used to rank definitions and snippets to push the most relevant and diverse information to the top.
Algorithmic Information Extraction
We then use natural language processing algorithms to ensure the right sections of content are being chosen.

Figure 2.3: *Topic Pages* as approach to manage and establish knowledge

- Topics pages[6] (see Figure 2.3) are algorithmically generated and represent foundational knowledge, definitions and links
- The potential for machine generated content syndication and curation has been illustrated by SpringerNature's machine-generated summary on lithium-ion batteries (Writer 2019) or (mostly) automatically translated content (e. g., Duwe 2022).

Elsevier has rebranded and calls itself as information analytics business. It functions as comprehensive academic infrastructure provider (content, data, tools, algorithms) and gatekeeper of academic (individual, group, institutional) reputation. It has systematically pursued a strategy of digitalization, expansion and diversification of services and customer segments. The strategy includes horizontal (acquisition of journals) and more importantly vertical integration, creating a research infrastructure. Elsevier has acquired 29 companies (`https://mergr.com/elsevier-bv-acquisitions#cma-tab`), in particular specialized service providers, which has enabled it to offer over digital tools and services. These services together represent an extensive, increasingly integrated research infrastructure, geared to increase research productivity for the users and revenue and data for Elsevier. Like Facebook, Elsevier has integrated user profiles from companies they acquired, notably SSRN and Mendeley. Figure 2.4 depicts the research, publishing, and evaluation process with the assigned Elsevier tools, platforms and services. The tools and services cover academic productivity tools, such as ScienceDirect or Mendeley, ranking tools, such as PlumX or CiteScore, as well as research (meta) information systems (RIS) such as Pure or SciVal.

The promise for productivity enhancements – in particular in a setting of academic capitalism (Münch 2014) – for individuals as well as for academic institutions amounts to a Faustian bargain with Elsevier (and comparable infrastructure providers): Is it worth in return for outsourcing critical academic infrastructures (Schonfeld 2018), to become increasingly dependent on a company, which surveils and exploits academics and so obviously prioritizes profit over the respect of academic values? Outsourcing critical infrastructures will over time lead to a hollowing of core academic competencies and will create path dependencies with respect to data collections and algorithmically-based data network effects.

6 `https://www.elsevier.com/solutions/sciencedirect/topics`

2.4 The Undermined Ethos of Science

The ethos of science has been undermined from within **academia**. Rankings as metric of academic 'performance' are increasingly replacing the search for knowledge and the 'extension of certified knowledge' (Merton 1942, p. 2), thus not only taking a problematic metric (journal rank) as proxy for academic quality (Münch 2014, for a detailled analysis see Münch 2022), but reinforcing the influence of commercial ranking organizations, including Elsevier with Scopus which is used in a number of rankings such as SciMago but also in university rankings. In as much as rankings have become the prime metric of academic excellence, the influence and power of publishers who own highly ranked journals has increased.

> *'in particular business school academics ... have progressively surrendered their autonomy and complied with the demands of managerialism. We argue this has been reinforced through coercive forms of power like rewards and punishment and bureaucratization; manipulation and mainstreaming through pushing a particular version of research to the top of the agenda; domination through shaping norms and values; and subjectification through creating new identities'* (Alvesson and Spicer 2016, p. 29).

Publishers are undermining and betraying the ethos of science by designing platforms as 'walled gardens' (Plantin et al. 2018, p. 303) and as substitutes for libraries, pursuing strategies similar to what Feigenbaum (1989, p. 122) had sketched but as commercial entities prioritizing economic returns over scientific ethos, while presenting themselves as the defenders of academic integrity. They engage in their own research activities but without the transparency and the checks and balances.

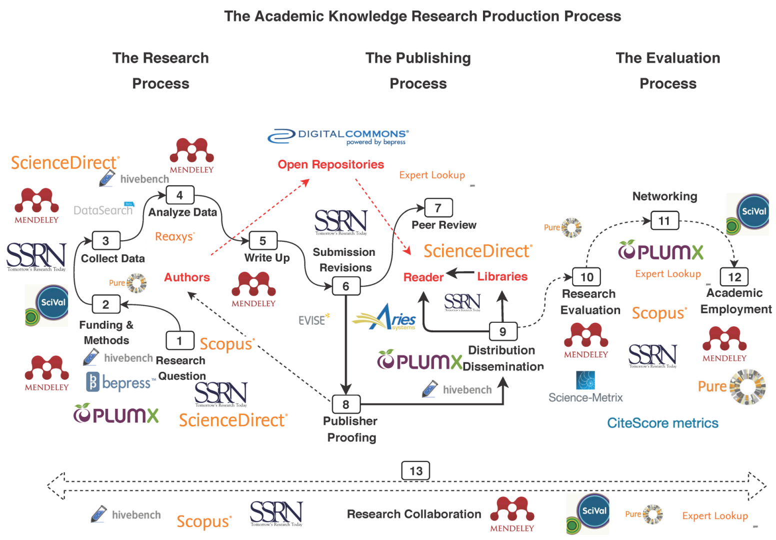


Figure 2.4: Elsevier's tools and the academic research process (Chen et al. 2019, p. 20) `https://books.openedition.org/oep/docannexe/image/9068/img-5.jpg`, for a comparison across different vendors see Bosman and Kramer (2016)

Like Jekyll and Hyde, they like to present themselves as benign company selflessly supporting research, when it suits them, and as a commercial company with obligation towards their shareholders, charging monopolistic prices for their services, exploiting millions of academics as authors, reviewers, data providers, while selling consulting services based on these data: *'What data are you missing? Draw from the largest corpus of trusted scientific data in the world, extracted from a variety of sources, including peer-reviewed, full-text literature. This data is used as vital inputs in training algorithms, knowledge graphs and other applied analytics'* (`https://www.elsevier.com/rd-solutions`). No one should deceive themselves: Elsevier is an information analytics business, relentlessly exploiting the academic system, overcharging for their journals (Grossmann and Brembs 2021) yielding an operating profit margin of 38 % in 2021 (RELX 2021, p. 23), appropriating information and data usage rights in an opaque manner and engaging in extensive surveillance of researchers' work patterns. In doing so, they have lost all credibility as a trustworthy partner of academics and academic institutions, exacerbated by communicating strategically, claiming best intentions to support open science and to benefit society and trying to justify their behavior, a behavior in line with what has been described as framing contests (Kaplan 2008) and bullshitting (Spicer 2020). They make unwarranted hyperbolic claims such as *'Pirate sites like Sci-Hub threaten the integrity of the scientific record, and the safety of university and personal data,'* the publishers behind the case in India told Nature in a statement (Else 2021).

Governments all over the world have engaged in attacking academic freedom and politicizing inconvenient academic findings. Trump and his administration has been a particularly visible example of disrespect of academic findings, e. g., regarding COVID-19 and actively destroying environmental data (Waldman 2018). Yet, this period has also been characterized by a few cases of **resistance and activism**.

- **Aaron Swartz** downloaded the JSTOR library in 2010 and committed suicide in 2013 after he was found guilty in court – even though JSTOR had dropped its charges – and faced a long prison sentence (`http://www.newyorker.com/magazine/2013/03/11/requiem-for-a-dream`).
- **Sci-Hub** is a shadow library hosted at different mirror sites, which provides free access to copyrighted papers protected by paywalls, was founded in 2011 by Alexandra Elbakyan. Legal action by leading publishers as copyright owners has not yet succeeded to block access. Another lawsuit at Delhi's high court has not been decided at the writing this chapter. Legal experts see, however, see a chance for Sci-Hub to prevail as India's copyright law has a broader concept of 'fair dealings' or fair use which has been used in the past to grant the right to lawfully copyrighted material for use in education (Else 2021).
- A number of countries (e. g., Germany with its Project DEAL (`https://www.projekt-deal.de/`) or networks of universities (University of California[7]) **cancelled** their Elsevier **subscriptions**, after negotiations of a path towards open access had failed.
- Social networking sites such as ResearchGate.net or open repositories such as Academia.edu have become de facto self-archiving and legitimate as well as grey **sharing platforms** for academics. Next to pre-print or post-print versions which authors are allowed to share, copyrighted versions are also made available, which has prompted publishers to submit take-down notes to authors and platforms.

7 `https://www.universityofcalifornia.edu/press-room/uc-terminates-subscriptions-worlds-largest-scientific-publisher-push-open-access-publicly`

2.5 Looking Ahead

The challenges going forward are formidable as tooling, algorithmization and infrastructuring of research are progressing rapidly and transforming research profoundly. Not only do the costs of developing and maintaining large scale generative AI solutions – as a prominent example of algorithmic tools – exceed the budgets of academic institutions, the impact of algorithmic support for literature reviews, paper reviews, writing of text, etc., will be fairly difficult to assess. To identify biases, omissions or mistakes will be beyond the abilities of most academics. A leading question for the future is therefore, whether and how we as academics will be able to regain and retain some of the benefits of IT facilitated research, while foregoing Faustian bargains with commercial companies, which have discredited themselves time and again by their disrespect for the work of academics and academic values. All the more are enforceable governance mechanisms and structures required to foster adherence to academic values, and fairness, accountability and transparency (see the ACM Conference on Fairness, Accountability, and Transparency series, ACM FAccT, see also Floridi 2019) as principles for the design and use of algorithmic tools and systems.

2.5.1 Access and Search

Digitalization has conflated the boundary between access and search and led to the development of new tools that are transforming how individuals can engage with (academic) content. For example, OpenKnowledgeMap[8] is a tool that structures knowledge domains and fields and visualizes streams of literature as a map. Generative AI tools, such as you.com, ChatGPT, GPT-4, elicit.com, and silvi.ai, are algorithmic research tools which generate responses such as structured summaries of the literature and some references, in response to specific questions. Metaphorically, they function as a powerful combination of search engine and individualized, algorithmically generated Wikipedia pages. Using such tools can help to accelerate research. However, given that these tools operate on large language models and extensive human training (including ‘prompt engineering’),[9] they have inherent limitations, in particular if they are viewed and assessed in categories of human intelligence, i. e., critical reasoning and explanations (Chomsky et al. 2023).

In his paper on the constitution of knowledge, Rauch (2018) highlights the critical role of the *‘epistemic honor code ...checks and balances (peer review and replication), separation of powers (specialization), governing institutions (scientific societies and professional bodies), voting (citations and confirmations), and civic virtues (submit your beliefs for checking if you want to be taken seriously)’*. These are threatened and undermined by social media trolling, *‘spreading lies and disinformation on an industrial scale’*. Trolls *‘... violate all those norms: They mock truth, sling mud, trash credentials, ridicule testing, and all the rest. ... they spread confusion and demolish trust.’* He sees the combination of social media as distribution platform, the psychological design of the software, which *‘humans are not designed to encounter or resist’* and, thirdly, the profitability of disinformation as key drivers of a *‘perfect storm of technological, economic, and political changes.’* While the technologically-enabled proliferation of misinformation may have been difficult to foresee, we are now in a better situation to recognize the dangers of commercial research infrastructures along similar lines: research infrastructures have become global search, access, and distribution platforms, they

8 `https://openknowledgemaps.org`
9 `https://www.businessinsider.com/prompt-engineering-ai-chatgpt-jobs-explained-2023-3`

increasingly shape both the knowledge production and the academic checks and balances (ranking, performance evaluation and reviewing), they have become institutionally and culturally widely accepted, and they have been developed into highly profitable business models. While the big publishing houses pay lip services to academic values, they do not adhere to those values and the *'epistemic honor code'* (Rauch 2018) with respect to their own operations.

We are still at an early stage of development and use of generative AI tools, yet profound and deep structural effects of such tools on education, research, the economy and other areas of society are already becoming visible. It seems like the next frontier of a global battle over power and control over knowledge fought by digital giants. In response, interoperability has been established as principle to improve access and search across content platforms and research repositories and to counter the influence of platform providers: *'A fair and neutral search is essential infrastructure for science. The publication system must not be dependent on algorithms of commercial companies ...'* (Wissenschaftsrat 2022, p. 42).

2.5.2 Research Infrastructures

In a WIRED interview, John Seely Brown responded to the question 'What's the key to research today': *'Tools now drive science. Not theory, not experiment – tools have completely changed the speed and nature of innovation. And they have transformed by orders of magnitude the questions we can ask and answer. We now have the ability to have fast, lasting impact. ... But the real key is how to create a charismatic milieu that combines leading-edge designers with leading-edge tools'* (WIRED Staff 2000). While I do agree with the optimistic assessment of the extend of potential impact on research, a reflection on the challenges of developing productive human-technology configurations (Klein and Watson-Manheim 2021) and the dark side of tool impact and use is missing.

The quote by Orlikowski and Iacono – by now over twenty years old – is a vivid reminder of the societal risks of technology and indeed our responsibilities as scholars to explore and make sense of the mechanisms and the direction, in which technology is transforming societies:

> *'Our future is becoming increasingly dependent on a multiplicity of pervasive and invasive technological artifacts. As IS researchers we have the opportunity and responsibility to influence what future is enacted with those technological artifacts. To do so, however, we must engage deeply and seriously with the artifacts that constitute a central component of that future. Otherwise, we will remain passive observers of the technosocial transformations occurring around us, and we will risk fulfilling our own worst prophecies of technological determinism'* (Orlikowski and Iacono 2001, p. 133).

Brembs et al. (2021) have sketched an alternative model for the design and governance of academic research and publication infrastructure, over which scholars take (more) control, a scholarly workflow governed by the scholarly community and built on open standards:

> *'... there is broad agreement on the goal for a modern scholarly digital infrastructure: it needs to replace traditional journals with a decentralized, resilient, evolvable net-work that is interconnected by open standards, that allow seamlessly moving from one provider to another, under the governance of the scholarly community, de-centering the journal article as the sole scientific output that "counts".*

> *Hence, the replacement needs not only to encompass the literature, but all components of the scholarly workflow, with modern technologies taking care of text, data, and code, allowing dynamic updating, version/quality control, and tracking of contributor-ship. It needs to replace the monopolies connected to the journals with a genuine, functioning and well-regulated market. In this new market, substitutable service providers compete and innovate according to the conditions of the scholarly community, avoiding another vendor lock-in. There is also agreement over the most fundamental requirement for such a replacement to materialize: open standards'* (Brembs et al. 2021, p. 4–5).

So there is hope for academia to escape the myopic focus on highly ranked journal papers as the prime performance metric, which reinforces the dependency on exploitative publishers and information analytics businesses, and to develop innovative formats for research, assessment, publications and distribution of knowledge based on academic values and principles.

2.5.3 Polycrisis, Research Integrity and Academic Values

We live at time of polycrisis (Tooze 2021; Whiting and Park 2023): Concurrent health, ecological, financial, political, legal/regulatory crises, *'where disparate crises interact such that the overall impact far exceeds the sum of each part'* (World Economic Forum 2023, p. 9). Digitalization is related to or implicated in all of these crises and the level of risks caused by technology has increased globally (World Economic Forum 2023, p. 10). Technology has been instrumentalized, politized and weaponized in egregious ways, perhaps most importantly and most visibly in the erosion of social cohesion. Disinformation and misinformation have become the cause or accelerator of crises, see e. g., thesocialdilemma.com, they have been used to undermine the recognition of crises and their causes and to weaken societies' ability and resolve to respond to crises. Moreover they have undermined established knowledge, societal institutions and values, while technology has furthered the *'corrosion of character'* (Sennett 1998).

It is thus all the more important to protect academic values and principles, which require independences but include civic responsibility (Observatory Magna Charta Universitatum 2020) and the integrity of research, the principles and mechanisms of developing and reviewing knowledge claims, including search, access and assessment infrastructures. When looking at infrastructure providers and their services, the trilogy of fairness, accountability and transparency (Shin and Park 2019) seem to be a good litmus test.

This could be complemented by a broader research agenda of phronetic social science *'the study of social phenomena based on a contemporary interpretation of the classical Greek concept phronesis, variously translated as practical judgment, practical wisdom, common sense, or prudence ... [whose] primary purpose ... is ... to contribute to society's practical rationality in elucidating where we are, where we want to go, and what is desirable according to diverse sets of values and interests. The goal of the phronetic approach is to add to society's capacity for value-rational deliberation and action'* (Flyvbjerg 2016, see also Ngwenyama and Klein 2023). One example for value-rational deliberations is ethical foresight analysis (Floridi and Strait 2020).

A number of initiatives have been set-up over the past few years which recognize the need for a value based assessment and development of technology. For example, the need to *'align technology with humanity's best interests. We envision a world with technology that respects our attention, improves our well-being, and strengthens communities.'* (`https://www.humanetech.com/who-we-are#team`), to change current technology paradigms to acknowledge *'the complex relationships between people, society, nature, and machines'*, Digital Humanism (Werthner et al. 2023) or the notion of social welfare computing (Clemons et al. 2022).

Postscript

The reflections in this chapter look back to understand where we are coming from and to avoid repeating the mistakes of the past. They aim to look into the future cognizant of the values of research and autonomous universities and the pattern of technology development and use.

Already in 1986, Winograd and Flores have attempted *'to create a new understanding of how to design computer tools suited to human use and human purposes'* which led them to *'seeking a better understanding of what it means to be human'* (Winograd and Flores 1986, pp. 8, 13). Yet so far, massive technological advances have not translated into more humane societies, well-being, social, economic or political stability. Even after almost 40 years, the questions that we are facing these days are still at this fundamental level, e. g., what is human knowledge, how can the integrity of research be protected, how can we facilitate a humane use of technology that promotes and protects basic human rights and furthers fairness and accountability, transparency and open access to knowledge? And how can we protect what we value from turning into our worst nightmares?

So I guess, looking for answers will require courage and the *'rarest of commodities, time for musing and reflection free of interruption'* (Macneil 1973, p. 691).

Acknowledgement

I am grateful for the many inspiring conversations I have had about the topics covered in this chapter with Bozena I. Mierzejewska, Ojelanki Ngwenyama, Diego Ponte, Frantz Rowe, Simeon Vidolov, and Helle Zinner Henriksen.

References

Aczel, B., Szaszi, B. and Holcombe, A. O. (2021). 'A billion-dollar donation: estimating the cost of researchers' time spent on peer review'. In: *Research integrity and peer review* 6.1, pp. 1–8.

Alvesson, M. and Spicer, A. (2016). '(Un)Conditional surrender? Why do professionals willingly comply with managerialism'. In: *Journal of Organizational Change Management* 29.1, pp. 29–45.

Barlow, J. P. (1996). *A Declaration of the Independence of Cyberspace.* Retrieved March 26, 2023. URL: `https://www.eff.org/cyberspace-independence`.

Bosman, J. and Kramer, B. (2016). *Innovations in Scholarly Communication: Changing Research Workflows.* URL: `https://101innovations.wordpress.com/workflows/`.

Brembs, B., Huneman, P., Schönbrodt, F., Nilsonne, G., Susi, T., Siems, R. and et al. (2021). *Replacing academic journals.* URL: `https://zenodo.org/record/7974116`.

Brin, S. and Page, L. (1998). 'The anatomy of a large scale hypertextual Web search engine'. In: Brisbane, pp. 1–20.

Buranyi, S. (2017). 'Is the staggeringly profitable business of scientific publishing bad for science?: It is an industry like no other, with profit margins to rival Google – and it was created by one of Britain's most notorious tycoons: Robert Maxwell'. In: *The Guardian* (June 27). URL: https://www.theguardian.com/science/2017/jun/27/profitable-business-scientific-publishing-bad-for-science.

Campbell-Kelly, M. and Garcia-Swartz, D. D. (2013). 'The history of the internet: The missing narratives'. In: *Journal of Information Technology* 28.1. URL: http://dx.doi.org/10.1057/jit.2013.4.

Carpenter, T. A. (2020). *Elsevier Has Deployed an End-user Tracking Tool for Security. Should Users Be Concerned About Their Privacy?* URL: https://scholarlykitchen.sspnet.org/2020/10/13/elsevier-has-deployed-an-end-user-tracking-tool-for-security/.

Chen, G., Posada, A. and Chan, L. (2019). 'Vertical Integration in Academic Publishing: Implications for Knowledge Inequality'. In: *Connecting the Knowledge Commons – From Projects to Sustainable Infrastructure. The 22nd International Conference on Electronic Publishing – Revised Selected Papers.* Ed. by L. Chan and P. Mounier. OpenEdition Press.

Chomsky, N., Roberts, I. and Watumul, J. (2023). 'The False Promise of ChatGPT'. In: *New York Times* (March 8). URL: https://www.nytimes.com/2023/03/08/opinion/noam-chomsky-chatgpt-ai.html.

Chubin, D. E. (1985). 'Open Science and Closed Science: Tradeoffs in a Democracy'. In: *Science, Technology & Human Values* 10.2, pp. 73–80.

Clemons, E. K., Schreieck, M., Krcmar, H. and Bui, T. (2022). 'Social Welfare Computing and the management and regulation of new online business models'. In: *Electronic Markets* 32.2, pp. 411–414.

Cusumano, M. A. (2022). 'Data platforms and network effects'. In: *Communications of the ACM* 65.10, pp. 22–24.

Deutsche Forschungsgemeinschaft (DFG) (2021). *Data tracking in research: aggregation and use or sale of usage data by academic publishers. A briefing paper of the Committee on Scientific Library Services and Information Services of the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation).* Bonn. URL: https://www.dfg.de/download/pdf/foerderung/programme/lis/datentracking_papier_de.pdf.

Dolata, U. (2021). 'Varieties of Internet Platforms and their Transformative Capacity'. In: *The Future of Work.* Ed. by C. Suter, J. Cuvi, P. Balsiger and M. Nedelcu. Zürich: Seismo, pp. 100–116.

Dolata, U. (2022). 'Platform Regulation'. In: *The Routledge Handbook of Smart Technologies.* Ed. by H. D. Kurz, M. Schütz, R. Strohmaier and S. S. Zilian. London: Routledge, pp. 457–477.

Dolata, U. and Schrape, J.-F. (2022). *Platform Architectures – The Structuration of Platform Companies on the Internet.* Tech. rep. 2022-01. Stuttgart: University of Stuttgart. URL: https://www.sowi.uni-stuttgart.de/dokumente/forschung/soi/soi_2022_1.Dolata.Schrape.Platform.Architectures.pdf.

Duwe, J. (2022). *Ambidextrous Leadership.* Berlin, Heidelberg: Springer.

Dyson, E., Gilder, G., Keyworth, G. and Toffler, A. (1996). 'Cyberspace and the American Dream: A Magna Carta for the Knowledge Age (Release 1.2, August 22, 1994)'. In: *Information Society* 12.3, pp. 295–308.

Else, H. (2021). 'What Sci-Hub's latest court battle means for research'. In: *Nature* 600.7889, pp. 370–371.

Elsevier (2022a). *Privacy Policy*. URL: Elsevier.com/legal/privacy-policy.

Elsevier (2022b). *Terms and conditions*. URL: https://www.elsevier.com/legal/elsevier-website-terms-and-conditions.

Feigenbaum, E. A. (1989). 'Toward the library of the future'. In: *Long Range Planning* 22.1, pp. 118–123.

Floridi, L. (2019). 'Establishing the rules for building trustworthy AI'. In: *Nature Machine Intelligence* 1.6, pp. 261–262.

Floridi, L. and Strait, A. (2020). 'Ethical Foresight Analysis: What it is and Why it is Needed?' In: *Minds and Machines* 30.1, pp. 77–97.

Flyvbjerg, B. (2016). *What is Phronesis and Phronetic Social Science?* Retrieved March 12, 2017. URL: https://www.linkedin.com/pulse/what-phronesis-phronetic-social-science-bent-flyvbjerg-???-.

Franceschi-Bicchierai, L. (2022). *Academic Journal Claims it Fingerprints PDFs for 'Ransomware' Not Surveillance*. URL: https://www.vice.com/en/article/4aw48g/academic-journal-claims-it-fingerprints-pdfs-for-ransomware-not-surveillance.

Frischmann, B. M. (2006). 'An Economic Theory of Infrastructure and Commons Management'. In: *Minnesota Law Review* 89, pp. 917–1030. URL: http://law.bepress.com/alea/16th/bazaar/art18.

Garcia Morgado, J. (2020). *Pure: Deep Dive*. Elsevier.

Goodman, S. (2022). 'Is It Time to Pay Peer Reviewers?: The system is in trouble. New incentives—money and more recognition—might fix it'. In: *The Chronicle of Higher Education* (December 1).

Gregory, R. W., Henfridsson, O., Kaganer, E. and Kyriakou, H. (2022). 'Data Network Effects: Key Conditions, Shared Data, and the Data Value Duality'. In: *Academy of Management Review* 47.1, pp. 189–192.

Grossmann, A. and Brembs, B. (2021). *Current market rates for scholarly publishing services*. 10, 20.

Haak, W. (2014). *ScienceDirect's Article Recommender gets smarter and casts a wider net: As tool becomes more effective, twice as many people are clicking on recommended articles*. URL: https://www.elsevier.com/connect/sciencedirects-article-recommender-gets-smarter-and-casts-a-wider-net.

Kaplan, S. (2008). 'Framing Contests: Strategy Making Under Uncertainty'. In: *Organization Science* 19.5, pp. 729–752.

Kim, S.-J. and Park, K. S. (2020). 'Market share of the largest publishers in Journal Citation Reports based on journal price and article processing charge'. In: *Science Editing* 7.2, pp. 149–155.

Klein, S. and Watson-Manheim, M. B. (2021). 'The (re-)configuration of digital work in the wake of profound technological innovation: Constellations and hidden work'. In: *Information and Organization* 31.4, p. 100377.

Knecht, S. d. (2020). *Dutch open science deal primarily benefits Elsevier*. URL: https://www.scienceguide.nl/2020/06/open-science-deal-benefits-elsevier/.

Lariviere, V., Haustein, S. and Mongeon, P. (2015). 'The Oligopoly of Academic Publishers in the Digital Era'. In: *PLOS ONE* 10.6, e0127502.

Lessig, L. (2005). *Creative Commons und Fair use: Lessig-Brief Nr. 4*. URL: irights.info/artikel/lessigletters-auf-deutsch-6/5442.

Lohr, S. (2010). 'Big Blue's Smarter Marketing Playbook'. In: *New York Times*. January 12.

Macneil, I. R. (1973). 'The Many Futures of Contracts'. In: *Southern California Law Review* 47, pp. 691–816.

Mazzucato, M. (2018). *The value of everything: Making and taking in the global economy*. New York: Penguin; Public Affairs.

Merton, R. K. (1942). *Science and Technology in a Democratic Order*. URL: `https://www.panarchy.org/merton/science.html`.

Mitomo, H. (2017). *Data Network Effects: Implications for Data Business*. Passau, Germany. URL: `http://hdl.handle.net/10419/169484`.

Monbiot, G. (2011). 'Academic publishers make Murdoch look like a socialist'. In: *The Guardian* (August 29). URL: `www.theguardian.com/commentisfree/2011/aug/29/academic-publishers-murdoch-socialis`.

Münch, R. (2014). *Academic Capitalism*. New York, London: Routledge.

Münch, R. (2022). *Die Herrschaft der Inzidenzen und Evidenzen: Regieren in den Fallstricken des Szientismus*. Frankfurt: Campus Verlag.

Nachtwey, O. and Schaupp, S. (2022). 'Ungleicher Gabentausch – User-Interaktionen und Wertschöpfung auf digitalen Plattformen'. In: *KZfSS Kölner Zeitschrift für Soziologie und Sozialpsychologie* 74.S1, pp. 59–80.

Negroponte, N. (1996). *Being Digital*. Vintage.

Ngwenyama, O. and Klein, S. (2023). 'Some Principles for Conducting Phronetic IS Research'. In: *Handbook of Qualitative Research Methods for Information Systems*. Ed. by R. Davison. Cheltenham, UK: Edward Elgar.

Observatory Magna Charta Universitatum (2020). *Magna Charta Universitatum*. URL: `http://www.magna-charta.org/magna-charta-universitatum/mcu-2020`.

Orlikowski, W. J. and Iacono, C. S. (2001). 'Research Commentary: Desperately Seeking the "IT" in IT Research – A Call to Theorizing the IT Artefact'. In: *ISR* 12.2, pp. 121–134.

Plantin, J.-C., Lagoze, C., Edwards, P. N. and Sandvig, C. (2018). 'Infrastructure studies meet platform studies in the age of Google and Facebook'. In: *New Media & Society* 20.1, pp. 293–310.

Ponte, D. and Klein, S. (2017). 'Research and Web 2.0: Technology, Innovation and Actor Constellations'. In: *Research 2.0 and the Impact of Digital Technologies on Scholarly Inquiry*. Ed. by A. Esposito. Hershey, PA: IGI Global, pp. 17–31.

Puehringer, S., Rath, J. and Griesebner, T. (2021). 'The political economy of academic publishing: On the commodification of a public good'. In: *PLOS ONE* 16.6, e0253226.

Rauch, J. (2018). 'The Constitution of Knowledge'. In: *National Affairs* 43. URL: `https://www.nationalaffairs.com/publications/detail/the-constitution-of-knowledge`.

RELX (2021). *Annual Report and Financial Statements*. URL: `https://www.relx.com/investors/annual-reports/2021`.

Schonfeld, R. C. (2018). *Big Deal: Should Universities Outsource More Core Research Infrastructure?* URL: `https://sr.ithaka.org/wp-content/uploads/2018/01/SR_Issue_Brief-_BIG-DEAL_-01042018.pdf`.

Sennett, R. (1998). *The Corrosion of Character: The Personal Consequences of Work in the New Capitalism*. New York, NY: Norton.

Shin, D. and Park, Y. J. (2019). 'Role of fairness, accountability, and transparency in algorithmic affordance'. In: *Computers in Human Behavior* 98, pp. 277–284.

Spicer, A. (2020). 'Playing the Bullshit Game: How Empty and Misleading Communication Takes Over Organizations'. In: *Organization Theory* 1.2, p. 263178772092970.

STM Association (2021). *STM Global Brief 2021 – Economics and Market Size.* URL: https://www.stm-assoc.org/2021_10_19_STM_Global_Brief_2021_Economics_and_Market_Size.pdf.

Tooze, J. A. (2021). *Shutdown: How COVID shook the world's economy.* New York: Viking an imprint of Penguin Random House LLC.

Waldman, S. (2018). 'Climate Web Pages Erased and Obscured under Trump: Critics of the new administration see a troubling pattern where information is hidden from the public'. In: *Scientific American* (January 10). URL: https://www.scientificamerican.com/article/climate-web-pages-erased-and-obscured-under-trump/.

Werthner, H., Stanger, A., Schiaffonati, V., Knees, P., Hardman, L. and Ghezzi, C. (2023). 'Digital Humanism: The Time Is Now'. In: *Computer* 56.1, pp. 138–142.

Whiting, K. and Park, H. (2023). *This is why 'polycrisis' is a useful way of looking at the world right now.* URL: https://www.weforum.org/agenda/2023/03/polycrisis-adam-tooze-historian-explains/.

Winograd, T. and Flores, F. (1986). *Understanding computers and cognition: A new foundation for design.* Norwood, NJ: Ablex Pub.

WIRED Staff (2000). 'The Debriefing: John Seely Brown'. In: *WIRED* (8.08).

Wissenschaftsrat (2022). *Recommendations on the Transformation of Academic Publishing: Towards Open Access.* Tech. rep. No. Drs. 9477-22_en. German Science And Humanities Council. Cologne. URL: https://www.wissenschaftsrat.de/download/2022/9477-22_en.pdf.

World Economic Forum, ed. (2023). *The Global Risks Report 2023: Insight Report.* URL: https://www.weforum.org/reports/global-.

Writer, B., ed. (2019). *Lithium-Ion Batteries: A Machine-Generated Summary of Current Research.* Cham: Springer International Publishing.

Zelewski, S. (1987). 'Der Informationsbroker'. In: *Die Betriebswirtschaft (DBW)* 47.6, pp. 737–748.

Chapter 3

Moral Responsibility in Conceptual Modeling

Alexander C. Bock and Jens Gulden

There is an irreducible ethical dimension to the activity of conceptual modeling. Conceptual models and conceptual modeling languages exert a profound influence upon our thinking and perception of real-world situations, and they govern the design of new artifacts. Ultimately, as Ulrich Frank has argued, they inform the development of possible future worlds. It is, therefore, imperative to attend to the many direct and indirect ways in which conceptual modeling is both the vehicle and the source of moral decisions. In this chapter, we reflect upon the notion of moral responsibility in conceptual modeling, which we take as an attitude of careful alertness to the ethical ramifications of both the activity and the products of conceptual modeling. Among the issues discussed are the contingency of the scope and content of conceptual models, the formative power and limits of modeling languages, the normative functions of conceptual models, and the role of conceptual modeling in the (re-)production of social reality itself. Our conclusions are intended to contribute to what might be developed into a body of guidelines for responsible conceptual modeling in the future.

3.1 Introduction

It is now almost half a century since *conceptual modeling* emerged as the distinctive modeling paradigm of the field of business information systems. At the time of its inception, conceptual modeling was primarily thought of as a means to support the design of computer-based information systems (IS), like database systems and object-oriented software systems (e. g., Brodie et al. 1984; Loucopoulos 1992; Rolland and Cauvet 1992). In the decades that followed, it gradually occurred to researchers and practitioners that conceptual models can be of value to many other areas of professional practice.

One result of this development is the field now known as *enterprise modeling*, which applies the principles of conceptual modeling to the integrated analysis and (re-)design of organizational structures, business processes, goal systems, and IS infrastructures (e. g., Zachman 1987; Frank 2002; Frank 2014a; Frank et al. 2014; Frank and Bock 2020a; Vernadat 2000; Sandkuhl et al. 2014).

Another product of the expansion of the use of conceptual modeling in recent decades is a sizable inventory of *domain-specific modeling languages* (DSMLs; e. g., France and Rumpe 2005; Frank 2013; Karagiannis et al. 2016). These languages provide modeling concepts specialized toward particular fields of private and public activity, like corporate strategy planning, production and manufacturing, education, and health care. In short, conceptual modeling has found use in the investigation, advancement, and sometimes radical (re-)invention of a variety of spheres of human life.

Every paradigm of analysis and design effective in molding the everyday world of human beings has an inherent and irreducible *ethical dimension*. This is no less true of conceptual modeling. Conceptual modeling is used to inquiry into (what appear to be) real-life situations; it is used to elicit requirements and goals from people; it is used to describe, explore, and evaluate alternatives in the (re-)design of technical, organizational, and social systems; and, ultimately, it is used to specify what, as Ulrich Frank has argued, may become part of *possible future worlds* (see Frank 2017; Frank 2009).[1]

Sometimes, what is designed by means of conceptual models may seem innocuous enough at first blush, as when a simple scheduling program is implemented for a business firm. At other times, what is studied and established by means of conceptual models is likely to have a major impact on the lives of a large number of individuals, as when the strategy of a large corporation is redefined, or when the procedure of operation of a public institution is changed in its essentials. Thus, because and to the extent that the making and use of conceptual models has the potential to affect people's lives, conceptual modeling is susceptible to ethical judgment.

However, with certain notable exceptions, the study of the morally significant features of conceptual modeling has garnered only limited attention to this day. One reason for this state of affairs may be that conceptual modeling is believed to be a morally neutral instrument of analysis. Another reason may lie in the subtlety and indirectness of many of the ways in which conceptual models and modeling languages affect our perception, thinking, reasoning, and action in the present and in the future. And still another reason may be that while certain areas of modeling—for example, medicine, politics, and ecology—are widely recognized as being replete with ethical issues, it is rarely the paradigm of conceptual modeling itself which is seen as harboring mechanisms of ethical significance.

The argument of the present chapter is that the very activity and paradigm of conceptual modeling, too, must be subjected to careful ethical scrutiny. Conceptual modeling and conceptual models exert various influences on the transactions with technical and social systems; and they have greater power to shape our thought and action than might initially be expected. We discuss these matters under the heading of the notion of *moral responsibility in conceptual modeling*, which we take as an attitude of critical alertness to the ethical features and implications of conceptual modeling.

The goal of this chapter is to explore and organize some of the aspects of responsibility in conceptual modeling, drawing on a variety of views from moral philosophy, psychology, research in conceptual modeling, and other fields. Our conclusions are thought of as a contribution to what may be developed into a body of guidelines for responsibility in conceptual modeling in future work. In view of the magnitude of the subject, of course, our investigation is necessarily limited and merely a first step toward a concept of responsible conceptual modeling. But it is our hope that this sketch will be further expanded upon in the future, and that it provides a useful initial orientation for researchers and practitioners interested in raising their awareness of the moral obligations of our field.

1 As the present chapter is written for the Festschrift to the honour of Ulrich Frank, we shall, throughout the text, highlight Ulrich Frank's contributions to the study of the various areas of discussion.

3.2 Fundamentals

In order to lay the foundation for this chapter, we attend to three tasks in this section. The first is a brief review of certain fundamental concepts of moral theory (Section 3.2.1). The second is a discussion of the concept of responsibility and the way in which we wish to tie it to conceptual modeling (Section 3.2.2). The third is a recapitulation of some of the basic principles and mechanisms of conceptual modeling (Section 3.2.3).

3.2.1 Ethics and Normative Moral Theory

The notion of responsible modeling is essentially moral in nature. It is well to sort out at the outset what this means. *Ethics*—or moral philosophy, or moral theory—is one of the first and oldest branches of philosophy, one which has been studied already by the ancient Greek (for a history, see MacIntyre 1966). In his Critique of Pure Reason, Immanuel Kant said that one of three major questions of human reason is, '*What ought I to do?*' (Kant 1996, A805/B833). This he viewed as the central *moral* question, which provides a serviceable indication of the subject of ethics to the present day. In the most general terms, ethics or moral theory is still often introduced as dealing with questions as, 'How should an individual act?,' or 'How should individuals live their lives?' (cf., e. g., Bunnin and Yu 2004, p. 229; Copp 2006, p. 4; Driver 2005, p. 31). Thus, the concern with questions of *ought* and *should* is one of the hallmarks of ethics.

The bodies of principles which provide answers to questions about oughts and shoulds are known as *morality* or *morals* (see, e. g., Crisp 2000, p. 256; Bunnin and Yu 2004, p. 228). The items of which these systems are compromised are usually called *moral principles* or *norms*. The analysis of morality characteristically invokes certain fundamental concepts. These include, above all, the cardinal evaluative concepts of *good* and *bad*, and *right* and *wrong*, as well as a variety of adjacent notions, like virtue, duty, justice and freedom (cf., e. g., Blackburn 2005, p. 121; Crisp 2000, p. 256; Bunnin and Yu 2004, p. 228; Copp 2006, p. 4). As may be noticed, the concepts of virtue and duty are, intuitively, closely tied to *responsibility*. We turn to the latter notion below.

One way to look at the realm of ethics is to divide it into three divisions. The first division, *normative ethics* or normative moral theory, attempts to answer the basic moral questions indicated above, which is to say, it attempts to establish moral principles (e. g., Copp 2006, p. 4; Driver 2005, p. 31). Driver says that normative ethics 'is concerned with the articulation and the justification of the fundamental principles that govern the issues of how we should live and what we morally ought to do' (Driver 2005, p. 31). The second division, *descriptive ethics*, investigates the actual systems of morality endorsed in different cultures. It is sometimes assigned to anthropology rather than philosophy (e. g., Crisp 2000, p. 256). The third division, *meta-ethics*, is directed to the study of the status and nature of the objects of ethics itself (e. g., Crisp 2000, p. 256; Copp 2006, p. 5). For example, a classical question of meta-ethics is whether moral principles can be objectively true or false.

On the basis of these concepts, it is easy to state the orientation of the present chapter. Our interest falls into the realm of *normative ethics*, because we are essentially raising the question, 'How should one conduct oneself in conceptual modeling?' And consequently, the tentative conclusions presented in Section 3.3 are effectively (precursors to) varieties of moral principles.

3.2.2 The Concept of Responsibility

We may then turn to the particular ethical concept at the center of this chapter: *responsibility*. Responsibility, like most other ethical concepts, is a difficult and complex notion. There are a variety of contemporary philosophical views of responsibility, and each of them has been worked out in a number of ways. In what follows, we concentrate on a few relatively basic and widely held ideas, seeking to tie them to the activity of conceptual modeling. This results in a first skeletal sketch of responsible conceptual modeling.

A number of accounts of responsibility, as just mentioned, have been advanced by philosophers (for a concise overview, see Clarke 2010, pp. 264–265). One view defines it in terms of *attributability* (see Clarke 2010, p. 264). It states that an individual can be responsible for anything attributable to them, causally or otherwise. According to this view, even thoughts or latent beliefs can be the objects of responsibility. However, in the present chapter we concentrate on the objects of responsibility which are at the center of most treatments, namely, 'actions and their consequences' (Bunnin and Yu 2004, p. 606). More specifically, the actions of interest here are, naturally, those surrounding the construction of conceptual models; and the consequences of interest are the effects of these models on technological design and human life.

Another important and almost invariably accepted condition for a person to be responsible for an action is that of *(moral) agency* (e. g., Uniacke 2010, p. 596). That is to say, it is necessary that 'the person responsible was acting as a moral agent: [...] she acted voluntarily and with understanding of the moral quality of her actions' (Uniacke 2010, p. 596). This chapter, too, takes it that responsibility is premised upon agency. Agency is closely related to another account of responsibility which has become popular in recent decades. It explains responsibility as *answerability*. One such analysis is that of Lucas (Lucas 1993; cf. also Duff 2000). After observing that the etymological origin of the word 'responsible' lies in the Latin *respondeo*, which means, 'I answer,' he suggests: 'So the central core of the concept of responsibility is that I can be asked the question: 'Why did you do it?' and be obliged to give an answer' (Lucas 1993, p. 5). That is, responsible conceptual modeling is seen as entailing, at bottom, the ability to explain why a conceptual model has been developed in the way it has.

Finally, a certain distinction between two species of responsibility is widely endorsed in philosophy: the distinction between *prospective* and *retrospective* responsibility (see, e. g., Duff 2000, p. 768; Uniacke 2010, p. 597; Clarke 2010, p. 263). Prospective responsibility concerns the 'things it is up to us to attend to' (Duff 2000, p. 768). It is reflected, for example, in professional obligations, such as the obligation of a firefighter to quench the flames when and where possible. In contrast, 'we are retrospectively responsible for what we do or bring about' (Uniacke 2010, p. 597). For example, a firefighter who had the unfettered opportunity to put out a fire but chose not to is responsible for that act (or omission). What we describe in this chapter as responsibility in conceptual modeling is essentially a form of *prospective* responsibility. This means that we interpret it as the careful attending to a variety of moral issues *prior to*, and *during*, the activity of modeling. In contrast, we are not so much concerned with the retrospective analysis of the effects of models which have already been constructed.[2]

2 A project devoted to this task would need to build upon still other views of responsibility, such as that which focuses on responsibility in the sense of *accountability* (see, e. g., Clarke 2010, p. 264; Bunnin and Yu 2004, p. 606).

3.2.3 Conceptual Modeling

Conceptual models form a distinct category of models. They are built according to certain specific principles and mechanisms, which together constitute the paradigm of *conceptual (meta-)modeling*. A variety of conceptual (meta-)modeling systems have been advanced in the past decades. For present purposes, it is enough to bring back to mind some basic concepts and distinctions, and to point out one special perspective on conceptual models.

A conceptual model is distinguished by the fact that each of its elements is an *instance* of exactly one previously specified *modeling concept.* A coherent set of modeling concepts for a certain purpose is normally called a *conceptual modeling language.* Accordingly, as Frank states, a 'conceptual model is created through the use of a *modelling language*' (Frank 2011, p. 24). The basic mechanism of instantiation is illustrated in Figure 3.1. For example, as is seen on the left-hand side of the figure, the modeling concept 'Class' might be instantiated into the model elements 'Car' and 'Engine', and the modeling concept 'Attribute' might be instantiated into the element 'fuelType', forming a part of 'Engine'. Every conceptual model is built up from instances of a predefined set of modeling concepts in this manner.[3] Usually, conceptual models are represented in the form of diagrams, as in Figure 3.1, but it is also possible to use other symbolic and pictorial forms of representation.

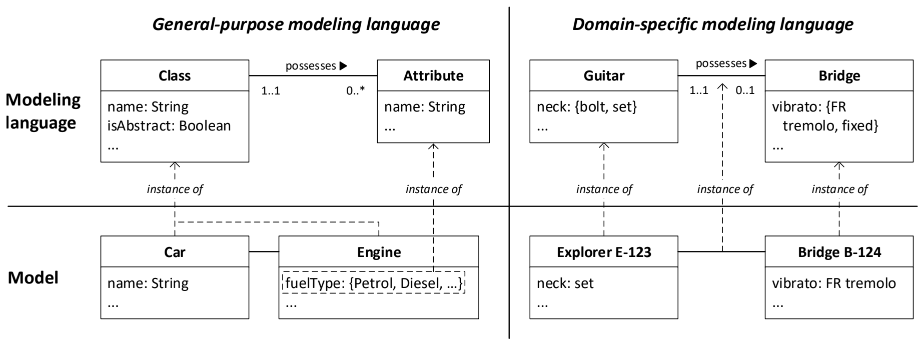


Figure 3.1: Conceptual modeling: Some central mechanisms and distinctions

Figure 3.1 shows another important distinction: that between *general-purpose modeling languages* (GPML) and *domain-specific modeling languages* (DSML; see, e. g., Frank 2010, pp. 1–2). Abstract and general modeling concepts such as 'Class', 'Attribute', and 'Association' are characteristic of GPML. DSMLs, in contrast, provide modeling concepts specifically intended to describe states of affairs in particular domains or areas of application. For example, as is shown on the right-hand side of Fig. 3.1, a DSML for the documentation of music instruments might provide such modeling concepts as 'Guitar' and 'Bridge'. The main advantage of DSMLs is that they are much more expressive than GPML and they can contain

3 We are here concentrating on the classical paradigm of conceptual modeling, which distinguishes clearly between the level of the modeling language and that of the model. In recent years, the paradigm of *multi-level* conceptual modeling, which transcends this distinction, has received growing attention. Multi-level conceptual modeling constitutes a fascinating extension of classical conceptual modeling. Among other things, it relaxes the traditional dichotomy between instantiation and generalization/specialization (see Frank 2014b; Frank 2022).

specialized rules and visual notations for the domain under consideration, but they are, of necessity, much more limited in their scope of application (for more on this conflict between GPMLs and DSMLs, see Frank 2011, pp. 1–2, 17–19). As we shall see in the next section, the scope of what can be expressed by means of a modeling language is of considerable relevance to responsibility in conceptual modeling.

One special perspective on conceptual models is that of *embodied cognition* (Foglia and Wilson 2013; Gulden 2004). This perspective emphasizes that conceptual models represent meaning in *spatially extended and logically organized* structures, chiefly exemplified by schematic diagrams. Because of this, conceptual models are amenable to other modes of cognitive access than spoken and written language. Conceptual models can be visually interpreted with greater ease than ordinary text, and, unlike text, they can also be explored in any order, linear or not. Another consequence is that conceptual models allow to perform reasoning based on spatial and pattern-based cognitive operations. The connection between meaning and symbolic representation is in parts constituted by spatial movements that interpreters may imagine when inspecting conceptual models in the form of visual diagrams. However, while the capability to facilitate embodied cognition has the aforementioned advantages, it also holds certain risks. Conceptual models may come to be seen as possessing a fixed, physical nature, which may erroneously lead some interpreters to take them as representing an objective reality or to carry impartial truth. We come back to this point in Section 3.3.3.

3.3 Responsibility in Conceptual Modeling

We now turn to the main subject of this chapter. The purpose of this section is to discuss some of the issues which, we suggest, are both among the sources and objects of responsibility in conceptual modeling. These fall into four rubrics: firstly, the *contingency of the scope and content* of models (Section 3.3.1), secondly, the *formative power and limits* of modeling languages (Section 3.3.2), thirdly, the *normative functions* of conceptual models (Section 3.3.3), and, fourthly, the role of conceptual modeling in the *(re-)production of social reality* (Section 3.3.4). These are, of course, not the only areas of conceptual modeling susceptible to moral considerations, but they are, arguably, some of the most important ones. Furthermore, these areas are not independent from one another; as we will indicate they build upon one another in more than one way.

In the discussion of each of these four themes, we address ourselves to two tasks. First, we single out certain issues and factors of moral significance which are easily overlooked in the practice of conceptual modeling. These include subtle factors exerting an influence on the design of conceptual models, principal sources of difficulty or contingency, and unintended or unforeseen effects of the resulting conceptual models. We discuss both the nature of these phenomena, and we indicate why they are of ethical relevance. Second, by way of conclusion, we suggest tentative elements of what may be developed into a body of guidelines for responsibility in conceptual modeling in the future. Of course, our suggestions are neither intended nor designed as definite and rigid rules. Instead, they are meant to encourage an attitude of careful attentiveness to moral matters in conceptual modeling—that is, an attitude of *responsibility* in conceptual modeling.

It should further be made clear at the outset that most issues we shall examine are not unknown in conceptual modeling research, nor are our conclusions hitherto unparalleled.

To the contrary, many of them will be familiar both to theorists and mindful practitioners of conceptual modeling. Yet while the basic themes are well-known, to the best of our knowledge the arguments that follow have not yet been pulled together and organized as presented here.

3.3.1 The Contingency of the Scope and Content of Conceptual Models

The first cluster of issues and arguments revolves around the *contingency of the scope and content* of conceptual models. By this we mean the fact that the design of conceptual models under practical conditions is influenced by a variety of contingent factors. What is contingent is possible but not necessary (e. g., Wolters 2010, p. 324). A contingent fact is one which turned out one way but could have turned out differently as well. The great significance of contingency in social affairs for the field of Information Systems Research is persistently emphasized by Ulrich Frank (see Frank 2009, pp. 162–165; Frank 2017, pp. 5728–5729). To put the present interest more simply, we are in what follows concerned with reasons for the cliched but nonetheless significant observation that in practice there is usually no such thing as a single correct conceptual model.

The first and simplest point to be made is that any individual act of communication – whether by means of spoken or written language, or by means of conceptual models – is preceded (and, perhaps, paralleled) by the conscious or unconscious determination of what is and is not communicated. It is impossible to communicate everything all at once, and it is also impossible to communicate every imaginable piece of information about the domain of interest at one time. Thus, the construction of a conceptual model necessitates decisions about the model's *scope* and *content.* These decisions cover at least two dimensions: decisions about the features of the area of discourse to be included in the model (*breadth*), and decisions about the degree of detail at which these features are described (*depth* or *granularity*). So the first thing to be kept in mind is that however careful and reflective the design of a particular conceptual model may have been, that model, by definition, covers certain aspects and ignores others.

The central area of difficulties pertaining to the scope and content of conceptual models we wish to discuss here, then, has to do with the possibility and limits of *knowledge about reality.* Almost all conceptual models are, at least to some extent, intended to give an account of what is or what may become the case in reality. For example, an enterprise model may be intended to describe the existing organizational structure and the existing IS infrastructure of an organization, and even a model of a system yet to be developed normally contains a number of assumptions about the way things are in the intended application domain. Thus, conceptual modeling virtually always involves (but, of course, is not limited to) the acquisition of knowledge and information about what is thought to be reality.

The capacity of human beings to obtain knowledge of the world is studied in a wide range of disciplines. First and foremost, it is one of the fundamental subjects of epistemology, the theory of knowledge (see, e. g., Audi 2010), and methodology, the study of scientific methods (see, e. g., Oldroyd 1986). Moreover, several branches of psychology, sociology, and other fields of research are, directly or indirectly, concerned with the ways in which human beings experience and interpret their surroundings. Investigators in these various disciplines have postulated over the past centuries a large and varied array of contingent factors potentially exerting an influence upon, and implying principal limits to, the generation of knowledge. Because of the reason just stated, all these epistemological and other arguments apply, *mutatis mutandis*, to the development of conceptual models as well. This point, of course, is well-known among theorists of conceptual modeling. Wolff and Frank, for example, have suggested that epistemological reflections can inform the evaluation of conceptual models: 'In many ways, models are similar to scientific theories. Therefore, it seems reasonable to adapt criteria being used to evaluate theories to judge model quality' (Wolff and Frank 2005).

What we seek to emphasize in the following, then, is that a *wide spectrum of contingent factors* may influence or limit the process of gathering of knowledge which underpins, and enters into, the design of conceptual models. These may occur at widely varying levels, ranging from the most elemental level of neurophysiology, to the level of cognition and thought, up to the level of cultural norms and values.

The argument that even the *neurophysiological* and biochemical constitution of human beings is a determinant of knowledge has been emphasized, in particular, by the radical constructivists (e. g., Maturana and Varela 1992; Glasersfeld 1995). After all, the nature of human sense-experience, and the ways in which it is pre-selected, processed, and even constructed neurophysiologically, essentially mark out the reach of our immediate transactions with the world. One of the most obvious areas where such elemental processes may lead to tensions in conceptual modeling is in the modeling of features relating to visual perception, as in the design of graphical user interfaces. How people conceptually think about colors, geometrical shapes, and so on, need not be congruent with how they actually perceive them.

Another important set of considerations related to responsible modeling has been presented, with different emphases, under a number of different headings. It involves, for example, what Simon famously described as *bounded rationality* (Simon 1957), what Tversky and Kahneman studied in their *heuristics and biases* program (e. g., Tversky and Kahneman 1974; Tversky and Kahneman 1981), and what more recently has been discussed under the label *cognitive economy* (see, e. g., Fischer 2011).[4]

Despite the variation between these proposals, there is a common thread: it is suggested that because of limitations in cognitive resources, the human mind consistently deploys a variety of simplifying mechanisms to facilitate, and reduce the cost of, the selection and processing of experience and information. For example, Simon has argued that the sheer complexity of what people perceive as reality inevitably requires them to settle with 'good enough', rather than optimal courses of action.

Tversky and Kahneman, in turn, have shown, among other things, that people often mistakenly make dependent their judgments about the likelihood of certain types of events on the ease with which instances can be retrieved from memory. These and other cognitive mechanisms and their effects may affect conceptual modeling in a number of ways. For example, in the design of a business process model, the operating staff may overemphasize the significance of some very rare type of event or activity, because they recently encountered an exceptional case.

The final group of hypothesized influencing factors to be mentioned here consists of *cultural norms and values.* It is known that members of different cultures substantially differ in their perception of, and attitudes toward, a variety of objects and behaviors (see, e. g., Hofstede 1980; Hofstede 2011; Schwartz 2012). A related but broader analytical concept embraced by some sociologists is that of a *frame* (see, e. g., Goffman 1974; Gitlin 1980). This concept is intended to cover the whole of the various beliefs, values, concepts, and principles

4 *Cognitive economy* is not to be confused with *cognitive economics.* The latter is the research field attempting to bring together economic decision theory and cognitive psychology.

which the members of a certain societal group typically tend to acquire in socialization.[5] Once learned and internalized, frames are argued to influence the interpretation of social reality, especially when it comes to the greater scheme of things. Gitlin has portrayed frames as 'principles of selection, emphasis and presentation composed of little tacit theories about what exists, what happens, and what matters' (Gitlin 1980, p. 6). Again, these and similar phenomena may influence conceptual modeling in various ways. To take a simple example, consider the modeling of the physical resources of a manufacturing company. Naturally, most people will tend to think here about such resources as raw materials, machinery, transport vehicles, and so forth, but not about the chairs, desks, and IT infrastructure in the administrative offices. Thus, the learned stereotypical image of a manufacturing firm may subtly steer the course of the analysis.

The foregoing are some examples of the many theoretical arguments to the effect that human knowledge of reality may be subtly or radically, unconsciously or consciously, affected by contingent factors and limits. We do not wish to make any claim as to the exact validity of these or any other theories. Also, we do not, of course, purport that this list is exhaustive. One significant cluster of additional arguments, for example, concerns the role of concepts and language; to this theme we turn in the next subsection. The point we seek to make is that because conceptual models are themselves both products and means of expressing knowledge, their design may be subject to all the indicated and other contingent factors and limitations.

Apart from the variety of epistemological contingencies in modeling, which, in a sense, are largely *internal* to the individual modelers, there are, of course, also a variety of very mundane practical contingencies, which, in a sense, are *externally* imposed upon the modelers. These mainly revolve around the requirements, constraints, and conventions of conceptual modeling in practice.

One important fact is that in professional practice, conceptual modeling usually forms part of a larger project or organizational function, such as a software development project or an enterprise architecture management process. Organizational projects and functions are invariably restricted by a variety of constraints. Time, money, and personnel are limited, and the management many times has all sorts of expectations regarding the results. For example, the conceptual modelers may be required to describe the organization in terms of some recent trend in information systems development, such as service orientation, big data, microservices, or low-code development, even though these perspectives may be insufficiently worked out or inappropriate to the matter under consideration. Another important point is that there are a host of already existing practical guidelines, bodies of rules, and conventions for conceptual modeling. These range from principles for proper modeling presented in research to informal conventions shared by individual modeling teams in practice. Often, the advice submitted by one source is in conflict with that provided elsewhere. And quite independently of the quality of the many available recommendations, it is clear that they substantially affect how conceptual models are built in practice.

Finally, it is worth reminding ourselves that the design of conceptual models is, ultimately, carried out by individuals who have their own interests and, perhaps, hidden

5 The concept of a frame is not exclusive to sociology; it is also adduced, for example, in psychology. In fact, so-called framing effects have been an important object of study in the heuristics and biases program of Tversky and Kahneman, as just mentioned above (see Tversky and Kahneman 1981). But naturally, psychologists have interpreted the concept of a frame as relating to features of individual mentality rather than cultural norms and values.

agendas. Therefore, it is well possible that a conceptual modeler leaves out, underplays, or overemphasizes certain aspects of the domain of discourse so as to influence decisions and activities to their own gain. This is particularly critical, because conceptual models can even figure in the (re-)construction of social reality, a point to which we return in Section 3.3.4.

In summary, there is a wide range of contingent factors potentially bearing upon the construction of conceptual models. The consequence is that the scope and content of even a professionally developed conceptual model can be, and usually is, subject to numerous influences and limitations other than the features of what might be perceived as the 'genuine', 'true', or 'correct' subject of the model. Moreover, many of these influences and limitations are subtle and can easily escape the attention of the makers and interpreters of the model. Now it is, of course, impossible to avoid the effects of contingent factors altogether, and it is also impossible be mindful of all of them at all times. However, since conceptual models have very real consequences for human lives (as we will further point out in Section 3.3.3), it would seem appropriate to make oneself aware of the possibility and potential magnitude of these factors of influence. This we suggest as the first major component of responsibility in conceptual modeling.

3.3.2 Modeling Language and Natural Language: Formative Power and Limits

The second area of arguments and complications we regard as fundamental to responsible modeling centers on the role of *language* and *concepts*. Language and concepts are of integral importance to conceptual modeling in a double sense. First, as pointed out in Section 3.2.3, conceptual models themselves are always the product of certain predefined conceptual modeling languages and modeling concepts. Thus, the very paradigm of conceptual modeling is premised upon *(semi-)formal* mechanisms that mimic certain features of natural language and mental concepts. Second, both modeling languages and conceptual models are usually designed in orientation to the *natural* languages and concepts used in the given area of application. Thus, certain effects of natural language and mental concepts can be expected to exert a considerable influence upon the design of conceptual models. Given this double significance, it is not surprising that the discussion of the role of language in model-building and our understanding of the world at large has an appreciable tradition in conceptual modeling research. Among the writers who have highlighted it most forcefully and persuasively is Ulrich Frank (see, e. g., Frank 1999; Frank 2017).

In what follows, we shall point out certain ramifications of language at both of the aforementioned levels of analysis. In a way, the leitmotif of this subsection is well captured by Wittgenstein's well-known and often-cited aphorism, 'the limits of my language mean the limits of my world' (Wittgenstein 1922, p. 5.6). However, just as Wittgenstein's early language philosophy has provoked much criticism (not least, of course, by himself), so most other theories of the role and nature of language and concepts have been and still are the subject of controversy in philosophy, psychology, and other disciplines. As before, it is not our intention to defend any particular account of language; instead, we solely seek to draw attention to some basic issues and certain widely accepted findings.

Conceptual Limits and Biases of the Modeling Language

The most important point to be made in what follows is that every conceptual modeling language permits to *express* certain aspects of the domain under consideration, while it simultaneously and necessarily forces the modelers to *ignore* others. The reason for this is the simple fact that, as noted in Section 3.2.3, every element of a conceptual model is an instance of exactly one predefined modeling concept, which in turn constitutes part of a given modeling language. Thus, a conceptual model can, in principle, cover only those features of the domain of interest which fall under the modeling concepts of the modeling language applied. An example which makes this obvious is the difference between static modeling languages (such as class diagrams in the Unified Modeling Language; UML Object Management Group 2011b) and dynamic modeling languages (such as the Business Process Model and Notation; BPMN Object Management Group 2011a). These modeling languages, by design, enable – and require – the modelers to take different perspectives on the object of study. To wit, UML class diagrams provide modeling concepts to define classes, attributes, associations, and so on, whereas BPMN provides modeling concepts to describe activities, events, conditions, procedural relations, and so forth. Consequently, a business processes cannot be properly described in the form of a UML class diagram; and classes and their static associations cannot be properly specified in the form of a BPMN model.

Analogous but still more pronounced limitations are manifest in domain-specific modeling languages. As noted in Section 3.2.3, each DSML makes available specialized modeling concepts to represent the states of affairs in more or less narrowly circumscribed spheres of thought and activity, while neglecting all others. For example, the simple DSML indicated in the right-hand portion of Fig. 3.1 allows to model music instruments, but all other imaginable topics and objects of potential interest are beyond its reach. For instance, it cannot be used to document, say, the organizational structure of a business firm. Some modeling frameworks, such as MEMO (Multi-Perspective Enterprise Modeling; Frank 2014a; Frank and Bock 2020a) incorporate a number of separate DSMLs to make possible the integrative description and analysis of a wide range of aspects of an organization. Yet even then, certain features will remain inexpressible in terms of the modeling languages provided. Thus, the modeling language determines what may and may not be represented in the model under construction.

To be sure, the limitations of individual modeling languages are perfectly obvious when contrasting static and dynamic modeling languages, or when comparing DSMLs tailored toward different domains. But many times the influence and the biases of the modeling language at hand are not so readily apparent. In particular, it is easily overlooked that even when it comes to their intended application area, conceptual modeling languages always bring to the fore certain aspects and disregard others. This is well demonstrated by the example of modeling approaches for problem solving and decision making (for a detailed presentation and illustration of the following argument, see Bock 2016). Examples of such approaches include the framework of orthodox decision theory, cognitive maps, goal models, and influence diagrams. Often, these approaches are presented by their respective originators or proponents as though they would enable the description of the problem situation in its entirety and 'as it is'. But in actuality, each of these approaches only affords to describe certain problem aspects, while completely dismissing others. For example, the framework of classical decision theory advises to define a decision problem in terms of goals, alternatives, and consequences; cognitive maps suggest to represent it in terms of causal relations between different phenomena; and goal modeling languages render it as a network of goals interrelated to one another as means and ends (among other things). Thus, depending on the modeling language one chooses, the exact same problem situation may be represented and interpreted in completely different ways.

And, of course, it does not end with these differences in representation. Models for problem solving and decision making constructed by means of different modeling languages will lend themselves toward different types of analysis (we discuss the normative role of models in more detail in Section 3.3.3). For example, a classical decision-theoretical model lends itself toward the analysis of expected utilities of various alternatives, whereas a cognitive map lends itself toward causal reasoning and the development of causal hypotheses. Thus, the choice of the modeling language has a profound effect on analysis and design. It affects what can and cannot be recognized as part of the situation in the first place, and it recommends certain modes of procedure over others.

Modeling Language and Natural Language: Some Analogies

What is interesting, then, is that arguments more or less similar to those just made in relation to conceptual models and modeling languages have been articulated, in many different degrees and forms, in relation to mental concepts and natural language as well. Theories and views of this sort have been advanced in the philosophy of language, linguistics, cognitive psychology, and other fields. It is beyond our concern to discuss any of these proposals in detail, but it is worthwhile to point out some salient analogies.

One enormously influential thesis related to the foregoing discussion is that *concepts*, *qua* mental representations, are the basic unit of organization of human cognition. This thesis has been, and still is, endorsed – albeit in widely varying ways – both in psychology and in philosophy. One philosophical work where it figures prominently, of course, is Kant's Critique of Pure Reason, which proceeds on the premise that 'thought is cognition through concepts' (Kant 1996, A69/B94). But even in modern cognitive psychology, the view that concepts constitute the components or constituents of thought is still widely held (see, e. g., Laurence and Margolis 1999; Murphy 2004). Although there are many subtleties and complications to these propositions, they bring to mind a fascinating analogy: mental concepts are to human thought what modeling languages are to conceptual models.

Another congenial family of views is discussed in linguistics and psychology under the heading of *linguistic relativity*, or linguistic relativism. The foremost proponent of linguistic relativity was Whorf (Carroll et al. 2012; for a discussion, see Regier and Kay 2009). In its strongest version, linguistic relativism holds that language is the vehicle of all human thought and perception. In other words, it is claimed that one cannot think or perceive what is not covered by one's language. For example, one hypothesis was that the color words of different natural languages substantially affect, or even determine, the perception of colors on the part of the speakers of that language. If this were true, then a person knowing only the words 'green' and 'blue' would not be able to distinguish the four different shades of green and blue shown in Fig. 3.2. But this strongest version of linguistic relativity has long been refuted (for a review, see Wolff and Holmes 2011). One reason for this is, in fact, demonstrated by Fig. 3.2: even if one knew only two colors words, one could still perceive more than two shades of color.

Nonetheless, subtler and more nuanced versions of linguistic relativity are still under discussion (see Wolff and Holmes 2011). The terminological and methodical apparatus of the literature on linguistic relativism may be useful in further examining the influences and biases of modeling languages indicated above. In fact, one might argue that modeling languages give rise to a much stronger variety of linguistic relativity than natural languages, because normally the semantics of, and distinctions between, modeling concepts are much

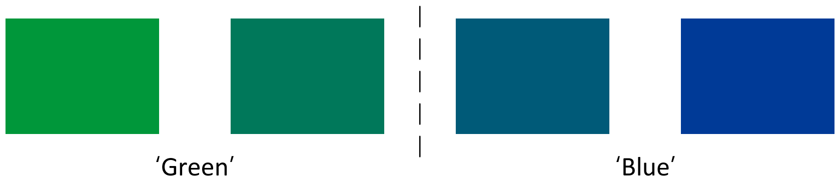


Figure 3.2: Two color words and four shades of green and blue (adapted from Regier and Kay 2009, p. 440)

more rigid and well-defined than those of natural language words, and the number of modeling concepts in a modeling language is much smaller than the number of words in a natural language. This, again, can be illustrated with reference to Fig. 3.2. Suppose an attribute of a modeling concept to describe physical objects could only take the value 'green' or 'blue'. Then the differences between the four shades of color shown in Fig. 3.2 were inexpressible by means of that modeling language, and they would be lost in the resulting model.

Some other thoughts about language and conceptual models can be brought out in relation to Wittgenstein's philosophy of language. The late Wittgenstein thought of the use of natural language as socially constructed 'language games' (Wittgenstein 1958, §66), and at the same time refused the idea that there could be a 'private language' (Wittgenstein 1958, §243). But in conceptual modeling the situation is rather different. Modeling languages are deliberate constructions, possibly created by one or a few individuals who are initially the only users of these languages. They are not necessarily naturally and socially developed games, as Wittgenstein's conception suggests, but may initially indeed be quasi-private languages, which only later may attain the status of a widely accepted language reflective of the social reality of its users. Because of this paramount role of a few select persons in the design of modeling languages, it seems prudent that careful attention be directed to their purposes, intentions, and attitudes.

In a similar vein, one may interpret the development of a modeling language as a sort of self-reflexive feedback loop where language development occurs through language use, mirroring another idea implicit in Wittgenstein's writings on language games. When designing a DSML, one usually begins by drafting a rudimentary version of the language and building initial example models to assess the merits and limits of the draft (see Frank 2010; Frank 2013). The first example models usually do not fulfill all practical requirements, so that the language draft is modified accordingly. Following this pattern, the language is, then, refined and sample models are evaluated until a satisfactory language definition is obtained. What is worth remarking here is that in every iteration of this process all the contingent factors indicated in the previous subsection may have an effect on the activities of the language developers. Moreover, since the process of developing a modeling language is itself mediated by the use of a natural language, all the subtle biases of the existing language may, directly or indirectly, influence the design of the new modeling language.

In summary, the construction of conceptual models is governed, influenced, and limited by language at two principal levels. The first and most obvious level is that of the modeling language. As has been illustrated, the modeling language determines what can and cannot be expressed in the conceptual modeling under design. In consequence, it not only places constraints upon the possible objects of representation, but it also directs the attention of the modelers and interpreters to certain aspects of the situation while leading them to neglect others. Sometimes, the analytical foci and gaps of the modeling languages in use may be obvious enough, but sometimes, they may be difficult to see or even actively denied by the originators of those languages. Thus, we suggest that responsibility in conceptual modeling involves careful reflection about the potential limits and biases of the modeling languages at hand. When and where possible, the integration of several distinct modeling languages may help avoid oversights in the analysis.

The next level at which language enters into conceptual modeling, then, is that of natural language. As we have seen (and as many other conceptual modeling researchers have long pointed out), the activity of conceptual modeling is in many ways tied to natural language and mental concepts. Thus, in the same spirit as in the previous subsection, we propose that responsibility in conceptual modeling should also consist in an attitude of alertness to the role of natural language and its concepts as contingent factors bearing upon the modeling process. We will return to another interesting form of interaction between modeling languages and natural languages in Section 3.3.4.

3.3.3 The Normative Functions of Models

We now turn to the reason why the problems and difficulties examined in the two previous subsections are of *moral* significance in the first place. According to what has been discussed until now, the activity of conceptual modeling is subject to a variety of sometimes overt, sometimes covert, contingent factors. But why is this of moral relevance? One simple, but crucial answer is that conceptual models can assume *normative* functions. As will be recalled from Section 3.2.1, the hallmark of (moral) norms is the *ought.* Thus, by saying that conceptual models can assume normative functions, we mean that conceptual models can, in various explicit and implicit ways, be designed or interpreted to say something about what *ought* to be done—how one *should* act or think. This is the reason why, we suggest, it is so important to pay heed to the various sources of influence pointed out before: they all may, ultimately, affect the design of artifacts which, intended or not, serve as an orientation for human thought and action. Normativity in (conceptual) models, of course, is an exceedingly deep and difficult subject in its own right. But for present purposes, we need not delve into the philosophy of normativity. In what follows, we shall only point out three overlapping levels at which conceptual models can assume a normative role.

The first might be called the *conceptual* level. By this we mean two related but distinct capabilities of conceptual models. First is the ability of conceptual models to shape the *thinking* about the modeled domain. A conceptual model can act as the framework within which the modelers and the interpreters of the model understand and reason about the modeled sphere of reality. The conceptual model can take on this role because it describes the domain of interest in conceptual terms, and at the same time it demarcates its borders, determining what is and is not part of the universe of discourse. The second relevant ability of conceptual models is that they can define the subject and structure of *social discussion.* It is, indeed, one of the prime purposes of conceptual models to provide a basis for social interaction and common understanding. When used in this way, models may steer the course of the debate, and they are likely to influence what aspects are recognized as relevant and made the topic of conversation to begin with.

The second might be called the *operational* level. By this we mean that the fact that conceptual models, regardless of whether or not they have been explicitly designed for normative purposes, can form the basis, in whole or in part, of decisions about what *actions* to take. The decisions formed on the grounds of models in practice are manifold. They may concern the acquisition of a new software system or a new production machine, resource allocations, and the distribution of workloads and responsibilities. They may even have to do with business model development, strategy formation, policy-making, and many other high-level decisions. It is only a slight exaggeration to say that conceptual models can, ultimately, affect the course of the world and society at large.

The third level might be called the *prescriptive* level. By this we mean the function of those conceptual models which are expressly intended to assume a normative role. This includes all conceptual models representing what ought to be done, what is to be constructed, and so on. Examples include the plan of a new IS infrastructure to be established, the architecture model of a house to be build, and the process model of a new standard procedure to be implemented in a company. Prescriptive conceptual models govern the actions of people different from the model-makers, which, once again, means that the effects of conceptual models may be deferred and distributed over time and place, within an organization or even society as a whole. Among the most ambitious sorts of prescriptive models would be prescriptive models of possible future worlds (Frank 2009; Frank 2017).

In summary, there are at least three levels at which conceptual models can assume a normative function. This is of relevance to responsibility in conceptual modeling in two major ways. First, it constitutes the reason why the arguments of the previous two subsections are of moral significance in the first place. Precisely because conceptual models can be intended or interpreted to make claims about what ought to be done, it is important to reveal and, where possible, reduce unintended influences upon the modeling process. Second, we suggest that an awareness of the potential normative functions of conceptual models is itself an important part of responsibility in conceptual modeling. It is important to keep in mind that, whether intended or not, conceptual models are often taken as a guide to how one ought to think and act. This is particularly important in settings where conceptual models or modeling methods are presented or taken as supposedly goal-neutral instruments.

Furthermore, it is in connection with normativity that the perspective of embodied cognition, previously mentioned in Section 3.2.3, becomes relevant. Because conceptual models are normally represented in the form of visual diagrams, hence spatial structures with a seemingly rigid and sometimes forbiddingly complex appearance, they may be erroneously interpreted by some as possessing an authoritative and immutable status which is out of place. This misconception might prevent some interpreters from questioning the content of a given conceptual model, which is especially serious when that model is simultaneously used as a normative basis of reference. Thus, we suggest that another part of responsibility in conceptual modeling is mindfulness to the normative functions which people – rightly or wrongly – may attribute to conceptual models.

3.3.4 The Role of Conceptual Models in the (Re-)Production of Social Reality

The effects of conceptual models can extend beyond the study of individual situations and the design of individual artifacts. Ultimately, conceptual models may affect even the *(re-)production of social reality* itself. By this we mean the processes through which human beings continuously construct the social structures, institutions, and norms which they then perceive as a given part of their everyday life, such as social conventions, formal and informal roles, public and private organizations, and societies at large. This is the fourth and final area of consideration considered here. Several roles and functions of conceptual models come to mind in this regard, two of which we shall cover in what follows.

One way in which conceptual models may influence what comes to be perceived as social reality is that they can subtly direct the *evolution of the language* of complete organizations or even larger spheres of public life. This is well illustrated by the example of an interesting reciprocal relationship between information systems development and natural language (for an analogous argument, see Frank and Bock 2020b, pp. 8–10). In the development of information systems, conceptual models are normally designed so as to reflect certain fragments of the *existing* language of the organization or the professional domain in question. For example, if a company produces certain types of manufacturing products, then analogous categories are likely to be captured in the data model of a new information system for that company. But conversely, conceptual models also usually contain concepts which, initially, have *no counterpart* in ordinary language. For example, standard Enterprise Resource Planning (ERP) system often define their own idiosyncratic concepts of deliverables, resources, and other objects of work. After the information systems have been deployed in the application domain, then, the users are routinely confronted both with the familiar and the new concepts. This often leads to a situation where the initially unfamiliar and artificial concepts of the software systems, over time, become part of the conventional language. One of the most striking examples of this phenomenon is the fact that the name of the best known internet search engine today is virtually synonymous with the very notion of searching the internet. The evolved natural language, then, will figure as the basis of other conceptual models for the design of still further new information systems in the future, thus giving rise to a cycle of reciprocal influence.

Since language is one of the chief mediums of expression of social reality, the capability of conceptual models to potentially alter it—even if only indirectly and gradually—deserves attention. It would appear appropriate to make oneself aware of this capability, and to attempt to preconceive in conceptual modeling the possible ramifications of the model under design on the evolution of the associated natural languages, be they those of a small organization or those of society at large.

A second way in which conceptual models may have a role in the (re-)production of social reality is that they can act as a means by which *social constructions* of varying natures themselves are perpetuated and reified. This can occur in many forms and at many levels of analysis. One of the simplest examples is the fact that a conceptual model of an information system, or, *a fortiori*, a conceptual model of an organizational structure, may (re-)define roles, positions, groups, and other units of formal organization. What is more, conceptual models cannot only capture these items, they also can help putting them into effect (see Section 3.3.3), for example, by incorporating them into the design of an accounting or management system. Since units of formal organization are directly associated with power, privileges, and rewards, thus, conceptual models can and do play a role in the distribution and allocation of these resources. Now the role of conceptual models in the (re-)production of social constructions is, of course, both obvious and intended when it comes to models of organizational structures and related subjects. But similar effects can occur in less overt, yet more momentous ways.

Many sociologists have drawn attention to the fact that social institutions often appear to have a real existence despite the fact they are wholly dependent upon human conduct. For example, Berger and Luckmann have noted the apparent 'paradox that [people are] capable of producing a world that [the individual] then experiences as something other than a human product' (Berger and Luckmann 1991, p. 78). There are, of course, countless theories of the (re-)construction of social reality, but one common theme is the idea that it involves a sort of recursive process. In socialization and throughout life, people acquire knowledge and language relating to certain institutions; this leads them to act in conformance with these institutions, thereby bringing them to life; and this, in turn, entails the reproduction of the same knowledge and the same language that gave rise to the belief in these institutions in the first place.[6] Knowledge and language, in their turn, can be passed on in many ways, but one of them, as underlined in the previous subsections, consists in the use of conceptual models. Thus, it is well to realize that whenever one devises a conceptual model of a social institution, social convention, or any other social construction, one thereby *perpetuates its (seeming) existence.* This is true of countless social institutions: family relationships, schools and universities, occupational groups and roles, national and international bodies, and many more.

To be sure, many times absolutely no objection need be made to the social institutions under study. But it is important to keep in mind that these institutions *need not be* as they are. And often, they have severe and drastic implications for the rights and responsibilities of individual persons. Many once established social conventions and institutions are now, for good reasons, considered untenable. Thus, we suggest that responsibility in conceptual modeling involves a constant awareness, and occasional questioning, of the social constructions whose existence is sustained by the conceptual model in design.

3.4 Conclusion

This chapter is intended to (re-)invigorate and contribute to the discussion of a subject which, we think, has not yet received quite the attention it deserves in conceptual modeling research. It is the systematic analysis of the ethical dimensions of conceptual models, and the attempt to develop an attendant conception of moral responsibility in conceptual modeling. The reasons why such a conception is needed are many, but, perhaps, the most fundamental is the one which Ulrich Frank never fails to emphasize: 'By focusing on conceptual models and modeling languages, IS research goes beyond [...] the design of possible future information systems,' as '[c]onceptual models also serve to illustrate possible *future action systems*' (Frank 2017, pp. 5730–5731, emphasis added). Conceptual models can be and are used to design what will become part of the world we live in, and, as such, it is well to reflect upon the principles and standards by which they are constructed.

The discussion in this chapter has concentrated on four areas: the contingency of the scope and content of models, the limits and biases of conceptual modeling languages, the intended and unintended normative functions of conceptual models, and the role of conceptual modeling in the (re-)production of social reality itself. For each of these areas, we have indicated some of the sources of difficulty, subtle factors of influence, and other complications surrounding the design and use of conceptual models which, although by no

6 For example, Berger and Luckmann (1991, p. 84) put it as follows: 'In its linguistic basis, this knowledge is already indispensable to the institutional 'programming' of these economic activities. [...] [T]he same body of knowledge is transmitted to the next generation. It is learned as objective truth in the course of socialization and thus internalized as subjective reality. This reality in turn has power to shape the individual.'

means unknown, are easily overlooked in practice. We suggest that the conclusions reached may be developed into a part of what might, eventually, become a body of principles and guidelines for responsibility in conceptual modeling.

Yet it is clear that much more work is needed if the field of conceptual modeling is to arrive at a defensible and sufficiently rich account of responsibility. There are at least three aspects to such a program. The first is to expand the analysis of the four areas outlined in this chapter. Although we have attempted to carve out certain points of significance, many of them invite further scrutiny, and others have not even been mentioned. For example, it is by no means easy to determine whether a blind spot of a conceptual model is a serious oversight or an expression of the model's intended analytical focus; and when thinking about the role of conceptual models in the (re-)production of social structures, one cannot afford to neglect the power that could be derived from the very objects of the models, such as the power that comes with a certain business model or an innovative technology.

A second area of future research may consist in the investigation of other major themes related to responsibility in conceptual modeling. For example, the present chapter has not touched upon the education and the pedagogy of conceptual modeling, the interaction and interpersonal relations between the group of modelers themselves, and the relation between conceptual modeling and artificial intelligence (which, in itself, is probably a near-inexhaustible source of ethical problems). Finally, the results obtained in the study of the aforementioned and other topics might be pulled together in an effort to formulate a system of principles for moral responsibility in conceptual modeling. Since such a body of norms would need to be concrete enough to inform practice, yet nuanced enough to reflect the depth and intricacy of moral theory, establishing anything of the sort is a formidable challenge to which all conceptual modeling researchers are invited.

References

Audi, R. (2010). *Epistemology: A Contemporary Introduction to the Theory of Knowledge*. 3rd. New York and London: Routledge.

Berger, P. L. and Luckmann, T. (1991). *The social construction of reality: A treatise in the sociology of knowledge*. Reprint. Harmondsworth: Penguin. ISBN: 0-140-13548-0.

Blackburn, S. (2005). *The Oxford Dictionary of Philosophy*. 2nd. Oxford and New York: Oxford University Press. ISBN: 0-19-861013-0.

Bock, A. C. (2016). 'How Modeling Language Shapes Decisions: Problem-Theoretical Arguments and Illustration of an Example Case'. In: *Enterprise, Business-Process and Information Systems Modeling*. Ed. by R. Schmidt, W. Guédria, I. Bider and S. Guerreiro. Lecture Notes in Business Information Processing. Berlin and Heidelberg: Springer, pp. 383–398. ISBN: 978-3-319-39428-2. DOI: 10.1007/978-3-319-39429-9{_}24.

Brodie, M. L., Mylopoulos, J. and Schmidt, J. W. (1984). 'Preface'. In: *On Conceptual Modelling*. Ed. by M. L. Brodie, J. Mylopoulos and J. W. Schmidt. New York: Springer, pp. v–viii. ISBN: 978-1-4612-9732-1.

Bunnin, N. and Yu, J. (2004). *The Blackwell Dictionary of Western Philosophy*. Malden, Oxford and Carlton: Blackwell. ISBN: 1-4051-0679-4.

Carroll, J. B., Levinson, S. C. and Lee, P., eds. (2012). *Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf*. 2nd. Cambridge: MIT Press. ISBN: 978-0-262-51775-1.

Clarke, R. (2010). 'Freedom and Responsibility'. In: *The Routledge Companion to Ethics.* Ed. by J. Skorupski. London: Routledge, pp. 263–274. ISBN: 978-0-415-41362-6.

Copp, D. (2006). 'Introduction: Metaethics and Normative Ethics'. In: *The Oxford Handbook of Ethical Theory.* Ed. by D. Copp. Oxford Handbooks in Philosophy. Oxford and New York: Oxford University Press, pp. 3–35. ISBN: 978-0-19-514779-7.

Crisp, R. (2000). 'Ethics'. In: *Concise Routledge Encyclopedia of Philosophy.* Ed. by Routledge. London and New York: Routledge, pp. 256–258. ISBN: 0-415-22364-4.

Driver, J. (2005). 'Normative Ethics'. In: *The Oxford Handbook of Contemporary Philosophy.* Ed. by F. Jackson and M. Smith. Oxford: Oxford University Press, pp. 31–62. ISBN: 0-19-924295-X.

Duff, R. A. (2000). 'Responsibility'. In: *Concise Routledge Encyclopedia of Philosophy.* Ed. by Routledge. London and New York: Routledge, pp. 768–769. ISBN: 0-415-22364-4.

Fischer, P. (2011). 'Selective Exposure, Decision Uncertainty, and Cognitive Economy: A New Theoretical Perspective on Confirmatory Information Search'. In: *Social and Personality Psychology Compass* 5.10, pp. 751–762. DOI: https://doi.org/10.1111/j.1751-9004.2011.00386.x.

Foglia, L. and Wilson, R. A. (2013). 'Embodied cognition'. In: *Wiley Interdisciplinary Reviews: Cognitive Science* 4.3, pp. 319–325.

France, R. and Rumpe, B. (2005). 'Domain specific modeling'. In: *Software & Systems Modeling* 4.1, pp. 1–3. ISSN: 1619-1366. DOI: 10.1007/s10270-005-0078-1.

Frank, U. (1999). 'Zur Verwendung formaler Sprachen in der Wirtschaftsinformatik: Notwendiges Merkmal eines wissenschaftlichen Anspruchs oder Ausdruck eines übertriebenen Szientismus?' In: *Wirtschaftsinformatik und Wissenschaftstheorie.* Ed. by J. Becker, W. König, R. Schütte, O. Wendt and S. Zelewski. Wiesbaden: Gabler, pp. 127–160. ISBN: 3-409-12002-5.

Frank, U. (2002). 'Multi-Perspective Enterprise Modeling (MEMO): Conceptual Framework and Modeling Languages'. In: *Proceedings of the 35th Hawaii International Conference on System Sciences.* Ed. by R. H. Sprague Jr. Los Alamitos: IEEE Computer Society. ISBN: 0-7695-1435-9.

Frank, U. (2009). 'Die Konstruktion möglicher Welten als Chance und Herausforderung der Wirtschaftsinformatik'. In: *Wissenschaftstheorie und gestaltungsorientierte Wirtschaftsinformatik.* Ed. by J. Becker, H. Krcmar and B. Niehaves. Heidelberg: Physica, pp. 161–173. ISBN: 978-3-7908-2335-6.

Frank, U. (2010). *Outline of a Method for Designing Domain-Specific Modelling Languages.* ICB Research Report 42. Essen: University of Duisburg-Essen.

Frank, U. (2011). *Multi-Perspective Enterprise Modelling: Background and Terminological Foundation.* ICB Research Report 46. Essen: University of Duisburg-Essen.

Frank, U. (2013). 'Domain-Specific Modeling Languages: Requirements Analysis and Design Guidelines'. In: *Domain Engineering.* Ed. by I. Reinhartz-Berger, A. Sturm, T. Clark, S. Cohen and J. Bettin. Heidelberg: Springer, pp. 133–157. ISBN: 978-3-642-36653-6.

Frank, U. (2014a). 'Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges'. In: *Software & Systems Modeling* 13.3, pp. 941–962.

Frank, U. (2014b). 'Multilevel Modeling: Toward a New Paradigm of Conceptual Modeling and Information Systems Design'. In: *Business & Information Systems Engineering* 6.6, pp. 319–337. ISSN: 1867-0202. DOI: 10.1007/s12599-014-0350-4.

Frank, U. (2017). 'Theories in the Light of Contingency and Change: Possible Future Worlds and Well-Grounded Hope as a Supplement to Truth'. In: *Proceedings of the 50th Hawaii*

International Conference on System Sciences (HICSS 2017). Los Alamitos: IEEE Computer Society, pp. 5727–5736.

Frank, U. (2022). 'Multi-level modeling: cornerstones of a rationale'. In: *Software & Systems Modeling* 21.2, pp. 451–480. issn: 1619-1366. doi: `10.1007/s10270-021-00955-1`.

Frank, U. and Bock, A. C. (2020a). 'Conjoint Analysis and Design of Business and IT: The Case for Multi-Perspective Enterprise Modeling'. In: *Advanced Digital Architectures for Model-Driven Adaptive Enterprises*. Ed. by V. Kulkarni, S. Reddy, T. Clark and B. S. Barn. Hershey: IGI Global, pp. 15–45. isbn: 978-1799801085.

Frank, U. and Bock, A. C. (2020b). *Organisationsforschung und Wirtschaftsinformatik: Zeit für eine Annäherung?* ICB Research Report 67. Essen: University of Duisburg-Essen. doi: `10.17185/duepublico/73339`.

Frank, U., Strecker, S., Fettke, P., Vom Brocke, J., Becker, J. and Sinz, E. J. (2014). 'The Research Field "Modeling Business Information Systems": Current Challenges and Elements of a Future Research Agenda'. In: *Business & Information Systems Engineering* 6.1, pp. 39–43. issn: 1867-0202. doi: `10.1007/s12599-013-0301-5`.

Gitlin, T. (1980). *The Whole World is Watching: Mass Media in the Making & Unmaking of the New Left*. Berkeley: University of California Press.

Glasersfeld, E. von (1995). *Radical Constructivism: A way of Knowing and Learning*. London and Washington: Falmer.

Goffman, E. (1974). *Frame Analysis: An Essay on the Organization of Experience*. Cambridge: Harvard University Press.

Gulden, J. (2004). 'Diagrammatizität und Bildhaftigkeit - zum Diagrammbegriff aus bildphilosophischer Perspektive'. Thesis for the Master of Arts in Philosophy. MA thesis. Technical University Berlin.

Hofstede, G. (1980). 'Culture and Organizations'. In: *International Studies of Management & Organization* 10.4, pp. 15–41.

Hofstede, G. (2011). 'Dimensionalizing Cultures: The Hofstede Model in Context'. In: *Online Readings in Psychology and Culture* 2.1, pp. 2307–0919.

Kant, I. (1996). 'Critique of Pure Reason'. In: *Immanuel Kant: Critique of Pure Reason*. Ed. by W. S. Pluhar. Indianapolis and Cambridge: Hackett, pp. 1–774. isbn: 0-87220-258-5.

Karagiannis, D., Mayr, H. C. and Mylopoulos, J., eds. (2016). *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*. Berlin, Heidelberg and New York: Springer. isbn: 978-3-319-39416-9.

Laurence, S. and Margolis, E. (1999). 'Concepts and Cognitive Science'. In: *Concepts*. Ed. by E. Margolis and S. Laurence. Cambridge: MIT Press, pp. 3–81. isbn: 0-262-63193-8.

Loucopoulos, P. (1992). 'Conceptual Modeling: Introduction to Section One'. In: *Conceptual Modeling, Databases, and CASE*. Ed. by P. Loucopoulos and R. Zicari. New York: Wiley, pp. 1–26. isbn: 0471554626.

Lucas, J. R. (1993). *Responsibility*. Oxford: Clarendon Press. isbn: 0-19-824008-2.

MacIntyre, A. C. (1966). *A Short History of Ethics: A history of moral philosophy from the Homeric age to the twentieth century*. London: Routledge & Kegan Paul and Macmillan. isbn: 0-7100-1775-8.

Maturana, H. R. and Varela, F. J. (1992). *The Tree of Knowledge: The Biological Roots of human Understanding*. Revised. Boston: Shambhala. isbn: 0-87773-642-1.

Murphy, G. L. (2004). *The Big Book of Concepts*. Cambridge and London: MIT Press. isbn: 978-0-262-63299-7.

Object Management Group (2011a). *Business Process Model and Notation (BPMN): Version 2.0.* OMG Document formal/2011-01-03. URL: `http://www.omg.org/spec/BPMN/2.0/` (visited on 10/07/2013).

Object Management Group (2011b). *OMG Unified Modeling Language (OMG UML), Infrastructure: Version 2.4.1.* OMG Document formal/2011-08-05. URL: `http://www.omg.org/spec/UML/2.4.1/` (visited on 21/03/2013).

Oldroyd, D. (1986). *The Arch of Knowledge: An Introduction to the History of the Philosophy and Methodology of Science.* New York and London: Methuen. ISBN: 0-416-01331-7.

Regier, T. and Kay, P. (2009). 'Language, thought, and color: Whorf was half right'. In: *Trends in Cognitive Sciences* 13.10, pp. 439–446.

Rolland, C. and Cauvet, C. (1992). 'Trends and Perspectives in Conceptual Modeling'. In: *Conceptual Modeling, Databases, and CASE.* Ed. by P. Loucopoulos and R. Zicari. New York: Wiley, pp. 27–48. ISBN: 0471554626.

Sandkuhl, K., Stirna, J., Persson, A. and Wißotzki, M. (2014). *Enterprise Modeling: Tackling Business Challenges with the 4EM Method.* Berlin, Heidelberg: Springer. ISBN: 978-3-662-43724-7.

Schwartz, S. H. (2012). 'An overview of the Schwartz theory of basic values'. In: *Online readings in Psychology and Culture* 2.1, pp. 2307–0919.

Simon, H. A. (1957). *Models of Man: Mathematical Essays on Rational Human Behavior in a Social Setting.* New York: Wiley.

Tversky, A. and Kahneman, D. (1974). 'Judgment under Uncertainty: Heuristics and Biases'. In: *Science* 185.4157, pp. 1124–1131. DOI: `10.1126/science.185.4157.1124`.

Tversky, A. and Kahneman, D. (1981). 'The Framing of Decisions and the Psychology of Choice'. In: *Science* 211.4481, pp. 453–458. DOI: `10.1126/science.7455683`.

Uniacke, S. (2010). 'Responsibility: Intention and Consequence'. In: *The Routledge Companion to Ethics.* Ed. by J. Skorupski. London: Routledge, pp. 596–606. ISBN: 978-0-415-41362-6.

Vernadat, F. B. (2000). 'Enterprise Modeling and Integration: Current Status and Research Perspectives'. In: *IFAC Proceedings Volumes* 33.17, pp. 1–9. ISSN: 14746670. DOI: `10.1016/S1474-6670(17)39366-7`.

Wittgenstein, L. (1922). *Tractatus Logico-Philosophicus.* New York: Harcout, Brace & Company.

Wittgenstein, L. (1958). *Philosophical Investigations: Translated by G. E. M. Anscombe.* Oxford: Basil Blackwell.

Wolff, F. and Frank, U. (2005). 'A Multi-Perspective Framework for Evaluating Conceptual Models in Organizational Change'. In: *ECIS 2005 Proceedings.* Paper 147.

Wolff, P. and Holmes, K. J. (2011). 'Linguistic relativity'. In: *WIREs Cognitive Science* 2.3, pp. 253–265. DOI: `10.1002/wcs.104`.

Wolters, G. (2010). 'kontingent/Kontingenz'. In: *Enzyklopädie Philosophie und Wissenschaftstheorie.* Ed. by J. Mittelstraß. Stuttgart and Weimar: Metzler, pp. 324–325. ISBN: 978-3-476-02108-3.

Zachman, J. A. (1987). 'A framework for information systems architecture'. In: *IBM Systems Journal* 26.3, pp. 276–292.

Chapter 4

Conceptual Modeling: A Still Unfinished Saga

About Prejudices, Aberrations, Solutions and Challenges

Heinrich C. Mayr and Bernhard Thalheim

Conceptual modeling has a long and chequered history that goes far back into the past. In Computer Science, it has been discussed since the 70s of the last century, when – after some preceding work on data and process abstractions – the fundamental paper by Peter P.-S. Chen appeared in which he introduced the Entity-Relationship Model. In honor of the 65th birthday of Ulrich Frank who throughout his career has been deeply and also critically engaged with conceptual modeling, we present in this paper a few observations and thoughts on this history.

4.1 Introduction

This paper is intended as a little thank-you for Ulrich Frank, who has over many years decisively shaped the modeling scene, especially in Business Informatics, and surely will continue to do so. We would like to point out right away, however, that this is not a thoroughly serious, purely scientific paper. It is rather a collection of observations, questions, theses, and examples on the state of modeling which we have dealt with in the course of our many years of involvement with modeling ans partly also discussed in keynote speeches at various occasions. Accordingly, we make no claim to completeness: There are many things we do not mention, for example because we do not know anything about them, because we did not think they were worth mentioning, or because we simply forgot about them. Ulrich's 65th birthday, on which we extend our sincere congratulations, is a welcome occasion to present this thank-you, as he has contributed significantly to a better understanding and foundation of modeling, especially through his critical analyses but also with his constructive concepts. Modeling and modeling methods are crucial to most disciplines. We cannot do without them, even if they are not always appreciated, often underestimated, and in practice often dismissed with '*it's useless*'. After all, modeling has been widely researched, taught, and also practiced with a certain systematic for many years. However, many concepts are regularly 'reinvented' or renamed, many mistakes are made again and again, and prejudices are readily cultivated.

But there is progress: both in the theoretical foundation and in the realization of what is needed for and in practice. We will sketch a picture of this here – with a focus on more recent developments. Of course, this cannot be done without some recourse to the gray

Middle Ages of the early Computer Science years and a sketch of the essence of modeling and its dimensions.

The paper is structured as follows: We first address some basic work that interested people either already know or can or should read up on. Then we try to answer the question '*Where do we stand?*' and contrast this with the situation in 1998, the year of the foundation of the German-speaking Modeling Conference, in which Ulrich Frank always played an important role. From this, we try to derive some hints for the future and shortly explain our contribution to the foundation of conceptual modeling by means of the Triptych paradigm (Mayr and Thalheim 2021).

4.2 Basic Work

Every discipline has its own foundational works – though, of course, opinions often vary widely about what is important and what is not. So, in order to not only out our own assessments, we reproduce here a list of works identified in a survey[1] conducted in 2021: This survey was distributed worldwide among representatives of the data, process, and software modeling communities and asked, among other things, what participants thought was the most important foundational work in their field.

A total of 153 participated in this survey and mentioned the following (at least twice each, in alphabetical order by first author):

- Batini, C., Ceri, S., Navathe, S. B. (1992). *Conceptual Database Design: An Entity-Relationship Approach* (Batini et al. 1992)
  The first comprehensive and systematic presentation of the ER methodology of database modeling for the development of database structures.
- Brambilla, M., Cabot, J., Wimmer, M. (2017). *Model-driven software engineering in practice* (Brambilla et al. 2017)
  The introductory book follows Stachowiak (1973) in an object-oriented setting and discusses the foundations of model-driven software engineering as well as its technical aspects.
- Broy, M., Stølen, K. (2012). *Specification and development of interactive systems: focus on streams, interfaces, and refinement* (Broy and Stølen 2001)
  A systematic and well-founded explanation for stepwise specification and development of interactive software and systems as programming in the large (FOCUS message interaction method).
- Carmona, J., van Dongen, B., Solti, A., Weidlich, M. (2018). Conformance checking (Carmona et al. 2018)
  The book investigates the process mining perspective of the relation between modelled behaviour and recorded behaviour of processes.
- Chen, P.P.S. (1976). *The entity-relationship model – toward a unified view of data* (Chen 1976) (reprinted in Embley and Thalheim 2012)
  The initial paper that introduces the notion of conceptual model(ing language) as documentation model for implementation-oriented relational database schemata, further used for the description modeling approach.

1 Michael, J.; Bork, D; Wimmer, M.; Mayr, H. C.: Quo Vadis Modeling? Findings of a Community Survey, an Ad-hoc Bibliometric Analysis, and Expert Interviews on Data, Process, and Software Modeling. To appear in the International Journal on Software and Systems Modeling (SoSym), Springer.

- Codd, E.F. (2002). *A relational model of data for large shared data banks* (Codd 2002)
  The initial paper on the relational approach to database structure everybody in the area should know.
- Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A. (2013). *Fundamentals of business process management* (Dumas et al. 2013)
  The book distills the entire business process management landscape.
- Elmasri, R., Navathe, S.B. (2000). *Fundamentals of Database Systems* (Elmasri and Navathe 2000)
  One of the main textbook for database analysis, design, and development based on the classical entity-relationship approach.
- Embley, D. W., Thalheim, B. (Eds.). (2012). *Handbook of conceptual modeling: theory, practice, and research challenges* (Embley and Thalheim 2012)
  A survey collection on different approaches, languages, and foundations to conceptual modeling.
- Ferstl, O. K., Sinz, E. J. (2015). *Grundlagen der Wirtschaftsinformatik* (Ferstl and Sinz 2015)
  The systematics of business information systems as the core of Business Informatics is systematically described for the level of tasks, the carriers of these systems and the design and operation.
- Friedman, J. H. (2017). *The elements of statistical learning: Data mining, inference, and prediction* (Friedman 2017)
  A systematic presentation of the statistical basis and algorithmics of systematic learning from data.
- Gamma, E., Johnson, R., Helm, R., Johnson, R. E., Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software* (Gamma et al. 1994)
  One of the first systematic representations of generic object-oriented programming solutions as classes of pattern.
- Guarino, N. (1994). *The ontological level* (Guarino 1994)
  Ontologies should be used for improving the quality of knowledge bases.
- Halpin, T., Morgan, T. (2010). *Information modeling and relational databases* (Halpin and Morgan 2010)
  A fully developed representation of the fact-oriented NIAM/ORM approach to conceptual modeling of database structures.
- Karagiannis, D., Mayr, H.C., Mylopoulos, J. (2016). *Domain-specific conceptual modeling* (Karagiannis et al. 2016)
  A survey collection on various languages and approaches for conceptual modeling that can be used for model development within the ADOxx realisation toolbox.
- Kent, W. (1978). *Data and reality* (Kent 1978)
  The classical book explains how human beings perceive and process information about the world we operate in, and how we struggle to impose that view on our data processing machines in whatever modeling language.
- Olivé, A. (2007). *Conceptual modeling of information systems* (Olivé 2007)
  A systematic presentation of the object-oriented approach to modeling database structures.
- Stachowiak, H. (1973). *Allgemeine Modelltheorie* (Stachowiak 1973)
  One of the most fundamental – but internationally little noticed – books on modeling with three properties characterising models: mapping, abstraction, pragmatic.
- Thalheim, B. (2000). *Entity-relationship modeling: foundations of database technology* (Thalheim 2000)
  A systematic compilation of achievements, extensions, and theory of entity-relationship modeling languages and their applications.

- Weske, M. (2007). *BPM – Concepts, Languages, Architectures* (Weske et al. 2007)
  A systematic explanation of the complete business process management lifecycle from the modeling phase to process enactment and improvement.
- van der Aalst, W.: *Process Mining – Data Science in Action* (Van Der Aalst 2016)
  Compiling approaches to process mining as the missing link between model-based process analysis and data-oriented analysis techniques.

In our personal opinion, however, at least the following works should also be mentioned:

- Becker, J., Probandt W., Vering, O. (2012). *Grundsätze ordnungsmäßiger Modellierung - Konzeption und Praxisbeispiel für ein effizientes Prozeßmanagement* (Becker et al. 2012)
  Principles of proper models are correctness, relevance, economic efficiency, clarity, comparability, and a systematic structure.
- Cabot, J., Vallecillo, A. (2022). *Modeling should be an independent scientific discipline* (Cabot and Vallecillo 2022)
  A vision according to which modeling is to be developed into an independent sub-discipline within computer science with high disciplinary gains.
- Frank, U. (2014). *Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges* (Frank 2014)
  Multilevel multi-perspective modeling is a powerful object-oriented framework for coherent co-development of models at various abstraction levels (meta …meta-meta-meta) with integrated perspectives (design, engineering, development, user, economic, enterprise).
- Guarino, N., Guizzardi, G., Mylopoulos, J.(2019): *On philosophical foundations of conceptual Models* (Guarino et al. 2019)
  The paper proposes an ontology-backed general grounding to conceptual modeling.
- Henderson-Sellers, B. (2015). *Why Philosophize; Why not Just Model?* (Henderson-Sellers 2015)
  Enhancement of the MOF metamodel approach by placing another ontological or definitional layer between the M1 model and the M2 meta-model layer.
- Ludewig, J. (2002). *Modelle im Software Engineering - eine Einführung und Kritik* (Ludewig 2002)
  The key fundamental concepts of modeling are critiqued based on their potential for meaningful use.
- Mahr B. (2015). *Modelle und ihre Befragbarkeit – Grundlagen einer allgemeinen Modelltheorie* (Mahr 2015)
  The generalization, systematization, and formalization of the approaches of H. Stachowiak and of the Berlin 20xy circle for the model-being presented in an axiomatic form.
- Moody, D. (2009). *The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations* (Moody 2009)
  Develops a set of principles (semiotic clarity, perceptual discriminability, semantic transparency, manageable complexity, cognitive integration, expressiveness, dual coding, economy, cognitive fit) for designing cognitively effective visual notations.

And in all immodesty

- Mayr, H.C., Thalheim, B. (2021) *The Triptych of Conceptual Modeling - A framework for a better understanding of conceptual modeling* (Mayr and Thalheim 2021)
  Develops a foundation for conceptual modeling based on three dimensions: model dimension, instrumentation dimension (e. g., language dimension), and meaning or integrateable knowledge dimension.

4.3 Where Do We Stand

One should think that with all these intelligent works, hardly any fundamental questions remain open. Especially since there are many more publications on modeling. For example, in the proceedings of the three main modeling conferences ER (Int. Conference on Conceptual Modeling), MODELS (Int. Conference on Model Driven Engineering Languages and Systems) and BPM (Int. Conference on Business Process Management) alone, more than 3,400 scientific papers by more than 4600 authors have been published since their respective existence, not including workshops, all other conferences and journals. So there is obviously a huge body of knowledge. And yet central questions are still open. For example, '*When is a model a conceptual one?*' (although Mayr and Thalheim 2021 make a proposal for this). Given the many modeling languages and tools offered as 'conceptual', such questions may seem quixotic and academic to practitioners. However, a scientist should be concerned if he or she researches and teaches in the field of modeling but cannot give an universally valid or at least broadly accepted answer to this question. We will now try to identify some reasons for these deficits.

4.3.1 Model: A Working Definition

Before we delve further into the world of modeling and its stumbling blocks here, we must at least briefly explain what we understand as a '*model*' – because even on this there are many different opinions and definitions. Here we go beyond the definition presented in Mayr and Thalheim (2021) and Mayr and Thalheim (2022) by distinguishing between the terms '*model*' and '*mental model*'. According to Mahr (2021) and theories of Cognitive Science, a mental model is a component of a person's *cognitive structure* (Forstmann and Wagenmakers 2015; Haier 2016) consisting of *ideas* and *judgments* which has at least the following characteristics:

1. Relation to an origin: *'A model is a model of something'.*
2. Concern and usage: e. g., understanding, communicating, agreeing.
3. Purpose and function, e. g., (a) descriptive: analyze, assess, explain, (b) prescriptive: plan, design, (c) explorative: search, predict.
4. Domain: the modeled 'perceived world'.
5. Context: the relevant personal, environmental, social and spatio-temporal circumstances.
6. Focus: the relevant aspects of the origin(s) for the given purpose.

Figure 4.1 shows a very simplified representation of our underlying conceptual framework: we first consider the derivation of the term 'mental model': It consists, as we think, of 'mental objects' that humans form with the help of 'cognitive processe' and integrate into the 'cognitive structure' of their brain (not shown here). According to model characteristic 1, there is an origin, i. e., an object in the perceivable world.

In order to be able to communicate about ideas (i. e., their mental objects) with others, humans designate them with physically recognizable symbols, for example, of textual or graphic kind, sound waves (e. g., spoken or sung words or texts), sequences of light or even smoke signals as they are said to have been used by Indians. We use the umbrella denomination 'description' for these symbols and assume that they basically belong to a

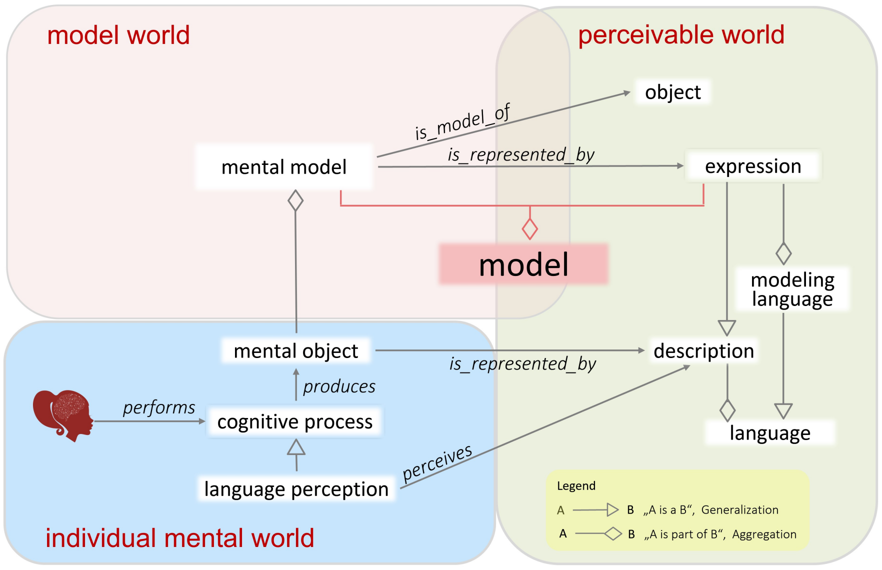


Figure 4.1: About the model concept

kind of (natural, i. e., evolved) language in a very broad sense (see Figure 4.1: descriptions are part of a language and represent mental objects).

In order to infer from a perceived description its meaning (i. e., the represented mental object), humans have the ability of 'language perception' (Tillmann 2012), with which they can understand, for example, the signal of a police siren not only as a sequence of tones but as a warning, i. e., language perception again is a kind of a cognitive process.

This allows us now to sharpen the notion of 'model' itself: We see it as an entity consisting of a mental model and an 'expression' representing it. We use 'expression' here (as a specialization of 'description'), to express that it is a more formal construct belonging to a modeling language. For more details and in particular the modeling and representation rule systems underlying model building we refer (Mayr and Thalheim 2021).

Modeling languages are used to simplify communication about models by enabling the communication partners to develop as similar an idea as possible about a model object. The simplest form are technical languages whose elements are explained in natural language. More sophisticated are formal languages. They consist of a set of modeling concepts, usually defined in terms of a metamodel, and a representation framework with notation and syntactic rules. The BPMN[2] can be mentioned here as an example.

In practice, the word 'model' is often also used for its representation (i. e., the term representing it). Colloquially, this is fine, e. g., when one speaks of a model car, a model train, the prototype of a product, or a 'model' that is supposed to give the viewer an idea of how a fancy dress or suit would look on him. Therefore, we will also practice this homonymic usage in what follows, specifying more precisely only when it might be unclear to which meaning we are referring.

2 https://www.omg.org/spec/BPMN/

4.3.2 Modeling, the Computer Science Cinderella

On the basis of the preceding clarification of terms, we can assume that human beings are continuously creating models (they are ‘modeling’), since they permanently form ideas about phenomena that they perceive through their senses and form judgments about them. Mostly, however, they does this unconsciously – not only in everyday life but also in professional activities.

Consciously dealing with modeling, i. e., defining modeling languages and using them consistently, on the other hand, is not everyone’s cup of tea. In engineering, and especially where life-critical systems are concerned, this is a matter of course. No electrical engineer, for example, would think of building a large electrical system without first designing and analyzing a model, e. g., an electronic circuit diagram. Accordingly, modeling and simulation play an important role in engineering education.

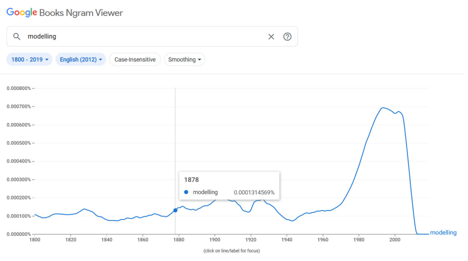
In Computer Science, surprisingly, this applies only to partial areas: Especially in software business practice, one often encounters the opinion that the effort for systematic modeling does not pay off, that the customer does not pay for it, or that the productivity or the effort of modeling cannot be measured ‘in lines of code’ (Ludewig 2002) like with programs, although other metrics, e. g., the function point method (Garmus and Herron 2001), could easily be adopted. Often also, the object under development is estimated so complex that it cannot be described with a modeling language, but must be programmed right away. This is of course a contradiction in terms. For example, programs are representations of ideas about processes which are to be processed by a computer. They are, therefore, representations of models, in short: *Programming is modeling*! Accordingly, requirements engineering also consists to a large extent of modeling, which is actually recognized in practice. Unfortunately, however, this is only true to the extent that requirements models are usually lost sight of during development: Necessary adjustments are no longer made in the model, so that the original design document is not a valid model of the end product.

Researchers live under the mistaken belief that description, and thus descriptive models, are the central modeling task in programming. Practitioners, however, need models as building templates, i. e., they need prescriptive models. And such models should not only reflect the end result but also show how one came to the model, why one chose this way and with what justification this way is good. Data analysis, for example, works with hidden and sometimes deliberately burrowed assumptions about the approach, the methods, the data and the tools. In contrast, database scientists have had to learn painfully how important metadata with its many facets is.

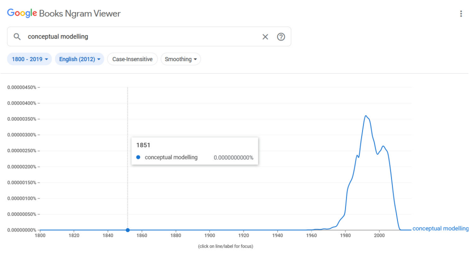
The two NGrams in Figure 4.2 show that there has been a general downward trend in modeling and conceptual modeling over the last two decades. And this despite the challenges posed by the breadth and diversity of applications. However, that might not be the whole truth, because models are now appearing under different names, for example as ‘digital twins’. In the next subsection, we will address some reasons that we believe have led to this shadowy existence of modeling.

4.3.3 What are the Problems?

In our opinion, the problems of modeling can be an be illustrated by three observations which we will explain below:



(a) Modeling (−24 per cent since 1992)



(b) Conceptual modeling (−43 per cent since 2005)

Figure 4.2: The seemingly waning interest in (conceptual) modeling

- Observation 1: *Entropy* (Ben-Naim 2007) is growing in the domain of modeling, i. e., substantial efforts ('energy') are necessary to put things in order.
- Observation 2: The *law of logistic growth* (Verhulst 1838) is valid for modeling methods, i. e., they start with a manageable set of instruments, grow stronger and stronger until the turning point, which heralds the decay.
- Observation 3: Fundamental questions are repeatedly reformulated, twisted and turned by the community, but not conclusively answered.

4.3.4 Entropy

Clues to which entropy in the Informatics field of modeling research can be traced are

- an unmanageable variety of methods and tools which are mostly not 'interoperable',
- a sparse methodology,
- fuzzy terminology.

There are a few methods that have become a certain standard over time and have endured, e. g., UML, ER modeling, Petri nets or BPMN. Apart from this, however, the review of relevant literature suggests that the pressure for 'innovation' at scientific institutions leads to a *continuous stream of new approaches or variants of existing ones* which differ from each other only in nuances but by varying wording and graphical notations. The 'not invented here' syndrome seems to prevail, according to which I prefer to invent my own (insufficient) method before helping to improve an insufficient method of someone else. Even the 'standard methods' mentioned above are not immune to this problem: new variants have been and are being introduced again and again, but they mostly burn out quickly like fireflies. Even worse, sometimes what is already known is recycled under a new name, or reinvented, because the publications in question date back too far (and are thus possibly considered out-of-date), or are from a related field. A good example of this is the OMG Metaobject Facility MOF[3] which is referenced in practice today in the context of metamodeling. The abstraction levels of MOF, however, correspond (just in inverse numbering) to the layers of the ANSI standard X3.138-1988 for information resource dictionary systems (IRDS) (Parker 1992). This is not indicated in MOF. All this tends to confuse practitioners rather than make them enthusiastically embrace them as the method inventors would like to see. So it is no

3 `https://www.omg.org/spec/MOF/`

wonder that in business practice, instead of any modeling tool, visualization programs such as Visio™ are used to represent models.

Another reason for the reluctance of practitioners, however, is probably that inventors of new modeling methods are usually happy with presenting the method's conception including the metamodel, the notation (and thus the language), and a corresponding software tool. However, as was recognized very early on, this is not enough to make a method practicable. Rather it requires also instructions, i. e., a *'proceeding model'* (Kaschek and Mayr 1976), how the language is to be used, for example:

- hints which language expressions are to be preferred or rejected (a 'style guide'),
- rules for applying the style appropriate for a particular context, and
- construction rules for the efficient production of expressions of good style.

Modeling language, tool and proceeding model together with a set of goal templates (for deriving the goals of a particular usage) then constitute what we mean by a *'modeling methodology'* (Jaakkola and Thalheim 2011). The provision of such modeling methodologies could form the basis of a *'modeling lore'* or 'modeling theory', which we believe is urgently needed and is taken for granted in other engineering disciplines as 'design theory'.

In our opinion, the lack of a modeling theory is also reflected in the almost Babylonian variety and vagueness of terms: Hard definitions of the form *'A is a ...'* are rarely found, mostly paraphrases are used like *'A is characterized by ...'* or 'A *is comprised of ...'*. A very recent example of this can be found in : *'Conceptual models are comprised of constructs, such as entities, events, goals, attributes, relationships, roles, and processes, connected by well-defined rules'.* Of course, such attempts of explanation have their justification in informal settings, but they lead to fuzzy semantics and thus inconsistent interpretations. To give some samples:

- still, the terms 'conceptional model' and 'conceptual model' are used synonymously,
- the term 'parallelism' is used where 'concurrency' is meant,
- the term 'role' (as an aspect of associations) is used very differently,
- the situation is similar for the term 'attribute',
- for cardinalities and multiplicities there are many variants and different notations,
- foreign words are used incorrectly and blurred, such as 'semiformal' or 'semi structured'. One talks for example of an 'semiformal language' if it is about an informal one, which may have a few construction rules. These seem to be magic words of Computer Science (you hardly find them in Mathematics or technical sciences) if something cannot be grasped exactly.

4.3.5 Logistic Growth

In the middle of the seventies, Teichroew and Hershey (1976) introduced the *'Program Statement Language'* and the *'Program Statement Analyzer'* (PSL/PSA) within the framework of the 'ISDOS' project (Information System Design and Optimization System). The focus was on requirements modeling and requirements analysis. At the beginning, the language was lean and transparent, but in the course of time new concepts were added, so that later 19 different modeling concepts for 'objects' and no less than 102 different relation types were distinguished. Originally, this method was quite used, especially in practice, but with the growth, transparency, maintenance, and availability problems seem to have arisen, so that today there is still a reference website [4], but the status is unclear.

The *'Structured Analysis and Design Technique', SADT* (Ross and Schoman 1977) developed by D.T. Ross and his company SofTech at the end of the sixties had a similar history. Initially, it came with only a few elements and a transparent graphical and hierarchically structured notation, but then was continuously extended. In 1981, the language IDEF0 based on it was published as part of the IDEFxx family of modeling languages in the field of Software Engineering, and in 1983 it was registered by the National Institute of Standards and Technology as a Federal Information Processing Standard (FIPS). In 2008, this standard was withdrawn. SADT was also the subject of academic teaching for a long time, but today it is hardly known. Related and caught by a similar fate is also the method *'Structured Analysis and Systems Specification'* (DeMarco 1978) presented by Tom de Marco at the end of the 1970s for which later IDEF could also be used.

A more recent example is provided by the *'Unified Modeling Language UML'* [5]. Originating from the merging and unification of various predecessors (methods of 'Object Oriented Analysis'), in 1996 it provided 5 different model views, which were represented by so-called *diagrams* (Use Case, Class, Sequence, Collaboration, State). In 2012 (UML 2.5), there were already 18 official and 7 unofficial diagrams. It is therefore no wonder that in corporate practice few people seem to have a comprehensive and deep understanding of the overall concept and, according to our experience at least with small and medium-sized software manufacturers, there is no consistent use. So the hype has softened and it remains to be seen when the next, simpler and more transparent method will be invented.

Incidentally, this up and down of methods could also be described quite well with the so-called Kondratjev cycles (Metz 2006), i. e., with the theory of long waves: The starting point for such long waves are paradigm shifts and the associated massive investments in innovation. Once an innovation has become generally accepted, investments decrease and a downturn occurs. During the downturn, however, work is already underway on a new paradigm.

4.3.6 Fundamental Questions

In this subsection, we give a few quotations from former times by which one can recognize that fundamental questions are not completely answered until today. The preface to the proceedings of the 1979 founding event of today's SIG EMISA in the Gesellschaft für Informatik GI states: *'The constantly growing complexity of operational and socio-technical information systems today touches the limits of our planning capacity. A formally sound set of tools for the planning, design and operation of such systems is therefore absolutely necessary. For example, methods for determining the requirements to be placed on an information system are needed, as are formal modeling concepts for its description, specification, and analysis, and finally techniques for evaluating its behavior'.* Note that the term 'formal' is used here, not 'semi-formal'.

In a keynote address to the first German Modeling Conference 'Modellierung' held in Münster in 1998, it was stated, among other things:

4 https://requirementsanalytics.com/index.php/consulting-and-training/pslpsa
5 https://www.omg.org/spec/UML/2.5.1

- *'Quantitative aspects associated with performance requirements or security aspects are still hardly covered by conceptual terms. For the developer, however, they are of great importance, a design document should therefore show them'.* and
- *'Another problem still lies in the abstraction distance between natural language requirements specifications and conceptual designs. Although almost every inventor of a conceptual modeling method (...) has claimed that his approach provides a suitable basis for understanding between system analysts/developers. However, practice shows that conceptual models are often often intransparent for the users and thus cannot be sufficiently validated by them'.*
- and finally: *'In conclusion, it can be stated that the field of operational information systems modeling is developing slowly but steadily. The basic framework of generally accepted concepts and terminology is growing, but we are still far away from a design methodology or even design theory'.*

In the foreword to EMISA Forum 1/98, Helmut Thoma writes: *'Modeling is still treated stepmotherly (in practice); This after decades of desperate struggle to put thinking and constituting before realizing'.* If one now goes through the list of topics on which papers are solicited in the Call for Papers of the ER Conferences of the last 10 years, one finds among others regularly (partly in different wording):

- Foundations and Concept Formalization
- Quality and Metrics of Conceptual Models
- Data Semantics and Integrity Constraints
- Interleaving Modeling and Development

The CFP for the upcoming 2023 ER also reads: *'Submissions that lead to new foundations, links, applications, or enlarge current boundaries of conceptual modeling are especially welcome'.* The MODELS conference 2023 is also looking for contributions to, among other things, *'Fundamentals of model-based engineering, including the definition of syntax and semantics of modeling languages and model transformation languages'* or *'Development of model-based systems engineering approaches and modeling-in-the-large, including interdisciplinary engineering and coordination'.*

To sum up: the set of instruments claimed since the 70ies and its foundation still exists only in rudimentary form. Consolidation has not yet taken place, so that academic teaching and practical training are also rather superficial: the languages that are in fashion at the moment are taught, but the theoretical foundations are rarely studied in depth. Moreover, the modeling languages do not keep pace with the technological development either. For example, new paradigms would have to be found and new techniques developed for Big Data and Data Analytics. Moreover, an automatic translation of models into code exists only in rudiments. Consequently, there is a mountain of unfinished homework from theory and practice of (conceptual) modeling, i. e., we face a lot of challenges that we need to address if modeling is not to continue to lose its importance.

4.4 Opportunities for Progress

The weaknesses pointed out in the previous chapter seem to have shaken up the community a bit. In any case, recently there have been an increasing number of contributions dealing with the state of the discipline, (e. g., Castellanos et al. 2018; Lima et al. 2020; Recker et al. 2020; Storey et al. 2023), foundations (e. g., Yu 2009; Guizzardi et al. 2015; Guarino et al. 2019; Mayr and Thalheim 2021), focal points and corresponding communities (e. g., Sandkuhl et al. 2018; Khatri and Samuel 2019; Lukyanenko et al. 2019; Eriksson et al. 2019), or even trying to give starting points for the characterization of research in conceptual modeling (Delcambre et al. 2021). So, we seem to be heading where the engineers already are: They have learned to develop and use blueprints, schematics, etc., which gives them great advantages regarding quality and productivity. Therefore, it is impossible to imagine engineering without models: the Cinderella is a beauty there - and should become so now in Computer Science. Accomplishments that suggest this but have yet to be worked through in disciplinary terms include the following:

- Even if the term 'model' is not the focus, requirements engineering is actually modeling. 'Brownfield' development and system evolution cannot exist without models. Web information systems design and development is model-backed. User-centered development of systems must also integrate models of users and usage.
- Large and complicated systems are created with action instructions, i. e., activity or 'proceeding' models, respectivels, possibly even with a waterfall architecture. Many methodologies have emerged that await generalization but have already resulted in low-maintenance systems.
- Process engineering, process elicitation, and process elaboration led to powerful tools and an improvement of business support.
- Systematic and thoughtful programming is always accompanied by and supported by models. Object-oriented and object-relational approaches demonstrate the silver bullet of stepwise specification for programming in the large.
- Despite the huge variety of modeling languages, systematic domain-specific design of languages is becoming acknowledged and more widely used.
- Generic, pattern-based, and reference solutions are the basis for macro- and meso-models as a reliable starting point instead of 'greenfield' development from scratch.
- Database and information systems development and design can not be imagined without models and modeling. User viewpoints are essentially derived models.
- Models are going to be used as prescription of coding. Models can be translated to code fragments as kernel elements of software.
- Quality assessment and improvement of models is becoming a common technique.
- 'Semantification' and conceptualisation results in models that are meaningful to all stakeholders. Ontology engineering is becoming a technique for semantic foundation and sense-making elaboration.
- Due the the large body of knowledge on models, modeling and model usage, claims are raised that modeling is becoming a sub-discipline and modeling is essentially the fourth dimension of Computer Science.

This marks the path from modeling as a luxury to a daily, deliberate, and purposeful activity. It is rumored from IBM's main office that there was a saying on the wall behind the CEO, 'Think, think, think, before doing anything' but then it was obscured by a cabinet so that only the first half was seen. Together it adds up, however: we are not rich enough to invent everything over and over again, sometimes even worse.

4.5 The Triptych Foundation of Conceptual Models

At the end of Section 4.2, we noted our contribution (Mayr and Thalheim 2021) to the foundations of conceptual modeling, which we had laid out as *'as a contribution to the "anatomy" of conceptual models'* and with which we intended to help better understand the nature of conceptual modeling. In particular, with the explanatory paradigm of the triptych we especially want to give a starting point for conceiving conceptual models as a special kind of models and to have a clear distinguishing criterion, i. e., we want to be able to say whether a given model is a conceptual one or not. This would solve one of the major deficits (see introduction to Section 4.3) in our field.

The basis of our considerations is that conceptual modeling consists of three parts: (1) a central model part, (2) the instrumentation by a language, and (3) the inherited and selected world semantics within the given application. The triptych paradigm illustrates these three parts through its three dimensions (wings):

- The *linguistic dimension* of language representation, where 'language' here is quite broad and can include textual, graphical, audio-visual, biosemiotic, and other physical forms of representation. In the case of well-known modeling languages such as UML, ER, or BPMN, it is the respective notation together with the grammatical composition rules. Languages enable and hinder, have to match with necessities in applications, have to be evolve together with technology, and have to reflect the accepted culture in the application domain.
- The *model dimension* defines the structure of the models at different levels of abstraction, forming a hierarchy via stepwise intension/extension relationships: Each level provides modeling concepts (elements, relationships, etc.) that can be used at the next level for model building. Usually four levels are considered (Metametamodel, Metamodel, Model, Individuals), but in general there can be any number of levels, e. g., if on the individuals level again concepts are offered, to which there are extensions[6]. Consider, as an example, the original entity-relationship model: As a metamodel it includes modeling concepts like 'Entity Set', 'Relationship Set', 'Attribute', 'Value Set', 'Role'. For a concrete application, extensions can be derived, e. g., an entity set 'Customer' to which certain attributes such as have extensions in the form of individuals (models of concrete objects of a considered mini-world), for example the 'customer' *Smith.*

  By using the metamodel as a 'controlled vocabulary', i. e., by using only words from this pool, and by using natural language identifiers at the various levels, a so-called 'a priori semantics' is created, i. e., the models and their extensions are interpretable on the basis of the meanings of the natural language designators used, but not necessarily unambiguous. Keep in mind that these designators correspond to the 'expressions' in the 'perceivable world' (see Figure 4.1), with which together mental models become models. It should be obvious that the linguistic dimension is part of the perceivable world.
- The *semantics dimension* provides an unambiguous meaning of the elements of the model dimension (at all levels and including the associated expressions from the linguistic dimension) through an ontological or encyclopedic grounding of each single

6 Note that unlike the MOF and IRDS metadata frameworks, we are not talking here about the class/instance relationship borrowed from object orientation, but rather about semantic intension/extension relationships.

element. With this we can define: *A conceptual model is a model that has a complete grounding in the semantic definition, i. e., if all its constituents are defined there.*

If we restrict ourselves to the model and linguistic dimension, we find ourselves in the traditional world of modeling, which provides the framework for many useful activities: With the addition of the semantic dimension, however, we create the basis for a semantically unambiguous form of modeling, as it is needed in Computer Science and especially for the specification and realization of complex systems.

The semantics dimension was ignored for a long time because world knowledge was taken for granted and it was not realized that each application brings in its specific background knowledge. Without knowledge of world knowledge, however, interoperability and integration of systems remains a dream.

In Figure 4.3, we use the example of business balance sheets to illustrate the dimensions and the model and language hierarchies to be considered in conceptual modeling. In the middle column (the *'model wing'* of the triptych), we see four model levels H^n - H^{n+3} as described above. The metamodel at H^{n+2} provides the desired concepts from accounting theory; at the model level H^{n+1}, extensions of this metamodel are derived as balance sheet schemas, whose extensions at the individuals level H^n are concrete balance sheets for a specific key date.

If we now look at the right column (the *'language wing'*), we see a somewhat different hierarchy, namely that of the language definitions (where we mean by 'grammar' a complete syntactic definition of a language, i. e., including terminal symbols/literals and all composition rules for valid words and expressions. Thus, (formal) languages for the definition of metamodel, model and data/object expressions are to be provided.

Picture (1) in this wing shows the (language of) terminology introduced by the accounting theory of business economics: it represents the accounting metamodel. Picture (2) shows the representation of a model derived from the metamodel, namely the balance sheet scheme of the company MaTha Limited, which is the basis for the preparation of the company's balance sheet at certain reporting dates: The grammar here apparently provides for a tabular

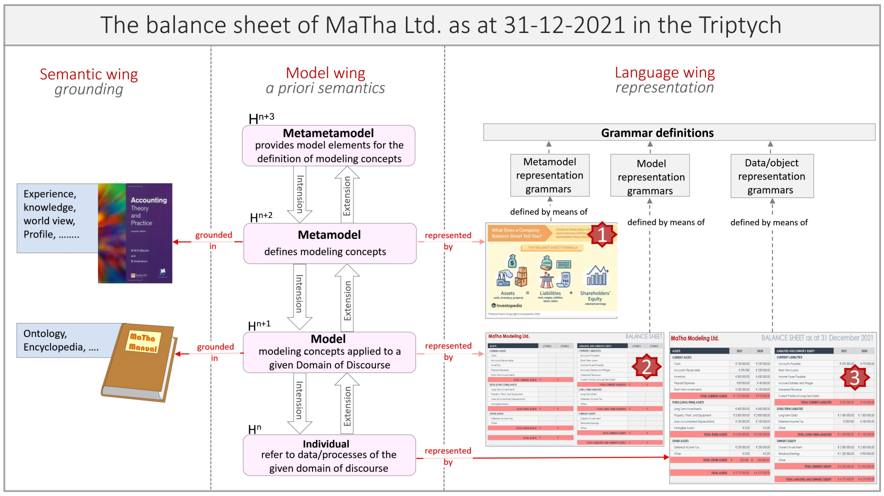

The balance sheet of MaTha Ltd. as at 31-12-2021 in the Triptych
Semantic wing
grounding
Model wing
a priori semantics
Language wing
representation
H^{n+3}
Metametamodel
provides model elements for the definition of modeling concepts
Intension
Extension
H^{n+2}
Metamodel
defines modeling concepts
H^{n+1}
Model
modeling concepts applied to a given Domain of Discourse
H^n
Individual
refer to data/processes of the given domain of discourse
Experience, knowledge, world view, Profile, ...
Accounting Theory and Practice
Ontology, Encyclopedia, ...
MaTha Manual
grounded in
represented by
Grammar definitions
Metamodel representation grammars
Model representation grammars
Data/object representation grammars
defined by means of
What Does a Company Balance Sheet Tell You?
THE BALANCE SHEET FORMULA
Assets
Liabilities
Shareholders' Equity
Investopedia
1
MaTha Modeling Ltd.
BALANCE SHEET
2
BALANCE SHEET as at 31 December 2021
3

Figure 4.3: Illustration of the triptych paradigm

representation with the columns ’Assets’ and ‘Liabilities’, as well as further horizontal subdivisions. An extension of this model is, for example, the ‘Balance sheet of the company MaTha as of 31.12.2021’ as shown in picture (3): Here it is no longer a kind of scheme, but a table occupied with concrete values. Quite obviously, then, we can say that this table is an expression representing a mental model of our object of observation MaTha Limited, so together they form the model ‘Balance sheet of the company MaTha as of 31.12.2021’. MaTha is therefore the object/origin of our model. Of course, it is also possible to imagine completely different models of a company, for example in terms of their organizational and process structure – you just have to configure the metamodel and the associated languages accordingly. Everything we have said in this example up to this point corresponds to the standard approach to the definition of models and technical languages in a wide variety of application disciplines. It is important for us to note that we have not talked about ‘conceptual model’ so far, nor did we need to.

This only happens if we look at the left column, the *‘semantic wing’*: There we find, among other things, codified knowledge about accounting (for example, in an encyclopedic work on accounting) and an organizational manual of MaTha Limited, in which every term occurring in this company is clearly defined, for example, in the form of a corporate ontology. If every element used in the metamodel and in the MaTha accounting model is grounded in this semantic dimension, then we speak of a *conceptual model* – both at the level of the metamodel and at the level of the model.

Two things should be mentioned here: First, it becomes clear that the same model can easily be applied in different settings: Let’s think of a tax advisor who, in addition to MaTha Limited, also looks after the company StoPa Inc. and uses the same accounting scheme for both: all he has to do is to change the literal of the company name on the language wing – and to replace the company manual of MaTha Limited with that of StoPa Inc. on the semantic wing. So the ‘reference model’ in the middle remains the same, but the conceptual semantics coorespond to the new company. Second, even if we use the terms metamodel, model etc. in the singular, this does not mean that on each level of abstraction only one model can exist. On the contrary, usually – with the exception of the top level which should ensure a certain comparability of the underlying extensions – many models are allowed on arbitrary levels; think, e. g., of different DDL relational schema definitions for a multi-database application. But of course this is also true for the metamodel level, for instance, if one wants to consider structural, process and functional models in a modeling compound and specify their interdependencies within that compound.

In other words, *‘multimodeling’*, *‘multilevel modeling’*, and *‘multiperspective modeling’*, as advocated by Ulrich Frank and the Duisburg school (Frank 2014), are intrinsic to the model dimension. Multilevel modeling attempts to overcome the classic MOF architecture with class-instance layering between the various levels. Multiperspective modeling supports the variety of views within a community of practice. It turns the global-as-design approach from head to toe with explicit orientation on language and modeling support for each of the community members. Ulrich’s MEMO approach integrates these and uses a model suite (Thalheim 2010) with overview, goal system, process, organisational structure, process control flow, IT infrastructure, decision support, performance indicator, and business process models.

To sum up: The Triptych paradigm is an explicit foundation of conceptual modeling. Terms imputed from linguistics and notions imported from semantics are the background of conceptual models.

4.6 Conclusio: Some Hints and Suggestions for the Future

As we have already indicated in Section 4.4, we assume that the importance of modeling will not decrease but rather increase if we do our homework in our community. The importance can be characterized with a Kondratieff wave sequence: A single wave includes phase states such as underappreciated and neglected, overappreciated and renamed, disappointed and devastating, enlightening practical, and uninteresting. Therefore, it is important that the knowledge and experience pool still is processed and systematized. Such a Body of Knowledge (BoK) is processed and systematized towards a comprehensive body of knowledge. Then a science and culture of models, model design and use, and modeling will continue to be an independent and at the same time fundamental sub-discipline of modern Computer Science (Cabot and Vallecillo 2022; Thalheim 2022a; Thalheim 2022b).

To be able to achieve this goal, modeling will have to provide convincing answers to questions like the following in the near future:

- Can we really avoid complexity through use of models?
  One of the main benefits of models is complexity reduction and focusing. This requires a mastery of the game of adequacy and concentration on the essential.
- What means capability-oriented application and usage of models?
  Use models whenever you can handle them in a proper way. They are not luxury goods, but goods that should facilitate life, activities, and practice purposefully.
- Are models supporting humanised thinking and acting instead being driven be the artificial?
  Technical and artificial worlds are far too complex and upsetting for humans. Models allow an understanding design and meaningful usage in the simplest possible and valuable way.
- Can models be used for generation, e. g., as powerful companion for all phases of programming, maintenance, and system realisation?
  Modeling as higher-order programming will be the real future of modeling in Computer Science. Models will become core elements of software and hardware after development of appropriate theories and sophisticated compilers.
- Do we have means for quality-aware modeling?
  Models have not to be true, but of good quality according to the form of use, i. e., right quality on the right place; but no more than that.
- What means to enhance models by interpretability?
  Models have to be intuitive for their Community of Practise. They are man-made and man-oriented instruments in all human endeavours.
- What benefit we could gain by model apprenticeship and eruditeness (Konstruktionslehre – modelology)?
  One should enjoy the benefits of each advancement. Let us try to develop a culture of models and modeling.
- Can we construct models for any scenario and function?
  No! But this is not necessary. We should focus on the core tasks of our discipline.
- Can we rely on a model?
  At least as long as it is a good model. As long as the model is designed for its proper use.

- Are models gifting us with conceptualisations?
  Hopefully, but this requires work. Conceptualisation and 'semantification' is, however, a nice task but not a must.
- What means skilled apprenticeship and mastery?
  Pure lore and teachings can be useful. But the real effect comes with use. Apprenticeship and mastery will become the killer modeling usage.
- Can we detect and avoid misuse, misapplication, and manipulation through models?
  Models are embedded in their own landscape and not arbitrarily transplanted or even misused and misapplied, To avoid this requires efforts and should be done with care on necessity.

References

Batini, C., Ceri, S. and Navathe, S. B. (1992). *Conceptual Database Design: An Entity-Relationship Approach.* Benjamin/Cummings.

Becker, J., Wolfgang, P. and Vering, O. (2012). *Grundsätze ordnungsmäßiger Modellierung: Konzeption und Praxisbeispiel für ein effizientes Prozessmanagement.* Berlin, Heidelberg: Springer.

Ben-Naim, A. (2007). *Entropy Demystified.* World Scientific.

Brambilla, M., Cabot, J. and Wimmer, M. (2017). *Model-Driven Software Engineering in Practice, Second Edition.* Synthesis Lectures on Software Engineering. Morgan & Claypool Publishers. DOI: 10.2200/S00751ED2V01Y201701SWE004.

Broy, M. and Stølen, K. (2001). *Specification and Development of Interactive Systems – Focus on Streams, Interfaces, and Refinement.* Monographs in Computer Science. Springer. ISBN: 978-1-4612-6518-4. DOI: 10.1007/978-1-4613-0091-5. URL: https://doi.org/10.1007/978-1-4613-0091-5.

Cabot, J. and Vallecillo, A. (2022). 'Modeling should be an independent scientific discipline'. In: *SoSyM Software and Systems Modeling* 21, pp. 2101–2107.

Carmona, J., Dongen, B. van, Solti, A. and Weidlich, M. (2018). *Conformance checking.* Springer.

Castellanos, A., Samuel, B., Recker, J., Jabbari, M. and Lukyanenko, R. (2018). 'Conceptual modeling research: revisiting and updating Wand and Weber's 2002 Research Agenda'. In: *AIS SIGSAND*, pp. 1–12.

Chen, P. P.-S. (1976). 'The entity-relationship model—toward a unified view of data'. In: *ACM transactions on database systems (TODS)* 1.1, pp. 9–36.

Codd, E. F. (2002). 'A relational model of data for large shared data banks'. In: *Software pioneers.* Springer, pp. 263–294.

Delcambre, L. M., Liddle, S. W., Pastor, O. and Storey, V. C. (2021). 'Articulating Conceptual Modeling Research Contributions'. In: *Advances in Conceptual Modeling.* Springer International Publishing, pp. 45–60.

DeMarco, T. (1978). *Structured Analysis and System Specification.* Yourdon Press.

Dumas, M., La Rosa, M., Mendling, J., Reijers, H. A. et al. (2013). *Fundamentals of business process management.* Vol. 1. Springer.

Elmasri, R. and Navathe, S. B. (2000). *Fundamentals of Database Systems, 3rd Edition.* Addison-Wesley-Longman.

Embley, D. W. and Thalheim, B. (2012). *Handbook of conceptual modeling: theory, practice, and research challenges.* Springer.

Eriksson, O., Johannesson, P. and Bergholtz, M. (2019). 'The case for classes and instances - a response to representing instances: the case for reengineering conceptual modelling grammars'. In: *European Journal of Information Systems* 28.6, pp. 681–693. DOI: 10.1080/0960085X.2019.1673672.

Ferstl, O. K. and Sinz, E. J. (2015). *Grundlagen der Wirtschaftsinformatik.* Oldenbourg Wissenschaftsverlag.

Forstmann, B. U. and Wagenmakers, E.-J., eds. (2015). *An introduction to model-based cognitive neuroscience.* Vol. 614. Springer.

Frank, U. (2014). 'Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges'. In: *Softw. Syst. Model.* 13.3, pp. 941–962.

Friedman, J. H. (2017). *The elements of statistical learning: Data mining, inference, and prediction.* springer open.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison Wesley.

Garmus, D. and Herron, D. (2001). *Function point analysis: measurement practices for successful software projects.* Addison-Wesley Longman Publishing Co., Inc.

Guarino, N. (1994). 'The ontological level'. In: *Philosophy and the cognitive sciences.* Holder-Pichler-Tempsky, pp. 443–456.

Guarino, N., Guizzardi, G. and Mylopoulos, J. (2019). 'On the Philosophical Foundations of Conceptual Models'. In: *Information Modelling and Knowledge Bases, IOS Press* XXXI, pp. 1–15.

Guizzardi, G., Wagner, G., Almeida, J. P. A. and Guizzardi, R. (2015). 'Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story'. In: *Appl. Ontology* 10, pp. 259–271. DOI: 10.25300/MISQ/2021/16027.

Haier, R. J. (2016). *The neuroscience of intelligence.* Cambridge University Press.

Halpin, T. and Morgan, T. (2010). *Information modeling and relational databases.* Morgan Kaufmann.

Henderson-Sellers, B. (2015). 'Why Philosophize; Why not Just Model?' In: *LNCS, Springer* 9381, pp. 3–17. ISSN: 1619-1366. DOI: 10.1007/978-3-319-25264-3_1.

Jaakkola, H. and Thalheim, B. (2011). 'Architecture-driven modelling methodologies'. In: *Information Modelling and Knowledge Bases.* Vol. XXII. IOS Press, pp. 97–116.

Karagiannis, D., Mayr, H. C. and Mylopoulos, J., eds. (2016). *Domain-Specific Conceptual Modeling, Concepts, Methods and Tools.* Springer. ISBN: 978-3-319-39416-9. DOI: 10.1007/978-3-319-39417-6. URL: https://doi.org/10.1007/978-3-319-39417-6.

Kaschek, R. and Mayr, H. C. (1976). 'A characterization of OOA tools'. In: *Proc. 4th International Symposium on Assessment of Software Tools.* IEEE, pp. 59–67. ISBN: 0-8186-7390-7. DOI: 10.1109/AST.1996.506478.

Kent, W., ed. (1978). *Data and Reality.* North-Holland.

Khatri, V. and Samuel, B. M. (2019). 'Analytics for managerial work'. In: *Communications of the ACM* 62(4), p. 100. DOI: 10.1145/3274277.

Lima, H. C., Laender, A. H., Moro, M. M. and Oliveira, J. P. de (2020). 'An analysis of the collaboration network of the international conference on conceptual modeling at the age of 40'. In: *Data & Knowledge Engineering* 130. DOI: 10.1016/j.datak.2020.101866.

Ludewig, J. (2002). 'Modelle im Software Engineering - eine Einführung und Kritik'. In: *Lecture Notes in Informatics, GI* P-12, pp. 7–22.

Lukyanenko, R., Castellanos, A., Parsons, J., Tremblay, M. C. and Storey, V. C. (2019). 'Using conceptual modeling research to support machine learning'. In: *Information Systems Engineering in Responsible Information Systems (CAiSE Forum 2019)*, pp. 70–81. DOI: 10.1007/978-3-030-21297-1_15.

Mahr, B. (2015). 'Modelle und ihre Befragbarkeit – Grundlagen einer allgemeinen Modelltheorie'. In: *Erwägen-Wissen-Ethik (EWE)* Vol. 26, Issue 3, pp. 329–342.

Mahr, B. (2021). *Schriften zur Modellforschung*. Herausgegeben von K. Robering. Brill Mentis, Michaelisbund.

Mayr, H. C. and Thalheim, B. (2021). 'The Triptych of Conceptual Modeling – A Framework for a Better Understanding of Conceptual Modeling'. In: *Softw. Syst. Model.* 20.1, pp. 7–24.

Mayr, H. C. and Thalheim, B. (2022). 'The Anatomy of Conceptual Models'. In: *Advanced Information Systems Engineering - 34th Int. Conf., CAiSE 2022.* Ed. by X. Franch, G. Poels, F. Gailly and M. Snoeck. Vol. 13205. Lecture Notes in Computer Science, pp. 551–553. DOI: doi.org/10.1007/978-3-031-07472-1.

Metz, R. (2006). 'Empirical Evidence and Causation of Kontratieff Cycles'. In: *Kondratieff Waves, Warfare and Worls Security.* IOS Press, pp. 91–99.

Moody, D. (2009). 'The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations'. In: *IEEE Trans. on Software Engineering* 35,6, pp. 756–779.

Olivé, A. (2007). *Conceptual modeling of information systems.* Springer Science & Business Media.

Parker, B. (1992). 'Introducing ANSI-X3.138-1988: a standard for information resource dictionary system (IRDS)'. In: *[1992] Proceedings of the Second Symposium on Assessment of Quality Software Development Tools*, pp. 90–99. DOI: 10.1109/AQSDT.1992.205841.

Recker, J., Lukyanenko, R., Jabbari, M., Samuel, B. M. and Castellano, A. (2020). 'From representation to mediation: a new agenda for conceptual modeling research in a digital world'. In: *MIS Quarterly* 45(1), pp. 269–300. DOI: 10.25300/MISQ/2021/16027.

Ross, D. T. and Schoman, K. (1977). 'Structured Analysis for Requirements Definition'. In: *IEEE Transactions on Software Engineering* 1, pp. 6–15. DOI: 10.1109/TSE.1977.229899.

Sandkuhl, K., Fill, H.-G., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A., Schwabe, G., Uludag, Ö. and Winter, R. (2018). 'Analytics for managerial work'. In: *Business & Information Systems Engineering* 60, pp. 69–80. DOI: 10.1007/s12599-017-0516-y.

Stachowiak, H. (1973). *Allgemeine Modelltheorie.* Springer.

Storey, V. C., Lukyanenko, R. and Castellanos, A. (2023). 'Conceptual Modeling: Topics, Themes, and Technology Trends'. In: *ACM Computing Surveys* 55.

Teichroew, D. and Hershey, E. A. (1976). 'PSL/PSA a computer-aided technique for structured documentation and analysis of information processing systems'. In: *ICSE'76: Proceedings of the 2nd international conference on Software engineering.* ACM.

Thalheim, B. (2000). *Entity-relationship modeling – foundations of database technology.* Springer.

Thalheim, B. (2010). 'Model Suites For Multi-Layered Database Modelling'. In: *Information Modelling and Knowledge Bases XXI.* Vol. 206. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 116–134.

Thalheim, B. (2022a). 'Auf dem Wege zur Modellkunde'. In: *Modellierung 2022.* Ed. by M. Riebisch and M. Tropmann-Frick. Vol. P-324. LNI. Gesellschaft für Informatik e.V., pp. 11–32.

Thalheim, B. (2022b). 'Models: The Fourth Dimension of Computer Science – Towards Studies of Models and Modelling'. In: *Software & Systems Modeling* 21, pp. 9–18.

Tillmann, B. (2012). ‘Music and Language Perception: Expectations, Structural Integration, and Cognitive Sequencing’. In: *Topics in Cognitive Science* 4.4, pp. 568–584. doi: 10.1111/j.1756-8765.2012.01209.x.

Van Der Aalst, W. (2016). *Process mining: data science in action*. Vol. 2. Springer.

Verhulst, P.-F. (1838). ‘Notice sur la loi que la population suit dans son accroissement’. In: *Correspondance Mathématique et Physique* 10, pp. 3–21.

Weske, M. et al. (2007). ‘Concepts, Languages, Architectures’. In: *Business Process Management.*

Yu, E. (2009). ‘Social Modelling and i*. Conceptual Modeling: Foundations and Applications’. In: *Conceptual Modeling: Foundations and Applications: Essays in Honor of John Mylopoulos*, pp. 99–121. doi: 10.1007/978-3-642-02463-4_7.

Chapter 5

On Views, Diagrams, Programs, Animations, and Other Models

Hend*erik* A. Proper and Giancarlo Guizzardi

Humanity has long since used models in different shapes and forms to understand, redesign, communicate about, and shape, the world around us; including many different social, economic, biological, chemical, physical, and digital aspects. This has resulted in a wide range of *modeling practices*. When the models as used in such *modeling practices* have a key role to play in the activities in which these *modeling practices* are 'embedded', the need emerges to consider the effectiveness and efficiency of such processes, and speak about *modeling capabilities*. In the latter situation, it becomes relevant to develop a thorough understanding of the artifacts involved in the modeling practices/capabilities. One field in which models play (an increasingly) important role is the field of system development (including software engineering, information systems engineering, and enterprise design management). In this context, we come across notions, such as views, diagrams, programs, animations, specifications, etc. The aim of this paper is to take a fundamental look at these notions. In doing so, we will argue that these notions should actually be seen as specific kinds of models, albeit for fundamentally different purposes.

5.1 Introduction

Whenever we are confronted with complex phenomena, such as the processes we observe in nature, the construction of buildings, the design of information systems, etc, we tend to *'work with'* an *abstraction* (in our mind) of the actual phenomenon; zooming in on those *'properties'* of the phenomenon that matter to us, and filtering out all the properties that are not germane to the goals at hand. When we externalize this *abstraction* in terms of some artifact, then this artifact is a model (to us, as an individual) of the observed phenomenon.

More generally, one can observe how humanity has long since used models to understand, redesign, communicate about, and shape, the world around us, including many different social, economic, biological, chemical, physical, and digital aspects. These models may take different shapes and forms, such as sketches, precise drawings, textual specifications, or tangible forms mimicking key physical properties of some original. This wide spread, and natural (Zarwin et al. 2014) use of models has resulted in many different *modeling practices*.

When the models as created and/or used in such *modeling practices* have a key role to play in the activities in which these *modeling practices* are 'embedded', a natural need emerges to consider the effectiveness and efficiency of such processes, and speak about *modeling capabilities.*[1] In the latter situation, it becomes relevant to develop a thorough understanding of the artifacts involved in the modeling practices/capabilities.

One field in which models play (an increasingly) important role is the field of system development (including software engineering, information systems engineering, and enterprise design management[2]). In this context, we come across notions, such as views, diagrams, programs, animations, specifications, etc. In this chapter, we aim to take a deeper look at these notions, where we will also argue that these notions should be seen as specific kinds of models; albeit for fundamentally different purposes.

In line with this, the remainder of this chapter is structured as follows. In Section 5.2, we start by zooming in on the notion of (domain) model. In doing so, we will discuss the importance of knowing a model's purpose and its potential Return on Modeling Effort (RoME) in particular. This will then also take us, in Section 5.3 to the notion of conceptual model, which is a class of models that has initially grown to play an important role in the field of information systems engineering, but has a much wider role to play. Identifying conceptual models as a distinct class of models, does suggests there to be a class of domain models that are not conceptual. This also results in a need to speak about *conceptual fidelity.*

Building on this grounding, Section 5.4 discusses the notion of *view* as complexity management mechanism that supports making complex models *cognitively tractable.* Finally, before concluding, Section 5.5 position notions such as diagrams, programs, and animations, as being specific kinds of models, covering different purposes, different audiences.

5.2 Domain Models and Their RoME

Based on foundational work by, e. g., Apostel (1960), and Stachowiak (1973), more recent work on the same by different authors (Rothenberg 1989; Harel and Rumpe 2004; Thalheim 2011; Sandkuhl et al. 2018), as well as our own work (Hoppenbrouwers et al. 2005; Proper et al. 2005; Guarino et al. 2020; Guizzardi 2007; Bjeković et al. 2014; Proper and Guizzardi 2020; Proper and Guizzardi 2021; Proper and Guizzardi 2022), we currently understand a *domain model* to be:

> A *social artifact* that is *understood*, and *acknowledged*, by a *collective human agent* to *represent* an *abstraction* of some *domain* for a particular *cognitive purpose.*

With *domain*, we refer to *'anything'* that one can speak and/or reflect about; i.e. the domain of interest. As such, *domain* simply refers to *'that what is being modeled'.*[3] Furthermore, the domain could be something that already exists in the *'real world'*, something that is desired to exist in the future, something imagined, or even something that is brought about by the existence of the model itself.[4] In the context of system development at large, more

1 Ontologically speaking, capabilities (or capacities) are *gradable dispositions*, i. e., properties that are manifested in certain situations via the occurrences of events of a certain kind (Azevedo et al. 2015).

2 Used here as a generalization of (among others) Business Process Management, Enterprise Architecture Management, Enterprise Engineering, Enterprise Transformation, and Organization(al) Design.

3 In Guizzardi (2013), a more precise characterization of the notion of the domain being modeled is provided in terms of usage: *abstraction* – when referring to a *mental model* (Guarino et al. 2020) of a *particular* state of affairs (e. g., a configuration of the existing subway lines of Milan); *conceptualization* – when we are referring to general notions that are *repeatable* across multiple state of affairs (e. g., the concept of a subway line, being an ancestor, or an offspring in genealogy).

4 In Guizzardi and Proper (2021), we discussed how models can socially be seen as complex speech acts and, as such, they can be characterized as having different relations to world in terms of *directions of fit.* In particular,

specific classes of domain models include enterprise (architecture) models, business process models, ontology models, organizational models, information models, software models, etc. We consider all of these as valued members of the larger family of *domain models*.

A model must always be created for some *cognitive purpose*, i. e., to express, specify, learn about, or experience, knowledge regarding the modeled domain. This also implies that, in line with the *cognitive purpose* of the model, some (if not most) *'details'* of the domain are consciously filtered out. As we regard a model to be an *artifact*, this also implies that it is something that exists outside of our minds, i. e., as *'represented abstractions'*. More specifically, a model is seen as a *social artifact* in the sense that its role as a model should be recognizable by a *collective human agent.*[5] The *understood, and acknowledged, by a collective human agent* phrase also differentiates *domain models* from, e. g., machine-learned models. A domain model can certainly involve complex mathematical formalisms, or computer readable specifications. However, it must be transparent expressions of a human agent's mental model (or conceptualization), in line with the *cognitive purpose*. In the context of system development, models typically take the *form* of some *'boxes-and-lines'* diagram. More generally, however, domain models can, depending on the *purpose* at hand, take other forms as well, including (controlled) natural language texts, mathematical specifications, games, sketches, animations, simulations, and physical objects (we will elaborate on this in Section 5.5).

Finally, the creation, administration, and use, of such domain models, as well as the development of modeling capabilities, require investments in terms of time, money, cognitive effort, etc. We contend that such investments should be met by a (potential) return. In other words, the resulting models and/or the processes involved in their creation, administration, and use, should add value that make these investments worth while. This has resulted in the notion of Return on Modeling Effort, which we first coined in a publication,[6] in Op 't Land et al. (2008) while a more elaborate discussion of the concept in Guizzardi and Proper (2021) and Proper and Guizzardi (2022) is provided as part of our joint endeavor to better understand the foundations of (domain) modeling and modeling practices.

The enactment of a modeling practice and the exercise of a modeling capability is an event, an intentional event (or an action). From an ontological point of view, events are always manifestations of dispositions. Ordinary events are manifestations of interacting dispositions (Molnar 2003; Mumford and Anjum 2011). In particular, modeling events are the manifestation of the modeler's beliefs, intentions and modeling capabilities interacting with capabilities/affordances that are present in the modeling ecosystem at hand, including modeling languages (in terms of primitives, semantics and pragmatics), other models, tools, methodologies, etc. In line with the ontology of value propositions proposed in Sales et al. (2017), we speak of value experiences with its *benefits* (afforded by a system of dispositions) and *sacrifices*. This view allows us to think of the enactment of modeling practices as value

we recognized the case of the *constructive models*, i. e., models with a *double direction of fit*, namely, they represent the propositional content of a creative speech act and end up describing the entities that they help to create. Examples include patent diagrams or diagrams that, e. g., create new real estate properties by dividing a piece of land.

5 The pre-noun *collective* does suggest that it would require the involvement of multiple people. We do, indeed, acknowledge the use of domain models by an individual person as well, but prefer to treat this as a special case concerning a *'self-shared'* model.

6 The notion of RoME actually made its first informal appearance in Proper (2005) as a leading principle in the research group of one of the authors, while a first informal elaboration of the concept was published as a blog posting (Proper 2009).

experiences and, hence, to think of RoME as *Return of Model Experiences*,[7] also underlining the fact that the value of models lies mainly in their *Value in Action* (ViA) (Proper and Guizzardi 2022) during their creation and/or use.

The complexity management tools (views) as discussed in Section 5.4, as well as the other types of modeling artifacts we discuss in Section 5.5, result in different types of artifacts that afford different modeling experiences leading to *Value in Action*.

5.3 Conceptual Fidelity

In the context of information systems engineering, an important role is played by *conceptual models*; which we see as a specific class of domain models. According to the traditional information systems engineering view (ISO/IEC JTC 1/SC 32 Technical Committee on Data management and interchange 1987), a conceptual model captures the essential structures of some *universe of discourse*. In this context, conceptual models are used to express the concepts, and their (allowed) relations, of the *universe of discourse* (while avoiding the inclusion of implementation/storage details).

The field of information systems engineering, indeed, provides a fruitful application area for conceptual modeling. At the same time, however, we suggest to avoid a *'framing'* of what a conceptual model is to this application area only. As such, we suggest a more generalized understanding of the notion of *conceptual model*. More specifically, based on Proper and Guizzardi (2020), Proper (2021) and Guarino et al. (2020), in our current understanding a *conceptual model* is:

A domain model, where:

1. the purpose of the model is dominated by the ambition to remain as-true-as-possible to the *conceptualization* of the domain by the collective agent, while
2. there is an explicit *mapping* from the elements in the model to the latter *domain conceptualization*.

The *domain conceptualization* identifies the fundamental concepts in terms of which the *collective agent* create(s) their conception of the world. This mapping specifies the real-world semantics of that model and characterizes the *ontological commitment* (Guizzardi 2007) of the model (and the *collective agent*) as well as its real-world semantics.

Returning to the above point regarding the need to consider the role of conceptual models beyond the field of information systems engineering, the ambition to *as-true-as-possible to the conceptualization of the domain by the collective agent* is not only of value in the context of information systems engineering, but other contexts as well. For instance, Guarino et al. (2020) already stated that the history of conceptual modeling can be traced back to at least the 60s (Quillian 1968). Furthermore, ontology engineering (Guizzardi 2007) also involves the construction of conceptual models representing a (domain) ontology.

At a more general level, we also observe that in many different endeavors in which we (as humans) aim to understand the workings of some domain and/or aim to express, or study, design alternatives, we actually do so in terms of (purpose and situation specific) domain models. This includes many examples across science and engineering at large. We

7 We thank Tiago Sales and Isadora Valle Sousa for calling this to our attention. We shall explore these notions together in a future paper.

also argue that in these cases, a deepening of our understanding of the essential mechanisms leads to a natural drive to create domain models that remain as-true-as-possible to the original domain (and our conceptualization thereof), i. e., conceptual models.

Since a conceptual model is meant to be used by human agents in tasks such as domain understanding and learning, communication, problem-solving, and meaning negotiation (Guarino et al. 2020), another fundamental quality attribute of a conceptual model is its *pragmatic efficiency*, i. e., how easy it is for those human agents to perform these aforementioned tasks with these models (Guizzardi et al. 2002; Guizzardi 2013).

As a result, a conceptual model provides an explicit – human understandable, ideally, pragmatically efficient – representation of a theory about the entities and their ties that are assumed to exist in a given *domain of interest* (the ontological commitment); as such explicitly capturing descriptive and/or prescriptive selected aspects of the modeled domain. Conceptual models, therefore, enable us to explicitly clarify the things we talk and reason about (at a chosen level of abstraction and from a desired perspective).

Identifying conceptual models as a specific class of domain models, does raise the question regarding the role of *'other'* domain models that are *'not conceptual'*. In Proper and Guizzardi (2020), it is suggested to, next to conceptual models, also identify *computational-design models*. These latter models may involve *'conceptual compromises'* (with regard to the ambition *to remain as-true-as-possible to the original domain conceptualization*) to cater for highly desirable computational considerations to, e. g., support simulation, animation, or even execution of the model. In Proper (2021), it is suggested to generalize this towards *utility-design models*, to cater for the fact that *'conceptual compromises'* may not only be introduced for *computational* purposes, but also for e. g., *experiential* purposes, such as the ability to touch, feel, or even *'enact'* a model.

An interesting analogy, which certainly needs further investigation, is the notion of *surrogate modeling* in the context of simulation (Razavi et al. 2012) of real-world systems. The level at which a simulation model reflects all (relevant) properties of a (planned/existing) real-world system is referred to as the *fidelity* of the simulation model: 'Fidelity in the modeling context refers to the degree of the realism of a simulation model' (Razavi et al. 2012). Likewise, one can speak of being as-true-as-possible-to a given domain as a sort of *conceptual fidelity*. Conceptual fidelity, also frequently called *domain appropriateness*, represents the level of homomorphism between a given representation and the underlying domain conceptualization it commits to. In the ideal case, this representation artifact is not only isomorphic to the structure of that conceptualization (i. e., it represents in a univocal and non-redundant way all its constituting concepts and only them) but it also only allows for interpretations that represent state of affairs deemed acceptable by that conceptualization (Guizzardi et al. 2005; Guizzardi 2013). As, in the case of simulation models of real-world systems, the involved high fidelity models may be too computationally intensive to simulate as a whole, one uses so-called *surrogate models* (Razavi et al. 2012) that are computationally more efficient, while approximating the high fidelity model good enough to meet the (optimization) purpose at hand.

In an information systems engineering context, the ambition for a conceptual model to *remain as-true-as-possible to the conceptualization of the domain by the collective agent*, has a direct correspondence to the *conceptualization principle* as put forward in the well known ISO report on the design of information systems (ISO/IEC JTC 1/SC 32 Technical Committee on Data management and interchange 1987).

It is important to note that we do not argue that non-conceptual models are a bad thing; far from it. However, it needs to be clear what the *'conceptual deviations'* are of a non-conceptual model in relation to the conceptual model of the same domain, and what the benefit are of these deviations in terms of e. g., computational efficiency or experiential properties. As such, it might quite well be the case that one conceptual model has different associated non-conceptual models catering for different needs (Guizzardi 2010). Conversely, we would expect that each non-conceptual model has been based on (at least one) corresponding conceptual model.

Returning briefly to the notion or RoME, we postulate that the RoME of a conceptual model is at least as high as the sum of the RoME of each of the non-conceptual models that have been derived from it.

5.4 Complexity Management and Views

As discussed in e. g., Arbab et al. (2007), Lankhorst et al. (2017) and Frank (2002), views are positioned as providing a powerful mechanism to create domain models that are more suitable in the communication with different stakeholders (and for different purposes) than the 'full scale' model would provide. At a fundamental level, views are a complexity management mechanism that supports making complex models *cognitively tractable*, while also tuning this to the audience (and their concerns/interests at hand). In this vein, IEEE (2000) defines the notion of *view* (in the context of the architecture of software systems) as: 'A representation of a whole system from the perspective of a related set of concerns'. Based on this definition, Lankhorst et al. (2017) speaks about a view as having an 'underlying model', making it explicit that a view is indeed based on an *underlying model.* At the same time, the fact that a view provides a *representation of a whole system from the perspective of a related set of concerns* implies that a view is a model as well. In line with this, we currently understand a *view* on another *domain model* (and the modeled domain) as being:

> A *domain model* of the modeled domain, which differs from the original domain model, while:
>
> - being of the same level of conceptual fidelity,
> - and providing an integrated subset of the *potential* information as provided by the original domain model.

which we hold as being a generalization of the way(s) the notion of view is used in IEEE (2000), Lankhorst et al. (2017) and Frank (2002) where, for instance, the *from the perspective of a related set of concerns* (IEEE 2000) corresponds to the need of a view to be an *integrated* subset of the information as provided by the original domain model. Note that, as a view is a domain model as well, one can recursively create views on views.

We do realize that the *information as provided by the original domain model* may be hard to formalize, as the *information as provided* does depend on the observer of the model.[8] This is also the reason why we have added the *'in potential'* qualification. For instance, a

8 This issue is actually analogous to the challenge of defining what information can potentially be provided by an 'information carrier' in general, in the context of information retrieval systems. A theoretical framework to explicitly reason about this has been reported in Proper and Bruza (1999) and Bommel et al. (2007). It remains a possible avenue for further research to apply this in the context of models and views. At least for some models with certain explanatory functions, the information content of a model can be associated with its ability to answer *why-questions* (Romanenko et al. 2022b).

large domain model might be so (cognitively) *'overwhelming'* to an observer that they might actually glean more information from the view than from the original model.

When the involved domain models (i. e., the view and the original domain model) are represented in terms of explicit modeling languages, one can mathematically think of these models as being typed-graphs that are typed in terms of the modeling concepts as provided by the modeling language(s). In that case, the notion of 'proper subset of the *potential* information as provided by the original domain model' can be formalized by requiring there to be a function that maps a sub-graph of the (implied) graph representing the original domain model to the graph that represents the view, where this function respects the syntax and (formal) semantics of the modeling language(s) used. The requirement that the (potential) information provided by a view should be *integrated*, then corresponds to the requirement of the graph representing the view being a connected graph.

Note that models (and views) are not required to be *'minimal'*. It is allowed for models to contain elements that can be derived from other parts of the model. This is also why, above, we added *'implied'* when writing '(implied) graph representing the original domain model'. Consider, for instance, the derived relationships in ArchiMate (Lankhorst et al. 2017) that, depending on the purpose at hand, may or may not be included in the model/view. As an example, if a business role *r* is assigned to (the execution of) business process *p*, while business process *p* is (part of) the realization of business service *s*, then this implies that business role *r* is also (part of) the realization of business service *s*.

In terms of the above discussed possible formalization of views, we currently posit there to be four *'stackable'* operations to construct views:

1. *Selection* – involving the focusing of the view on a specific part of the original model.

   In this case, the mapping function should involve a bijective function between from the (selected!) part of the graph of the original domain model, and the view's graph.

   The central question in using this operation is *what is the (sub)domain to focus on in the view?* When one would be taking a photo of a subject, this would correspond to the question of where to point the camera at. In this case, as the metaphor goes, the angle (perspective) from in which the picture (i. e., model) is taken reveals or occludes certain elements from the scene (i. e., domain).

   In a system development context this operation pertains to both the scoping of what is to be included in the model/view in terms of e. g., enterprise-wide, business unit specific, etc., as well as the perspective in terms of the high level structures the well-known 'engineering frameworks' (e. g., Sowa and Zachman 1992; Spewak 1993; Boar 1999; The Open Group 2011; van't Wout et al. 2010; DoD Deputy Chief Information Officer 2011; Frank 2014; Band et al. 2016). For each of the 'cells' of the latter engineering frameworks, one can create a view on the model of the system (of systems) as a whole.

2. *Distillation* – involving a further abstracting away from the original domain, by distilling specific aspects of the domain.

   In *'distilling'*, one *may* need to combine certain elements/properties from the original domain model. Therefore, in this case, the mapping from the (selected part of the) original domain model to the view's graph would involve a surjective (but possibly bijective) function.

The central question in using this operation is *what phenomena to include in the view?* In terms of the analogy of taking photos, this would correspond to the question if one would make a color photo, a gray-scale photo, or possibly even an infra-red photo.

In a system development context, this is where we find the need to hone in on specific aspects (e. g., process flows, resource use, information flows, etc.), and/or cross-cutting concerns (e. g., security, privacy, sustainability, etc.).

3. *Summarization* – also involves a further abstracting away from the original domain, but now by clustering of different elements in the original domain model into more coarse grained elements.

   As a result, the mapping from the (selected part of the) original domain model to the view's graph would again involve a surjective function, but in this case, this is not allowed to be a bijective function.

   The leading question here is *what level of detail is needed?* In terms of taking a photo, this is the question of the level at which one zooms in/out on the subject in relation to the resolution of the optical sensors as used in the camera.

4. *Translation* – involving a translation between one modeling language to another modeling language (and medium).

   This implies that the mapping needs to provide a *'translation'* between the modeling languages used.

   The main question for this operation is *what is the best language and medium to represent the view?*

   In a system development context, this is where also find the variety of representations that are tuned to different stakeholders in terms of e. g., heat-maps, matrix-like representations, textual descriptions combined with info-graphics, animations, etc. In Section 5.5, we will return to this point in terms of the representation of a model actually involving a connection between its informational payload and a concrete *'medium system'*.

The needed mix of operations used in creating a specific view, depends on the specific purpose for the view at hand. Even more, given one domain model, one can even construct an entire hierarchy of views, using the different operations.

As mentioned above, in the context of system development, the *selection* operation has a natural link to the high level perspectives of engineering frameworks. Collectively, the views that correspond to the different cells in such frameworks provide different *'chunks'* that make up the model of the entire (as-is/to-be) system. In addition, strategies exist that enable a *recoding* or *modularization* of models using an underlying foundational ontology. In *model recoding* (Figueiredo et al. 2018), models are re-organized by grouping elements in terms of higher-granularity modeling primitives. In *model modularization* (Guizzardi et al. 2021), models are reorganized in cognitively tractable chunks that can be understood as a whole.

The *distillation*, *summarization*, and *translation* operations are directly linked to the question of the concepts and relations to be used in modeling the domain. In other words, the ontological commitment by which we will look at the domain. This is where we find the meta-models[9] underlying actual modeling languages and methods (e. g., OMG 2003; Band

9 The term meta-model is often overloaded in the area. Often, it refers to the description of a language's *abstract syntax*. In contrast, we are using it here in the sense of what is termed the *ontological meta-model*

et al. 2016), as well as explicit *'content frameworks'* (e. g., van't Wout et al. 2010; The Open Group 2011) that (albeit without a *'concrete syntax'*) do define the concepts and relation in terms of which the system is to be observed and modeled.

The question of *what level of detail is needed?* behind the *summarization* operation is also directly linked to the role of views as a complexity management mechanism to make complex models *cognitively tractable.* What is needed for summarization is some kind of *'(un)folding'* mechanism. Having a (recursive) (un)folding mechanism also enables the construction of a dynamic hierarchy of views, that allows for a navigation in terms of zooming in/out akin to the way we use Google Maps.

The needed (un)folding mechanism can be based on *mereology* (i. e., part-whole relations), as well as different forms of *'attribution'.* For instance, in the case of processes, it is quite commonplace to decompose these in terms of their mereological structure, i. e., decomposing a process into smaller temporal parts (sub-processes/tasks). In the case of organizational structures, the organizational hierarchy (which is structure of delegation power) coincides with a mereological structure in which organizations, their branches and units are decomposed into other smaller (functional parts).

'Attribution' refers to the fact that one concept might be considered as an attribute of another concept. For instance, the height of a person, the name of a person, etc are an attribute of a person. This *'attribution'* can be applied recursively in the sense that a person in the role of a manager of a department, might be seen as an attribute of that department.

General strategies for the (un)folding mechanisms needed for model summarization have been the subject of study in the past in the context of dealing with large conceptual models (Hofstede et al. 1992; Campbell et al. 1996; Creasy and Proper 1996), as well as more recently utilizing foundational ontologies to generate ontologically founded (un)foldings (Guizzardi et al. 2019; Romanenko et al. 2022a) of large conceptual models. The general idea of the latter approaches is to let the models undergo an (automatic) lossy transformation based on the underlying foundational ontology, to yield another model but capturing the gist of what the original model was about. By applying this recursively, a hierarchy of (ontology based) summarizations results.

Since a view involves a mapping from the original model to the view that is generally a surjective function (and not a bijective one), updating/editing a view can lead to a variation of the *'view update'* problem as known from the field of databases. Operationally this means that if a change is made in a view based on some original domain model, it may not be clear how to then make a corresponding change in the original domain model.

In practice, this *'view update'* problem becomes even more pressing as in the context of system development one is likely to actually start modeling a system from different angles; not unlike taking photos of the same object. Of course, knowing that these would be models of the same system, would imply that these models are essentially views of a larger model. Indeed, during a modeling process, one may use views to gradually (in a bottom up fashion) construct the *'larger picture'.*

A challenge is, of course, to ensure linkages between these views to maintain consistency and maintain/obtain the *'larger picture'* (see discussion, e. g., in Boiten et al. 2000). When, across an engineering framework, the *'cell-specific'* meta-models are aligned well – as is

of the language, or simply, the *ontology of the language*, i. e., a description of the worldview embedded in the language's modeling primitives. For example, Peter Chen's Entity Relationship model commits to a worldview that accounts for the existence of four types of things: entity, relationship, attribute and attribute value spaces (Guizzardi 2007).

explicitly the case for, e. g., ArchiMate (Band et al. 2016), IAF (van't Wout et al. 2010) and MEMO (Frank 2014), views corresponding to the different *'cells'* of the framework can be connected to maintain/obtain the *'larger picture'*.

5.5 Diagrams, Programs, Animations, and Other Models

Before we review (some of) the *'other'* model kinds, such as *tables*, *diagrams* and *animations*, it is important to have a closer look at the actual representation of models.

A model will be expressed in terms of some modeling language. This can be a precisely (a-priori) defined modeling language, but can also be a highly informal ad-hoc (emerging) language. In a (modeling) language, one can make a distinction between the conventions that govern what constructions are allowed in the language, and conventions that govern the way these are concretely presented in terms of representational mechanisms associated to a medium system. The former corresponds to the *grammar* and the *abstract syntax* of the language, while the latter pertains to the *concrete syntax*. The *concrete syntax* also ties a model's *abstract syntax* to an actual medium system. Obvious examples of medium systems that can be used to *'render'* the concrete syntax models include paper (including the back of a napkin), a 2D vector graphics (and textual fonts) rendering engine, or a simulation engine. Less obvious examples, include game engines or even tangible objects, in order to create models (and views) that may provide a more tangible experience.

Building on this, it is also important to note that a model (qua representation on a medium system) may be of a static nature or of a dynamic (and even interactive) nature. This distinction is actually orthogonal to the question if the modeled domain is static or dynamic.

If a domain is static, one could, indeed imagine creating a model with a static representation, such as a simple paper-based 'org chart' of the structure of an organization. However, one could also opt to, for example, create a *'navigable'* model where a 'viewer' can interactively navigate over/through the model, as we, e. g., already hinted at when discussing the (un)folding of models in Section 5.4.

Conversely, a dynamic domain such as a business process can be represented in terms of a simulation or animation, but also as a static representation in terms of, for example, a BPMN model.[10]

In the remainder of this section, we will argue how *specifications*, *programs*, *diagrams*, *tables*, *spreadsheets*, *simulations*, and *animations* can all essentially be seen as models; albeit with fundamentally different purposes and represented on different media systems. The chosen set (*specifications*, *programs*, *diagrams*, *tables*, *spreadsheets*, *simulations*, and *animations*) is not intended as a complete coverage of all *'things model'* that may be used in a system development context. Together, however, they do illustrate the richness of the kinds of models we may come across in the *modeling practices* as embedded in system development:

- *Specification* – A specification is a model that normatively prescribes the properties of a (to be designed, to be elaborated, to happen, to be brought about, …) phenomenon.

  In terms of our earlier work (Guizzardi and Proper 2021) towards a taxonomy of modeling-related goals, a specification has a world-to-model direction of fit (in a

10 Where this model can, of course, be complemented with a simulation of actual process instances.

world-to-word vein, cf. Searle, Willis et al. 1983) with the aim to *change the world* (the world in the sense of the phenomenon which' properties are described normatively).

Specifications tend to be represented using precise/formal languages on 2D text/-graphics based medium systems. For instance, a specification of business rules in as a text file in controlled natural language format, a mathematical specification on (digital) paper, or a graphical Petri-net based specification.

- *Program* – A specification which models the *required behavior of a computer* in an actionable way, such that a computer can directly exhibit this required behavior (via interpretation or compilation).

  Traditionally, programs are specified in some controlled textual form. In the past, programs were specified in terms of other medium systems, such as punched cards. Meanwhile, there has also been an increase in the use of visual ways to represent programs.

  Some of the modern editors for programs allow for some forms of (un)folding, de-facto resulting in a more dynamic (navigable) representation.

  Although we consider program as models (again, of computation), we do not consider programming languages as appropriate conceptual modeling languages. Programming languages are designed with computational concerns in mind (e. g., computational complexity, performance, to facilitate compiler construction) and as a result, for the purpuse of conceptual modeling, they: compromise expressivity and conceptual fidelity; hinder separation of concerns by forcing the modeler to consider at the same time conceptual, design and implementation issues.[11]

- *Diagrams* – Diagrams, in particular involving boxes-and-lines, are a common way to represent models. In general, diagrams involve some (static) graphical structure, possibly adorned with icons and/or text. In principle, diagrams provide a static representation.

  However, as hinted at before, such models can be made dynamic in a Google Maps like style by (un)folding and/or blending in/out specific (types of) elements. See the earlier discussions in Section 5.4 regarding the *distillation* and *summarization* operation for the creations of views.

  Diagrammatic notations are often part of the concrete syntax of general-purpose and domain-specific modeling languages alike. When designed in a proper way, diagrammatic notations can increase the pragmatic efficiency of the models it produces (Guizzardi 2013; Guizzardi et al. 2002). However, in order to attain these properties, these notations have to be properly designed (Moody 2009) lest denting problem-solving and producing unintended cognitive inferences (*implicatures*) (Guizzardi et al. 2002; Guizzardi 2013).

- *Table* – A table essentially provides a two-dimensional grid representation on a 2D medium (such as paper or a computer screen) that can capture a (possibly derived) ternary relation (type) concerning the modeled domain.

11 For this discussion in the context of ontology engineering, see, e. g., Guizzardi (2007); for the trade-off between expressibity and tractability in knowledge representation languages, see, e. g., Levesque and Brachman (1987). Another manifestation of this problem is the well-known impedance mismatch problem in mapping ontologically-rich conceptual models to relational databases, see, e. g., Guidoni et al. (2022).

In principle, a table is a static representation. However, by *'allowing'* one to blend in-/out specific rows/columns, thus changing the *'informational payload'* which the table (qua model) provides to us at that moment, the representation becomes interactive.

- *Spreadsheet* – A spreadsheet is a specific way to represent/render one or more connected tables on a computational medium system. Using formulas, derived parts can be included as well.

  A spreadsheet with *'intentionally left open'* cells to enable *'what if analysis'* is an example of an *interactive* model as it allows one to *'play'* with the model.

  Spreadsheets, with their traditional numbered columns and rows, are, of course, not the most suitable to capture the structures of a domain in a clear way.
- *Animation* – A model that is represented on a video-based medium system (i. e., a 'movie') that illustrates the dynamic behavior in the modeled domain in terms of the involved agents, subjects, etc.
- *Simulation* – A model that is represented on a simulation engine (as the medium system) and that provides a simulation of the dynamic behavior of the modeled domain.

  If simulation-runs can be generated 'on the fly' based on different scenario's, the simulation (qua model) becomes an interactive model

  Note: a 'screenshot' of an animation or a simulation, can be seen as a model as well. In that case, it would be a view based on a 'temporal distillation'.

5.6 Conclusion

In this paper, we zoomed in on notions such as views, diagrams, programs, animations, specifications, that play an important role in the *modeling practices* that take place in the context of system development (including software engineering, information systems engineering, and enterprise design management). In doing so, we took the view that these notions are be seen as specific kinds of models, albeit for fundamentally different purposes.

In future work, we intend to develop a more completer ontology of models dealing with aspects such as the mereology of models, models as artifacts (i.e., property connecting modeling acts to intentions), identity as aspects of models and how they relate to other artifacts in the ecosystem of modeling (in the spirit of the ontology of software as proposed in Wang et al. (2014). Finally, we intend to investigate the role of different types of *assumptions* (Wang et al. 2016) to modeling practices and to models as artifacts.

References

Apostel, L. (1960). 'Towards the Formal Study of Models in the Non-Formal Sciences'. In: *Synthese* 12, pp. 125–161.

Arbab, F., Boer, F. S. de, Bonsangue, M., Lankhorst, M. M., Proper, H. A. and Torre, L. van der (2007). 'Integrating Architectural Models – Symbolic, Semantic and Subjective Models in Enterprise Architecture'. In: *Enterprise Modelling and Information Systems Architectures* 2.1, pp. 40–57. doi: 10.18417/emisa.2.1.4.

Azevedo, C. L. B., Iacob, M.-E., Almeida, J. P. A., Sinderen, M. v., Pires, L. F. and Guizzardi, G. (2015). 'Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate'. In: *Information Systems* 54, pp. 235–262.

Band, I., Ellefsen, T., Estrem, B., Iacob, M.-E., Jonkers, H., Lankhorst, M. M., Nilsen, D., Proper, H. A., Quartel, D. A. C. and Thorn, S. (2016). *ArchiMate 3.0 Specification*. The Open Group.

Bjeković, M., Proper, H. A. and Sottet, J.-S. (2014). 'Embracing Pragmatics'. In: *Conceptual Modeling – 33rd International Conference, ER 2014, Atlanta, GA, USA, October 27-29, 2014. Proceedings*. Ed. by E. S. K. Yu, G. Dobbie, M. Jarke and S. Purao. Vol. 8824. Lecture Notes in Computer Science. Springer, pp. 431–444. DOI: 10.1007/978-3-319-12206-9_37.

Boar, B. H. (1999). *Constructing Blueprints for Enterprise IT architectures*. New York, NY: John Wiley & Sons.

Boiten, E., Bowman, H., Derrick, J., Linington, P. and Steen, M. (2000). 'Viewpoint consistency in ODP'. In: *Computer Networks* 34.3, pp. 503–537.

Bommel, P. van, Proper, H. A. and Weide, T. P. van der (Nov. 2007). 'Information coverage in advisory brokers'. In: *International journal of intelligent systems* 22.11, pp. 1155–1188. DOI: 10.1002/int.20240.

Campbell, L. J., Halpin, T. A. and Proper, H. A. (1996). 'Conceptual Schemas with Abstractions: Making Flat Conceptual Schemas More Comprehensible'. In: *Data & Knowledge Engineering* 20.1, pp. 39–85. DOI: 10.1016/0169-023X(96)00005-5.

Creasy, P. N. and Proper, H. A. (1996). 'A Generic Model for 3-Dimensional Conceptual Modelling'. In: *Data & Knowledge Engineering* 20.2, pp. 119–161. DOI: 10.1016/0169-023X(95)00043-R.

DoD Deputy Chief Information Officer (2011). *The DoDAF Architecture Framework Version 2.02*. URL: https://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_v2-02_web.pdf (visited on 15/07/2016).

Figueiredo, G., Duchardt, A., Hedblom, M. M. and Guizzardi, G. (2018). 'Breaking into pieces: An ontological approach to conceptual model complexity management'. In: *2018 12th International Conference on Research Challenges in Information Science (RCIS)*. IEEE, pp. 1–10.

Frank, U. (2014). 'Multi-perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges'. In: *Software & Systems Modeling* 13.3, pp. 941–962.

Frank, U. (2002). 'Multi-perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages'. In: *HICSS '02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 3*. Los Alamitos, CA: IEEE Computer Society Press.

Guarino, N., Guizzardi, G. and Mylopoulos, J. (2020). 'On the Philosophical Foundations of Conceptual Models'. In: *Information Modelling and Knowledge Bases XXXI* 321, p. 1.

Guidoni, G. L., Almeida, J. P. A. and Guizzardi, G. (2022). 'Preserving conceptual model semantics in the forward engineering of relational schemas'. In: *Frontiers in Computer Science* 4. DOI: 10.3389/FCOMP.2022.1020168.

Guizzardi, G. (2010). 'Theoretical foundations and engineering tools for building ontologies as reference conceptual models'. In: *Semantic Web* 1.1–2, pp. 3–10.

Guizzardi, G. (2013). 'Ontology-based evaluation and design of visual conceptual modeling languages'. In: *Domain engineering, Product Lines, Languages, and Conceptual Models*. Ed. by I. Reinhartz-Berger, A. Sturm, T. Clark, S. Cohen and J. Bettin. Springer, pp. 317–347.

Guizzardi, G., Ferreira Pires, L. and Sinderen, M. v. (2005). 'An ontology-based approach for evaluating the domain appropriateness and comprehensibility appropriateness of modeling languages'. In: *International Conference on Model Driven Engineering Languages and Systems*. Springer, pp. 691–705.

Guizzardi, G. and Proper, H. A. (2021). 'On Understanding the Value of Domain Modeling'. In: *Proceedings of 15th International Workshop on Value Modelling and Business Ontologies (VMBO 2021), Bolzano, Italy, 2021.* Ed. by G. Guizzardi, T. P. Sales, C. Griffo and M. Furnagalli. Vol. 2835. CEUR Workshop Proceedings. CEUR-WS.org.

Guizzardi, G. (2007). 'On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta) Models'. In: *Frontiers of Artificial Intelligence and Applications*. Vol. 155. IOS Press, p. 18.

Guizzardi, G., Figueiredo, G., Hedblom, M. M. and Poels, G. (2019). 'Ontology-Based Model Abstraction'. In: *13th International Conference on Research Challenges in Information Science, RCIS 2019, Brussels, Belgium, May 29–31, 2019.* Ed. by M. Kolp, J. Vanderdonckt, M. Snoeck and Y. Wautelet. IEEE, pp. 1–13. doi: 10.1109/RCIS.2019.8876971.

Guizzardi, G., Pires, L. F. and Van Sinderen, M. J. (2002). 'On the role of domain ontologies in the design of domain-specific visual modeling languages'. In: *Proc. 2nd Workshop on Domain-Specific Visual Languages*. ACM Press New York.

Guizzardi, G., Sales, T. P., Almeida, J. P. A. and Poels, G. (2021). 'Automated conceptual model clustering: a relator-centric approach'. In: *Software and Systems Modeling*, pp. 1–25.

Harel, D. and Rumpe, B. (2004). 'Meaningful Modeling: What's the Semantics of "Semantics"?' In: *IEEE Computer* 37.10, pp. 64–72. doi: 10.1109/MC.2004.172.

Hofstede, A. H. M. ter, Proper, H. A. and Weide, T. P. van der (May 1992). 'Data Modelling in Complex Application Domains'. In: *Advanced Information Systems Engineering, CAiSE'92, Manchester, UK, May 12–15, 1992, Proceedings*. Ed. by P. Loucopoulos. Vol. 593. Lecture Notes in Computer Science. Springer, pp. 364–377. doi: 10.1007/BFb0035142.

Hoppenbrouwers, S. J. B. A., Proper, H. A. and Weide, T. P. van der (June 2005). 'A Fundamental View on the Process of Conceptual Modeling'. In: *Conceptual Modeling – ER 2005, 24th International Conference on Conceptual Modeling, Klagenfurt, Austria, October 24-28, 2005, Proceedings*. Ed. by L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos and O. Pastor. Vol. 3716. Lecture Notes in Computer Science. Springer, pp. 128–143. doi: 10.1007/11568322_9.

IEEE (Sept. 2000). *Recommended Practice for Architectural Description of Software Intensive Systems*. Tech. rep. IEEE P1471:2000, ISO/IEC 42010:2007. The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE. Piscataway, New Jersey: IEEE Explore, Los Alamitos, California.

ISO/IEC JTC 1/SC 32 Technical Committee on Data management and interchange (1987). *Information processing systems – Concepts and Terminology for the Conceptual Schema and the Information Base*. Tech. rep. ISO/TR 9007:1987. ISO.

Lankhorst, M. M., Torre, L. van der, Proper, H. A., Arbab, F. and Steen, M. W. A. (2017). 'Viewpoints and Visualisation'. In: *Enterprise Architecture at Work – Modelling, Communication and Analysis*. 4th. The Enterprise Engineering Series. Springer, Heidelberg, Germany, pp. 171–214. isbn: 978-3-662-53932-3. doi: 10.1007/978-3-662-53933-0_8.

Levesque, H. J. and Brachman, R. J. (1987). 'Expressiveness and tractability in knowledge representation and reasoning 1'. In: *Computational intelligence* 3.1, pp. 78–93.

Molnar, G. (2003). *Powers: A study in metaphysics*. Clarendon Press.

Moody, D. (2009). 'The "Physics" of notations: toward a scientific basis for constructing visual notations in software engineering'. In: *IEEE Transactions on software engineering* 35.6, pp. 756–779.

Mumford, S. and Anjum, R. L. (2011). *Getting causes from powers*. Oxford University Press.

OMG (2003). *UML 2.0 Superstructure Specification – Final Adopted Specification*. Tech. rep. ptc/03–08–02. Object Management Group, Needham, Massachusetts.

Op 't Land, M., Proper, H. A., Waage, M., Cloo, J. and Steghuis, C. (2008). 'The Results of Enterprise Architecting'. In: *Enterprise Architecture – Creating Value by Informed Governance*. The Enterprise Engineering Series. Springer, Heidelberg, Germany. Chap. 4. ISBN: 978-3-540-85231-5. DOI: 10.1007/978-3-540-85232-2.

Proper, H. A. (2005). *TEE Group – Focus & Drives – Return on modelling effort*. URL: http://www.cs.ru.nl/tee/focus-drives.htm (visited on 20/01/2005).

Proper, H. A. (2009). *Models that matter; Return on Modelling Effort*. Blog. URL: http://erikproper.blogspot.com/2009/02/models-that-matter-return-on-modelling.html (visited on 04/01/2021).

Proper, H. A. (2021). 'On Model-Based Coordination of Change in Organizations'. In: *Engineering the Transformation of the Enterprise: A Design Science Research Perspective*. Ed. by S. Aier, P. Rohner and J. Schelp. Springer, Heidelberg, Germany, pp. 79–98. ISBN: 978-3-030-84655-8. DOI: 10.1007/978-3-030-84655-8_6.

Proper, H. A. and Bruza, P. D. (1999). 'What Is Information Discovery About?' In: *Journal of the American Society for Information Science* 50.9, pp. 737–750.

Proper, H. A. and Guizzardi, G. (2020). 'On Domain Modelling and Requisite Variety – Current state of an ongoing journey'. In: *The Practice of Enterprise Modeling, PoEM 2020*. Ed. by J. Grabis and D. Bork. Vol. 400. Lecture Notes in Business Information Processing. Springer, pp. 186–196. DOI: 10.1007/978-3-030-63479-7_13.

Proper, H. A. and Guizzardi, G. (2021). 'On Domain Conceptualization'. In: *Advances in Enterprise Engineering XIV – 10th Enterprise Engineering Working Conference, EEWC 2020, Bozen-Bolzano, Italy, September 28, October 19, and November 9-10, 2020, Revised Selected Papers*. Ed. by D. Aveiro, G. Guizzardi, R. Pergl and H. A. Proper. Vol. 411. Lecture Notes in Business Information Processing. Heidelberg: Springer, pp. 49–69. DOI: 10.1007/978-3-030-74196-9_4.

Proper, H. A. and Guizzardi, G. (2022). 'Modeling for Enterprises; Let's go to RoME ViA RiME'. In: *PoEM 2022 Forum Proceedings*. CEUR-WS.org.

Proper, H. A., Verrijn–Stuart, A. A. and Hoppenbrouwers, S. J. B. A. (2005). 'On Utility-based Selection of Architecture-Modelling Concepts'. In: *Conceptual Modelling 2005, Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005), Newcastle, NSW, Australia, January/February 2005*. Ed. by S. Hartmann and M. Stumptner. Vol. 43. Conferences in Research and Practice in Information Technology Series. Sydney, New South Wales, Australia: Australian Computer Society, pp. 25–34.

Quillian, M. R. (1968). 'Semantic memory, Semantic Information Processing'. PhD thesis. Massachusetts: MIT.

Razavi, S., Tolson, B. A. and Burn, D. H. (2012). 'Review of surrogate modeling in water resources'. In: *Water Resources Research* 48.7. DOI: 10.1029/2011WR011527.

Romanenko, E., Calvanese, D. and Guizzardi, G. (2022a). 'Abstracting ontology-driven conceptual models: objects, aspects, events, and their parts'. In: *Research Challenges in Information Science: 16th International Conference, RCIS 2022, Barcelona, Spain, May 17–20, 2022, Proceedings*. Springer, pp. 372–388.

Romanenko, E., Calvanese, D. and Guizzardi, G. (2022b). 'Towards Pragmatic Explanations for Domain Ontologies'. In: *Knowledge Engineering and Knowledge Management: 23rd International Conference, EKAW 2022, Bolzano, Italy, September 26–29, 2022, Proceedings.* Springer, pp. 201–208.

Rothenberg, J. (1989). 'The Nature of Modeling'. In: *Artificial Intelligence, Simulation & Modeling.* Ed. by L. E. Widman, K. A. Loparo and N. Nielson. New York, NY: John Wiley & Sons, pp. 75–92.

Sales, T. P., Guarino, N., Guizzardi, G. and Mylopoulos, J. (2017). 'An Ontological Analysis of Value Propositions'. In: *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*, pp. 184–193. doi: 10.1109/EDOC.2017.32.

Sandkuhl, K., Fill, H.-G., Hoppenbrouwers, S. J. B. A., Krogstie, J., Matthes, F., Opdahl, A. L., Schwabe, G., Uludag, Ö. and Winter, R. (2018). 'From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling'. In: *Business & Information Systems Engineering* 60.1, pp. 69–80.

Searle, J. R., Willis, S. et al. (1983). *Intentionality: An essay in the philosophy of mind.* Cambridge University Press.

Sowa, J. and Zachman, J. A. (1992). 'Extending and formalizing the framework for information systems architecture'. In: *IBM Systems Journal* 31.3, pp. 590–616.

Spewak, S. H. (1993). *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications, and Technology.* New York, NY: John Wiley & Sons.

Stachowiak, H. (1973). *Allgemeine Modelltheorie.* Heidelberg: Springer.

Thalheim, B. (2011). 'The Theory of Conceptual Models, the Theory of Conceptual Modelling and Foundations of Conceptual Modelling'. In: *Handbook of Conceptual Modeling.* Springer, Heidelberg, Germany, pp. 543–577.

The Open Group (2011). *TOGAF Version 9.1.* 10th ed. Zaltbommel, NL: Van Haren Publishing.

van't Wout, J., Waage, M., Hartman, H., Stahlecker, M. and Hofman, A. (2010). *The Integrated Architecture Framework Explained.* Heidelberg, Germany: Springer.

Wang, X., Guarino, N., Guizzardi, G. and Mylopoulos, J. (2014). 'Towards an ontology of software: a requirements engineering perspective'. In: *Formal Ontology in Information Systems.* IOS Press, pp. 317–329.

Wang, X., Mylopoulos, J., Guizzardi, G. and Guarino, N. (2016). 'How software changes the world: The role of assumptions'. In: *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS).* IEEE, pp. 1–12.

Zarwin, Z., Bjeković, M., Favre, J.-M., Sottet, J.-S. and Proper, H. A. (July 2014). 'Natural Modelling'. In: *Journal Of Object Technology* 13.3, 4: 1–36. doi: 10.5381/jot.2014.13.3.a4.

Chapter 6

State Consistency – A Matter of Conceptual Modeling?

Elmar J. Sinz

Most conceptual models are expressed as data schemes, some as process schemes and only a few in other modeling languages. Modeling of state consistency, i. e., the goal-oriented alignment of the states of system components does not seem to play a role. However, the more the business world becomes distributed, cooperative, and volatile, this matter gets increasingly important. Has conceptual modeling a 'blind spot' here? The article shows the need for conceptual state consistency modeling. It is based on the concept of compensating transactions, borrowed from data management. Modeling of state consistency is demonstrated using the SOM approach.

6.1 Conceptual modeling and its 'blind spot'

Conceptual modeling has been an important topic in information systems engineering for decades (Frank et al. 2014). In particular, modeling of business reality is used here as a tool for analyzing and designing tasks and resources. The adjective conceptual indicates, on the one hand, that the focus here is on structures that are stable over the longer term and, on the other hand, that these can be seen independently of technical details.[1]. This allows to look at a conceptual model from two perspectives, a domain perspective, and an implementation perspective (Sinz 2021). The domain-oriented perspective describes the real-world context, while the implementation-oriented perspective describes its realization with IT.

Most conceptual models are expressed as data schemes; they capture aspects of the structure of a system. Sometimes process schemes are used, which describe behavioral properties. But what about state consistency, i. e., the goal-oriented alignment of the states of components of a distributed system? State integrity constraints (Schlageter and Stucky 1983, p. 291), formulated in a data schema by means of cardinalities of relationships, only capture the states within the respective system component, but not the overarching states of a distributed and volatile system. Does conceptual modeling have a 'blind spot' here?

One of the long-term development lines of our world is that production of goods and services increasingly takes place in distributed systems. Application systems (IT systems) have changed from monolithic blocks to highly distributed systems, consisting of a multitude of interacting components. The same is true for business operations. Hardly a production process is carried out by a single company; rather many larger and smaller companies are

1 https://www.umo.wiwi.uni-due.de/forschung/forschungsgebiete/konzeptuelle-modellierung/

usually involved. They cooperate toward the goal of producing goods and services. And they are volatile, as companies change over time. Business activity increasingly takes place in so-called ecosystems.

So, why an explicit consideration of state consistency? A simple example is the reconciliation of the financial account of a bank customer with that of the bank. Discrepancies between the two indicate an error, i. e., a lack of consistency of corresponding states, that must be resolved in the interest of both parties. Another example is the matching of an order sent by a customer with the order received by the corresponding supplier. Here, avoiding discrepancies is a prerequisite for smooth business operations.

Other state consistency problems arise at the interface between the service area and the payment area of an IT system. The author himself has experienced the following examples: A ticket vending machine for bus and streetcar tickets collects the money but does not issue a ticket. At a gas station, the transaction is aborted when trying to pay by credit card. The payment is switched to cash, but the credit card is still charged. In both cases, there is obviously a problem of state consistency.

6.2 Scope and understanding of state consistency

For further considerations, the *scope of state consistency* is looked at. The following cases can be distinguished (Sinz 2014):

1. *Business task*. Such a task is, for example, creating and sending a purchase order. State consistency ensures that this task is done completely and is not aborted.
2. *Business transaction.* A business transaction is performed by multiple tasks. To continue the example, the customer's order must be received and correctly processed by the supplier.
3. *Business object*. A business object consists of several tasks. Their attribute values must be consistent with each other. So, the received order must be added to the stock of open orders, otherwise the state of the object will be inconsistent.
4. *Business system.* A business system consists of several interacting business objects. For example, an open order must be passed correctly from the sales object to the production planning object.

The scope of state consistency ranges from a single task to a transaction, an object, and a system. While (1) is typical for a simple database transaction in an application system, (4) expresses the scope focused here. In distributed and cooperative business systems with volatile components, this scope goes far beyond that of conventional database transactions.

Another issue is the *understanding of system consistency.* Can an enterprise that has undelivered orders or unpaid invoices be consistent with respect to its states? If temporary (intermediate) states are to be considered consistent, open items have to be introduced and managed.

If we look at the main interactions of a company on the sales and procurement side as well as the flow of goods/services and the flow of payments, there are four open items: open purchase orders, open delivery orders, receivable items and liability items. These open items are generally accepted in modeling because they pragmatically describe a working company. Therefore, a receivable item for a specific customer does not violate state consistency. On the other hand, it is violated if a purchase order is fulfilled and no receivable item is created.

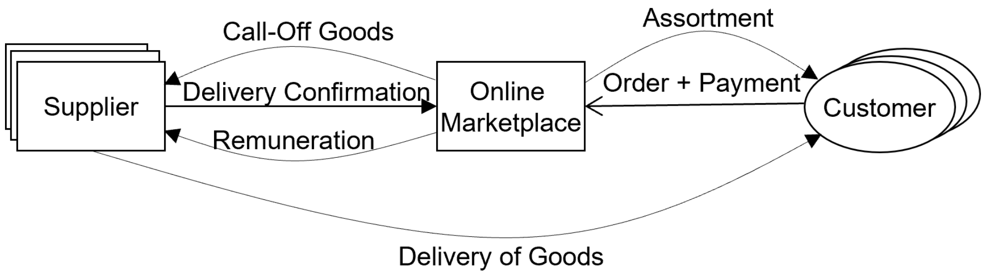


Figure 6.1: *Online marketplace*

6.3 When is modeling of state consistency important?

Modeling of state consistency is not important for all business systems. As mentioned above, there are three characteristics of a business system that, taken together, draw attention to state consistency modeling.

- *Distributed*: A distributed system (Enslow 1978) consists of several subsystems, each of them is autonomous. The subsystems act by themselves, they cannot be controlled from the outside. All subsystems are subject to the goals of the system.
- *Cooperative*: Subsystems work together according to a specific protocol to achieve the goals of the system.
- *Volatile*: Subsystems can change the availability of their services. They can even leave the system while other subsystems come in.

The assumption, that a system with these characteristics always works as planned, cannot be sustained. Deviations from the planned behavior will rather be the normal case. However, if a subsystem cannot provide its service as planned, the state consistency of the overall system is at risk.

The modeling approach required here is an extension of behavioral modeling. It must ask the question, which functions on the objects are necessary to enable state consistency of the entire business system? And what are the processes to restore state consistency to the system when an object does not behave as intended? Since these questions can only be answered from the domain-oriented perspective, they are a matter of conceptual modeling.

An example is an *online marketplace* that presents an assortment for a variety of goods of different suppliers (Figure 6.1). It accepts orders for suppliers and receives the payments from the customers. The orders are passed from the marketplace to the respective supplier, who delivers the ordered goods to the customer and confirms this to the marketplace. The marketplace then remunerates the supplier.

What if the payment goes from customer to the marketplace, but the goods are not delivered by the supplier, maybe they are out of stock, or the supplier has meanwhile left the system? In this case, it is foreseen that the money will be refunded by the marketplace in the same way as the payment was made. Then, the necessary functions of the objects as well as the process must be provided.

The case can be extended. What if the refund does not work, for example, because the customer's credit card has meanwhile expired? In this case, it may be necessary to consult with the customer to determine an alternative payment method. Again, functions and process have to be provided too.

In summary, conceptual modeling of state consistency aims to describe distributed, cooperative, and volatile business systems with respect to the domain-oriented perspective. The modeling of the real-world context will help to better understand these business systems and develop better IT systems to support them.

6.4 Methodological foundation – A borrowing from database transactions

The tool for ensuring state consistency in IT is the concept of transactions[2] known from database systems. A transaction follows the ACID principle (Härder and Reuter 1983), it is

- *atomic* (A), i. e., it is executed completely or not at all,
- *consistent* (C), it satisfies the given integrity constraints,
- *isolated* (I), parallel user orders do not influence each other, and
- *durable* (D), the effect of a completed transaction is persistent.

The realization of the properties A, I and D is based on the concept of sphere of control (Gray and Reuter 1993, p. 174ff.), i. e., the set of states on which the transaction operates. The sphere of control builds up as the transaction is executed and disappears after the transaction becomes persistent.

A simple (flat) transaction is, for example, the creation of a sales order in a central, database-driven application system. Here, the customer data, the article data and the created order enter the sphere of control one after the other. If an error occurs during this process, the transaction is reset (abort transaction), i. e., it is undone. As soon as it is completed and error-free, it becomes persistent (commit transaction) and can therefore no longer be reset.

But what if the transaction is executed in a distributed system with correspondingly distributed application systems? Because of the autonomy and volatility of the subsystems, concepts such as distributed database systems are not applicable. The supplier who takes the specific customer order only once, wants to keep his IT equipment autonomous. He wants not to get involved in interface agreements. Possibly he wants to use no IT support at all.

The sphere of control of the global transaction (related to the entire business system) thus breaks down into several sub-parts (each related to a subsystem) that can become isolated persistent. To still be able to preserve the ACID property of the global transaction, the usage of compensating transactions is necessary (Gray and Reuter 1993, p. 204ff.). These are installed during the execution of a partial transaction. They are triggered if the effect of the partial transaction, although becoming persistent, must be compensated. This means, a partial transaction (DO) is compensated (UNDO) and, if necessary, replaced by another partial transaction (REDO) (Gray and Reuter 1993, p. 538ff.).

The difference between resetting and compensating transactions is explained by an example: With an accounting program a payment of € 820.00 must be booked. However, the amount turns out to be wrong, the correct amount is € 795.00. If the error is corrected before

2 The term transaction is used homonymous in literature. It denotes, on the one hand, a coordinated transfer of an information package or of goods/services (business transaction). On the other hand, it stands for a set of operations following the ACID principle (database transaction).

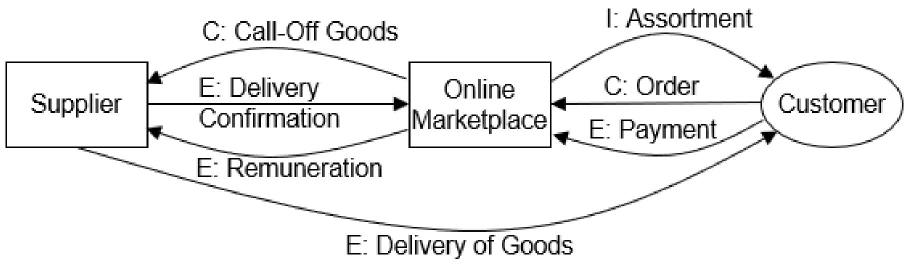


Figure 6.2: Structural view of *online marketplace* using SOM

the transaction has become persistent, it can be reset. Nothing is visible in the accounts or in the general ledger. However, if the error is discovered after the transaction has become persistent (DO), it will be compensated by a cancellation booking (UNDO) and replaced by a correct booking (REDO). The semantic effect of the erroneous booking is thus cancelled, but the error remains visible in the accounts and in the general ledger.

The previous considerations apply to database transactions. But what if you consider real-world transactions. In the example of the *online marketplace*, this is the payment that has already been collected and now must be refunded. In many cases, however, it is not so easy to specify the compensating transaction. As an example, imagine a machining center that performs several machining steps on a workpiece on a single machine. What to do if a hole is already drilled (DO) and then the transaction is to be compensated (UNDO)? Should the hole be filled again or is the workpiece a reject? This question cannot be answered from an IT perspective, but only from a domain-oriented perspective. It is therefore a matter of conceptual modeling.

Another example: A passenger has booked a connecting flight, shows up at the gate, but the plane has already departed. Should the contract be fulfilled by rebooking the passenger on another flight or should the contract be cancelled? Both alternatives are probably associated with costs. Since the case occurs more frequently and can become arbitrarily complex, it is analysed and supported by a conceptual model.

Summarizing, the borrowing from database transactions is the concept of compensating transactions, adopted and extended to real-world transactions.

6.5 An example using the Semantic Object Model

The Semantic Object Model (SOM) is a comprehensive approach to modeling business systems (Ferstl and Sinz 1995; Ferstl and Sinz 2006; Ferstl and Sinz 2013). Business process models are an important part of SOM. A business process model is represented in two views, a structural view, and a behavioral view.

The *structural view* shows the business objects and their relationships (Figure 6.2). For system delimitation, the objects are divided into discourse world objects and environmental objects. The relationships between the objects are business transactions that coordinate the objects. Two forms of coordination are distinguished: non-hierarchical coordination according to the ICE principle (initiating, contracting, and enforcing transaction) and hierarchical coordination according to the CF principle (control and feedback transaction).

Objects and transactions can be refined hierarchically. Objects can be decomposed into sub-objects. The same applies to transactions. For example, an enforcing transaction (E) can be decomposed into an initiating (I), a contracting (C), and an enforcing (E) sub-transaction according to non-hierarchical coordination. During the initiating sub-transaction, the objects exchange information on the goods or services to be delivered. The objects conclude a contract while doing the contracting sub-transaction. They exchange the goods or services while running the enforcing sub-transaction. All this is done with respect to the goal of the global enforcing transaction.

The *behavioral view* shows the tasks assigned to business objects and their relationships (Fig. 6.3). All tasks of an object are based on one and the same object-internal memory. Task relationships between objects correspond to business transactions. The relationship of tasks within an object are based on object-internal events. The behavioral view is basically based on the Petri net concept (Reisig 2010), whereby this is extended and interpreted semantically.

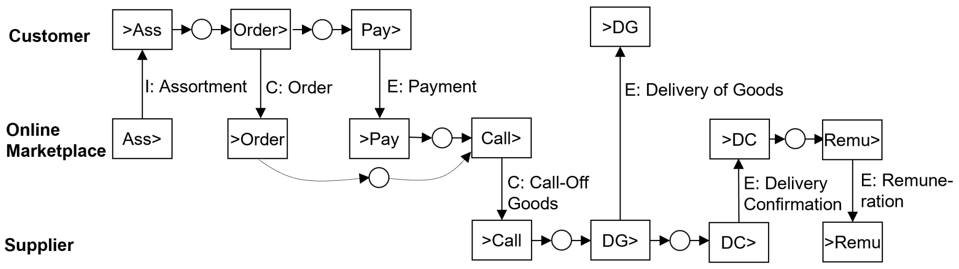


Figure 6.3: Behavioral view of *online marketplace* using SOM

The Semantic Object Model has several features that are useful for modeling the state consistency of business systems:

- Business processes with structural and behavioral views: In the behavioral view the tasks are assigned to objects.
- Binary business transactions: Business transactions are executed by the tasks of two business objects, thus building one database transaction.
- Decomposition of objects and transactions: If an *E* transaction is decomposed according to the ICE principle, the semantics of the whole transaction is transferred to its parts.

Figure 6.4 shows the behavioral view of the *online marketplace* with spheres of control and compensating transactions for the assumed case that a supplier cannot perform the delivery of goods (symbol lightning). Hence, the state consistency of the business system is violated. The handling of the exception, which is an error from the perspective of the whole business system, is now explained:

- At the time of the error, the spheres of control of the objects *customer*, *online marketplace* and *supplier* are partially established (highlighted in light grey), but not completed. They must therefore be reset.
- The tasks that have already become persistent are shown with thick border. Their spheres of control, which were embedded in the global sphere of control, have disappeared. These tasks must be compensated.

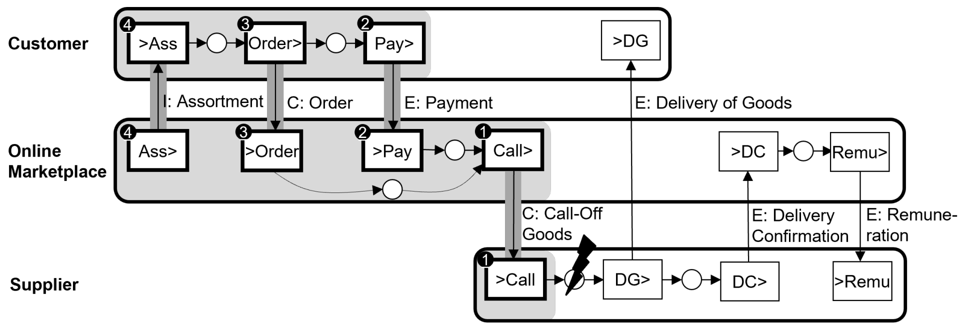


Figure 6.4: *Online marketplace* with spheres of control and compensating transactions

- However, each task of a business object performs a business transaction with a task of another object. So, the two tasks can only be compensated together, they are coupled (dark grey bar between two tasks). For example, the compensation of *>Call* in the object *supplier* also causes the compensation of *Call>* in the object *online marketplace* (step 1).
- The compensation of *Call>* triggered in the object of *online marketplace* leads to further compensation steps here (steps 2 and 3).
- The initiating transaction (step 4) was not considered necessary for compensation, because neither of the two objects entered into an agreement. Its treatment serves the continuation of the process.

In total, four steps are necessary to handle the error 'supplier cannot perform delivery of goods', involving eight tasks (see Table 6.1). Three of these steps are compensations (UNDO), one is used to restart the system (REDO). After these tasks are executed, the state consistency of the system is re-fulfilled, and the continuation of the business activity is initiated.

These tasks were revealed by conceptual modeling of state consistency. It concerns conceptual modeling in the sense used here because the semantics can only be derived from the domain-oriented perspective. The modeling approach and its borrowing from database transactions focus the modeler's attention on the relevant issues.

Step	Task	Description of the task	Type
1	>Call	Reject of call-off goods	UNDO
1	Call>	Register the rejection of call-off goods	UNDO
2	>Pay	Refund of payment to customer	UNDO
2	Pay>	Register the refund	UNDO
3	>Order	Reject of sales order	UNDO
3	Order>	Register the rejection of sales order	UNDO
4	Ass>	Submit an alternative offer to the customer	REDO
4	>Ass	Register the alternative offer	REDO

Table 6.1: Restoration of state consistency after 'supplier cannot perform delivery of goods'

6.6 Conceptual modeling of state consistency – Lessons learned

In general, a conceptual model serves to identify features of a system that would not have been recognized, or not as clearly, without a model. Conceptual modeling of state consistency is about identifying and embedding those functions in process flows that are required to restore the global state consistency of the system when a subsystem has not behaved as planned. If the UNDO function is required, then the model must say how. This is especially important for distributed, cooperative, and volatile systems.

In addition to these execution time aspects, conceptual modeling of state consistency is also helpful at design time. An example is the execution of a travel booking, consisting of flight, car rental and hotel booking. The trip is only offered in full or not at all. In this case the system design is supported by the expected risk for unavailability of a travel component and the cost of rebooking (UNDO/REDO) of already booked components.

Overall, it is expected that through the conceptual modeling of state consistency, a better understanding and more systematic investigation of the business side of a system is achievable. This leads incidentally to specifications of more robust and complete IT systems. Since the modeling can only be done considering the domain-oriented perspective, it is a matter of conceptual modeling.

References

Enslow, P. H. (1978). 'What is a "Distributed" Data Processing System?' In: *IEEE Computer* 11.1, pp. 13–21.

Ferstl, O. K. and Sinz, E. J. (1995). 'Der Ansatz des semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen'. In: *WIRTSCHAFTSINFORMATIK* 37.3, pp. 209–220.

Ferstl, O. K. and Sinz, E. J. (2006). 'Modeling of Business Systems Using SOM'. In: *Handbook on Architectures of Information Systems.* Ed. by P. Bernus, J. Blazewics, G. Schmidt, M. Shaw, P. Bernus, K. Mertins and G. Schmidt. International Handbooks on Information Systems. Berlin, Heidelberg: Springer, pp. 347–368.

Ferstl, O. K. and Sinz, E. J. (2013). *Grundlagen der Wirtschaftsinformatik.* 7th ed. Berlin: De Gruyter Oldenbourg.

Frank, U., Strecker, S., Fettke, P., Brocke, J. vom, Becker, J. and Sinz, E. (2014). 'The Research Field "Modeling Business Information Systems" (Research Note)'. In: *Business & Information Systems Engineering* 6.1, pp. 39–43.

Gray, J. and Reuter, A. (1993). *Transaction Processing: Concepts and Techniques.* San Mateo, CA: Morgan Kaufmann.

Härder, T. and Reuter, A. (1983). 'Principles of Transaction-Oriented Database Recovery'. In: *ACM Computing Survey* 15.4, pp. 287–317.

Reisig, W. (2010). *Petri-Netze. Modellierungstechnik, Analysemethoden, Fallstudien.* Wiesbaden: Vieweg und Teubner.

Schlageter, G. and Stucky, W. (1983). *Datenbanksysteme: Konzepte und Modelle.* 2., neubearb. und erw. Aufl. Teubner-Studienbücher. Stuttgart: Teubner.

Sinz, E. J. (2014). 'Konzeptuelle Modellierung der Zustandskonsistenz verteilter betrieblicher Informationssysteme'. In: *Modellierung 2014, 19.-21. März 2014, Wien, Österreich.* Ed. by H. Fill, D. Karagiannis and U. Reimer. Vol. P-225. LNI. GI, pp. 241–256.

Sinz, E. J. (2021). 'Five questions to be clarified before starting to model conceptually'. In: *EMISAJ – International Journal of Conceptual Modelling* 16, 3:1–3:4.

Acknowledgement

Thanks to Felix Härer (University of Fribourg, Switzerland) and to Dominik Bork (Technical University Vienna) for reading and commenting on an earlier version of this article.

Chapter 7

Epistemologically Motivated Modeling of Computer-Integrated Systems

Peter Fettke and Wolfgang Reisig

This paper frames and positions modeling in science and in engineering as an epistemological problem. Three requirements concerning structure, objects, and behavior that each modeling technique for computer-integrated systems must fulfill are identified and formulated as postulates. Corresponding modeling concepts are identified and epistemologically motivated. Furthermore, they are integrated into a comprehensive infrastructure for all basic tasks necessary to model computer-integrated systems.

7.1 Modeling in Science and Engineering as an Epistemological Problem

7.1.1 Modeling as an Epistemological Concept

Modeling in science and in engineering is characterized by models for *processes*, that is, the description of ‘changes in time’. Usually, the real numbers are used as a time axis. The reason for the use of real numbers is the numerous properties of the modeled processes that can be derived from a model with the help of continuous functions, differential equations, and everything else known as ‘the calculus’.

Informatics (or computer science) can be contrasted and characterized as the science of anything digital: Changes of facts (processes) are understood as discrete steps and not as continuous, gradual transitions. Facts of reality itself are represented symbolically. Is there a comparatively epistemologically well-founded, generic, comprehensive and outstanding modeling method for digital processes, analogous to the ‘calculus’ of the natural sciences, with deep-seated structural analysis methods? The remainder of this paper explains and motivates such a modeling method.

7.1.2 Modeling in Informatics from a Technical and a User’s Point of View

Before going into details, we consider two approaches to the understanding of informatics as a science. Informatics emerged from the capabilities of *computational technology*. On a physical basis, computing technology offers a digital surface on which character strings are stored and transformed according to certain rules, and with the help of software. For this purpose, a multitude of programming and specification languages, databases, et cetera have been developed and used for decades. They are employed to meet specific aims, where the aims themselves remain outside the formal framework. All these concepts have a common epistemological basis: the theory of computable functions and anything typically subsumed under ‘theoretical computer science’. Alternatively, informatics can be developed from its *application side*, by adequate descriptions of systems of real-world processes and of digitally represented facts, especially when real-world as well as symbolic facts and steps are interwoven. A typical example is a model of the behavior of a company, for instance a retailer, with ordering, delivery, payment departments for goods, corresponding updating of databases, et cetera. Such a model can be quite formal. For example, in a formal model of the company, it can be proved that every ordered item will eventually be delivered and that every delivered item has been ordered beforehand. From this conception of informatics, we develop an epistemologically well-founded, generic, comprehensive modeling method.

7.1.3 Three Postulates on Modeling Computer-Integrated Systems

A theoretical basis to bridge the gap between computation technology and its applications must meet various requirements. We formulate what we consider the central requirements, in terms of three postulates:

Postulate 1. *A comprehensive model of computer-integrated systems is* structured. *A large computer-integrated system is not amorphous, but is composed of a number of sub-systems. In one sentence:* Composition matters!

Postulate 2. *A model includes both* data *and* real life objects, *for instance, products, customers, machines, paper money, contracts, et cetera. In one sentence:* Objects matter!

Postulate 3. *A model is* locally *updated. Behavior cannot be understood as sequence of steps in one, ‘global’ state space. Instead, every change of the system has its* local causes *and* local effects. *Behavior evolves from local events. Some events are causally related; others are not. In one sentence:* Causality matters!

7.2 Epistemologically Motivated Modeling Concepts

On the background of the above three postulates, we suggest a modeling technique that is motivated and justified by epistemological arguments. It combines well established and more recently suggested concepts.

7.2.1 The Three Dimensions of Formal Models in Informatics

It is undisputed that static and dynamic aspects are fundamental dimensions of digital systems. As a third dimension we suggest *architecture*: A large system does not simply consist of an amorphous mass of data and events, but is partitioned into *modules* which are designed and composed in a meaningful way. It is quite possible to identify several alternative architectures for a system, based on different criteria and aspects. Thus, *statics*, *dynamics*, and *architecture* are three dimensions for the understanding of a computer-integrated, digital system (Figure 7.1). It has frequently been discussed that these three dimensions are necessary (Fettke and Reisig 2021a; Fettke and Reisig 2021b). Practice must show that they are also sufficient as a starting point to systematically classify all interesting aspects. The Heraklit case studies constructed so far support this assumption. In the remainder of this paper, we show how the three dimensions can be adequately conceptualized and how they cohere. In particular, we highlight where and why Heraklit uses novel concepts.

7.2.2 First Dimension: Architecture

It has frequently been suggested to compose a large real-world system out of *modules.* A module then has a *interface*, consisting of labeled elements. Two modules A and B are composed to a module $A \bullet B$ by merging identically labeled elements of the two interfaces to one element of the composition $A \bullet B$, which then moves inside $A \bullet B$. This composition operator is fundamentally flawed, because it is not associative: in general, $A \bullet (B \bullet C)$ and $(A \bullet B) \bullet C$ are different. The composition of many modules $A_1, \dots, A_n$ must therefore be parenthesized. For example, one cannot simply split them into two modules $A_1 \bullet \dots \bullet A_i$ and $A_{i+1} \bullet \dots \bullet A_n$. Hence, in general, meaning preserving exchange of a module A_i by a module B is a challenging endeavor.

Heraklit builds on the composition calculus by Reisig (2019). The surface of a module A consists of a *left* and a *right* interface, written $^*A$ and A^*, respectively. The interior of a module is not specified further; for describing behavior, Petri nets are recommended. An *abstract* module consists of its two interfaces only, or has only its name inside. There can also be software inside a module, which, for example, sends remote procedure calls along its interfaces, and receives corresponding answers. Note that the interior of a module can even contain text documents or other informal objects such as so-called boundary objects or knowledge objects, typically discussed in (management) information systems research as important links between the real and the formal world.

A given set Λ of labels defines the set $M(\Lambda)$ of all modules with interface labels in Λ. Two modules A and B are composed to one module $A \bullet B$ in $M(\Lambda)$ by merging identically labeled elements of A^* and $^*B$ to an element of $A \bullet B$, which then moves inside $A \bullet B$. Hence, composition is a *total* operation on $M(\Lambda)$. It is fundamental to the use of Heraklit, that this composition is *associative*, too. Algebraically formulated, for a given set Λ of labels, the set $M(\Lambda)$ of all modules with interface labels in Λ is a *monoid.* So, $M(\Lambda)$ behaves exactly like the structure of the set $L(\Lambda)$ of all words (symbol strings) over Λ. Such words and sets of words (languages) are often conceived as the beginning and the core of the foundations of informatics. Instead, we suggest to take modules for this purpose.

The partition of the surface of a module into a left and a right interface is extremely useful for practical applications. To cover the cases where in a composition $A \bullet B$ a merged element is intended not to move into the interior of $A \bullet B$ but should remain on the (left or right) interface of $A \bullet B$, left and right interfaces of a module are not necessarily disjoint. This fundamental concept allows to easily construct hierarchies and to integrate not only formal, but also semi- and informal descriptions into one model.

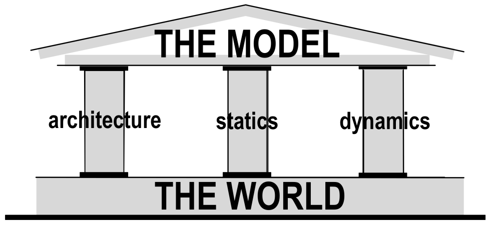


Figure 7.1: The three pillars of Heraklit

No other composition operator for modules or comparable constructs is total and associative at the same time; hence, no other composition operator exhibits a similar degree of flexibility and expressive power.

7.2.3 Second Dimension: Statics

In the systematic development of mathematics and logic in the 20th century, *structures* have become generally accepted: A structure consists of finitely many sets, some distinguished elements of these sets, so-called *constants*, and some operations on these sets. The *arity* of an operation f describes from which sets the arguments $a_1, \dots, a_n$ for f come, and in which set the result $f(a_1, \dots, a_n)$ lies.

The sets and operations of a structure can be chosen freely. This differs from usual informatics, where structures are usually limited to finite or infinite sets of symbol sequences. In particular, real-world sets and functions can be included, such as:

- the set of potential customers of a bank,
- a function which assigns a database entry to each real customer,
- a function which assigns a car to its owner or current driver,
- a function which assigns a digital stamp to a matrix code and vice versa for identification (Fettke and Reisig 2022a).

It also includes extreme cases, for example:

- uncountable sets whose elements cannot be written down completely, for example the real numbers;
- functions which are not computable, for example, deciding whether a Turing machine terminates (the halting problem).

Heraklit is based on this general concept of sets and functions. In particular, real-world objects can be captured in Heraklit and be included in formal arguments.

Here, a fundamental quest for a useful kind of symbolic representation of elements and components of such structures arises. As usual, a *signature* Σ for a structure S contains for each set, each constant and each operation of S a symbol. Such a symbol also represents the arity of the corresponding element. On this basis, *ground terms* can be formed. Each ground term can be "calculated" and thus describes an element of a set of S. To this end, every constant symbol is replaced by the corresponding constant and every function symbol by the corresponding function. In a *general term*, variables can stand in place of constants. In order to calculate a general term, each variable must first be assigned an element of S. This way, a general term has a value with respect to an assignment. The central idea of this procedure: A signature contains only finitely many symbols, but generates infinitely many terms. By first assigning a meaning to each symbol of the signature, the inductive structure of the terms then provides a meaning for each term.

Often a structure S is extended by relations, also called *predicates*, resulting in a *relational structure*. A relation is a set of tuples $(a_1, \dots, a_n)$, with elements a_i from sets of S. Predicates p are often used to describe properties of elements or of sets of elements. This justifies the phrasing that p *applies* to a tuple $(a_1, \dots, a_n)$ if $(a_1, \dots, a_n)$ is an element of p. The *extension* of a predicate p is the set of elements to which p applies; that is, the set of elements of p. An example in a business administration is the predicate *available managers*, with the set of available customer managers as its extension.

Additionally, a signature of a relational structure contains for each relation a symbol. The signatures of a structure differ only in the choice of symbols. Conversely, the structures with the same signature Σ ('Σ-structures') generally differ considerably.

Σ-relational structures are the semantic basis of many mathematical fields, but especially of predicate logic. To talk about properties of elements and of sets of elements, predicate logic uses predicates, applied to terms with variables, and the two quantifiers 'for all' and 'there exists'. Furthermore, such propositions are connected using propositional operators 'and', 'not', et cetera. Predicate logic is an approved basis for the formulation and proof of properties for relational structures.

For the systematic description of data and objects, informatics has long used Σ-structures for object-orientation, abstract data types and algebraic specifications. In this context, Σ-terms represent objects. Frequently, textual arrangements, keywords, et cetera are employed, rendering the basic form of the Σ-structures less visible. For example, a database with its tables and the usual database operations specify a structure. The abstract syntax of a program is a term of a Σ-structure. Heraklit also uses Σ-structures. Two aspects are of particular importance:

1. The meaning of a Σ-term is generally, in fact, a real-world object, not just a symbolic representation of an object.
2. Heraklit uses predicates of a relational structure not (primarily) to describe relations between objects, but to describe *local* states: a local state of a model of a system is a predicate p together with an object o to which the predicate applies. An example from a business model is the predicate p: *customers waiting* in the entry hall of a bank branch. A corresponding object o is the customer Bob. If he sits in the waiting area of the entry hall, he belongs to the extension of p. Graphically, a local state is represented as a Petri net place: a circle or ellipse, labeled p and o.

7.2.4 Third Dimension: Dynamics

A computer-integrated system is typically modeled as a *transition system*: A transition system consists of a set of states and steps between states, representable as a directed graph. Often, an initial state is distinguished. A behavior of the system is then a finite or infinite path through the graph, starting from the initial state. For small systems, this approach is useful. For example, the essentials of the functionality of an automated teller machine (ATM) can be described sufficiently precisely as a transition system with about 10 states. However, transition system models for large systems have a number of weaknesses:

- State explosion: How to model a system G, consisting of two ATMs G_1 and G_2 standing side by side and accessing the same money store? With Z_i as the set of states of G_i $(i = 1, 2)$, a state of G may be conceived as a tuple (z_1, z_2), representing a state z_i of each of the two machines G_i. The set of states of G is then the Cartesian product $Z_1 \times Z_2$. The model G has about $10 \cdot 10 = 100$ states. A larger bank branch may have five cash machines installed; the model of this system then has $10^5 = 100,000$ states. German banks currently have about $86,000$ cash machines installed; the model of their common state space thus has about $10^{86,000}$ states. For comparison, the universe consists of about 10^{80} molecules.
- Measuring: A large system includes events whose immediate preconditions and effects are not related; that is, they occur independently of each other. An example is two

customers entering their respective PINs at two different ATMs. In a transition system, these two events are represented in sequence; thus, there is a global state in which one of the pins has already been entered and the other has not yet been entered. Now, how is this decided in a concrete case? Which order is valid? Which intermediate state has been reached? For every technical decision mechanism (for example clocks), one can construct two events that are so close in time that this mechanism fails. The objection that in such a case the order is irrelevant anyway is correct. But then such sequences should not be modeled at all. Nevertheless, most of the modeling methods work with such sequences.

- Intuitive understanding: The behavior of a really large system, for example a company or a cyber-physical system, can not be understood 'as a whole'. At best, the behavior of single components can intuitively be conceived, without identifying global states.
- Modern physics: The concept of global states requires a concept of time that is available everywhere and immediately in any precision (Fettke and Reisig 2022b). This contradicts insights of modern physics.

Heraklit solves this problem in a very unusual and fundamental way: In a model of a large system, global states are entirely avoided. As a consequence, behavior is no longer modeled as a sequence of steps between states of one, global state space. This raises the question of how to model behavior without a global state space.

For this purpose we use *local* states: We take a local state as a predicate p, which currently applies to an object o, that belongs to the extension of p. Dynamic behavior is then described as updating local states. In its most elementary form: an object leaves or reaches the extension of a predicate. As an example, let the predicate *present* describe the set of customers present in the entry hall of a bank. A step then describes, for example, that the customer Bob enters the room: Bob reaches the extension of *present*.

In general, one step updates several local states instantly. Technically, a step is formulated as the firing of a transition of a Petri net. An entire behavior, a run, of a system is composed of such steps. An example is the service provided to two customers, Alice and Bob, by the account manager of a bank:

- Step a: Alice enters the entry hall of the bank.
- Step b: The account manager Ron sends Alice to advisor A.
- Step c: Alice goes to A.

These steps are ordered. Representing immediate predecession and succession by <, the result is $a < b < c$. In addition, there are the steps of Bob:

- Step d: Bob enters the entry hall of the bank.
- Step e: The account manager Ron sends Bob to advisor B.
- Step f: Bob goes to B.

This set of steps is ordered, too: $d < e < f$. The account manager Ron serves the two customers in order, Alice before Bob. Thus b and e are also ordered: $b < e$. The steps $a, \dots, f$ together form a behavior, a run. However, this run is only partially ordered; for example, c and d are not ordered. The run is non-sequential (distributed). In general, a system can potentially have many such partially ordered runs.

7.3 Conclusions

7.3.1 The Fundamental Concepts of HERAKLIT

With relational structures, HERAKLIT uses a generic and comprehensive concept for the description of static aspects, following the model of predicate logic. New, but intuitively obvious, is the conception of a local state as a predicate that applies to an object. From this follows almost inevitably the description of dynamics as an update of a local state, that is, an object leaves or reaches a predicate. This can be extended slightly: One step may update several local states at once. The formal technical manifestation of this idea is the occurrence of a transition of a Petri net.

Also new is the *composition calculus* for the description of composition, refinement, hierarchy formation, et cetera of modules. With its strict, formal version of interfaces and the composition of modules as well as its minimal assumptions about the interior of modules, the formal apparatus is also extremely expressive and extremely flexible. The monoid structure, in analogy to formal languages, again emphasizes the universal conception of the formalism.

The main objective of informatics is to build a bridge between computational technology and applications. For the area of computational technology there is a canonical theoretical and conceptual basis: the computable functions with Church's thesis, complexity theory, et cetera. From the perspective of applications there is nothing comparable so far.

HERAKLIT now proposes three postulates for the modeling of systems, that any reasonable scientist would accept (cf. Section 7.1.3). We like to put forward the following conjecture:

Conjecture. *If a system* S *satisfies the three postulates of Sec. 7.1.3, then* S *can be modeled in the framework of* HERAKLIT.

This conjecture is not yet proven. However, the available case studies and our intuition clearly show that this conjecture holds. Furthermore, Gurevich (2000) already proved a comparable thesis for sequential systems, based on the idea of *abstract-state machines* (ASM). We have good reasons to believe that a similar property holds for distributed systems and HERAKLIT.

7.3.2 An Epistemological Justification of HERAKLIT

The fundamental novelty of HERAKLIT becomes apparent in a comparison with the *firewall* concept that Dijkstra propagated over decades (Dijkstra 1989). This firewall separates two worlds of informatics: on one hand is the formal world with the correctness problem of informatics, in which the correctness of an implementation with respect to a specification is purely a problem of proving mathematical theorems. On the other hand is the informal world with the 'pleasantness problem' of the real world, where the adequacy of a formal specification or an implementation is discussed with psychological and experimental means. Because of these fundamentally different methodological approaches, Dijkstra advises to strictly separate the two worlds.

In contrast, HERAKLIT follows the tradition of logic, where formal arguments are entangled with real-world facts: Predicate logic organizes real-world and imagined objects in relational structures and represents them with terms of a corresponding signature. Properties and assertions about real-world and imagined objects are then formally formulated using the terms of the signature, together with the predicates, quantifiers and propositional operations.

Perhaps the best known example is the logical conclusion: *All humans are mortal; Socrates is a human, therefore Socrates is mortal.* This conclusion is not less formal than the statement $x < x^2$ for all $x > 1$. Heraklit represents and mutually relates facts and statements of (business) informatics, artificial intelligence, and information systems, but also far beyond, in terms of formal logic.

Then, however, Heraklit adds a fundamentally new aspect: The extension of predicates can be updated. This is achieved by means of steps, formally formulated with the terms of a signature, as well as with concepts of Petri nets. This makes it possible, for example, to formally prove in a model of a retail company that for each article ordered by a customer, a suitable product is eventually delivered. Generally, the formulation of static aspects of a system model with predicate logic is extended in Heraklit by dynamic aspects of real-world systems. Thus Heraklit integrates formal arguments into the discourse on real-world objects and their behavior. Thus, Heraklit has an epistemological basis that is scientifically, systematically and methodologically justified; unlike almost any other modeling method for computer-integrated systems and for digital sciences in general.

References

Dijkstra, E. W. (1989). 'Reply to comments'. In: *Commun. ACM* 32.12, p. 1414.

Fettke, P. and Reisig, W. (2021a). 'Handbook of Heraklit'. Heraklit working paper, v1.1, September 20, 2021, `http://www.heraklit.org`.

Fettke, P. and Reisig, W. (2021b). 'Modelling Service-Oriented Systems and Cloud Services with Heraklit'. In: *Advances in Service-Oriented and Cloud Computing*. Ed. by C. Zirpins, I. Paraskakis, V. Andrikopoulos, N. Kratzke, C. Pahl, N. El Ioini, A. S. Andreou, G. Feuerlicht, W. Lamersdorf, G. Ortiz, W.-J. Van den Heuvel, J. Soldani, M. Villari, G. Casale and P. Plebani. Springer, pp. 77–89.

Fettke, P. and Reisig, W. (2022a). 'Breathing Life into Models: The Next Generation of Enterprise Modeling'. In: *Proceedings of the 17th International Conference on Software Technologies (ICSOFT 2022)*. Ed. by L. M. Hans-Georg Fill Marten van Sinderen. SCITEPRESS – Science and Technology Publications, pp. 7–14.

Fettke, P. and Reisig, W. (2022b). 'Discrete Models of Continuous Behavior of Collective Adaptive Systems'. In: *Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning*. Ed. by T. Margaria and B. Steffen. Springer, pp. 65–81.

Gurevich, Y. (2000). 'Sequential abstract-state machines capture sequential algorithms'. In: *ACM Transactions on Computational Logic* 1 (1), pp. 77–111.

Reisig, W. (2019). 'Associative composition of components with double-sided interfaces'. In: *Acta Informatica* 56.3, pp. 229–253.

Chapter 8

Executable Multi-Level Modelling: Establishing Foundations, Methods and Tools

Tony Clark

System modelling has been a cornerstone of Information System Engineering for over 30 years. It aims to provide a basis for systematic representation, abstraction and analysis of system properties. Meta-modelling and Domain Specific Modelling both enhance system modelling through the use of abstraction at the type level. Until recently, there has been little attention to execution, methodology and structure at the type level of modelling. Ulrich Frank has made significant contributions to the field of *executable multi-level modelling* with the aim of addressing this gap. This paper reviews these contributions and discusses current and future challenges.

8.1 Introduction

System modelling has been a cornerstone of Information System Engineering for over 30 years. It aims to provide a basis for systematic representation, abstraction and analysis of system properties. Meta-modelling and Domain Specific Modelling both enhance system modelling through the use of abstraction at the type level. Until recently, there has been little attention to execution, methodology and structure at the type level of modelling. Ulrich Frank has made significant contributions to the field of *executable multi-level modelling* with the aim of addressing this gap. This chapter reviews these contributions and discusses current and future challenges.

The rest of this chapter is organised as a series of short sections introducing the key features of executable multi-level modelling with reference to the key contributions made by Ulrich Frank. The chapter is not intended to be self contained, nor will it contrast the contributions with those of other researchers (some of whom present significantly different approaches to MLM). It represents a review of the key contributions of a major contributor to an important field with links to publications for the interested reader to explore the details.

8.2 Multi-Level Modelling

The field of Multi-Level Modelling (MLM) can be traced back over 20 years (Atkinson and Kühne 2001) where it was introduced to address shortcomings in standard approaches to modelling as exemplified by the Unified Modelling Language (UML).

The field is brought up to date by Frank (2022) which identifies MLM as aiming to reduce *accidental complexity* arising from a lack of expressive power in standard modelling languages when representing domain concepts. This limitation arises due to the semantics of the standard inheritance relationship represented in Frank (2022) as shown in Figure 8.1 where we would like to represent the abstraction PeripheralDevice that defines certain properties for instances. The type Printer can be viewed in two ways: it is a specialisation of PeripheralDevice since it can add new instance-level properties; however, it is also an instance of PeripheralDevice since it can provide values for some of the properties. The same argument applies to CPL-844 which can be viewed as a specialisation and an instance of Printer.

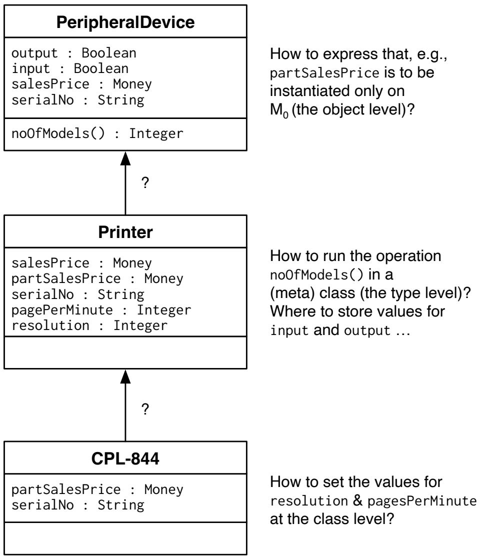

PeripheralDevice
output : Boolean
input : Boolean
salesPrice : Money
serialNo : String
noOfModels() : Integer
How to express that, e.g., partSalesPrice is to be instantiated only on M_0 (the object level)?
?
Printer
salesPrice : Money
partSalesPrice : Money
serialNo : String
pagePerMinute : Integer
resolution : Integer
How to run the operation noOfModels() in a (meta) class (the type level)? Where to store values for input and output ...
?
CPL-844
partSalesPrice : Money
serialNo : String
How to set the values for resolution & pagesPerMinute at the class level?

Figure 8.1: Limitations of Standard Inheritance

The relationships shown as question marks (?) in Figure 8.1 represent this new category of type-relationship. Standard approaches such as UML do not have the ability to represent both *instance-of* and *specialisation-of* at the same time. However this is required in order to avoid accidental complexity.

Why would we want to avoid the accidental complexity in such a manner? An underlying aim of MLM is to promote domain information to the highest level of abstraction possible, thereby achieving the greatest reuse possible across a number of applications. Essentially this allows the construction of *domain specific languages* that are reusable. Figure 8.2 (taken from Frank et al. 2017) compares the reuse of three exemplary concepts to represent printers (`Class`, `Device`, `Laser-Printer`) using standard non-MLM approaches (based on a single language abstraction: `Class`) shown as the curve from a higher scale of reuse (for `Class`) to a lower degree of specificity, to that of MLM approaches shown as the curve from lower scale of reuse to higher degree of specificity (for `Laser-Printer`). It can be seen that the standard approaches rapidly decrease in the scale of reuse as the specificity increases (classes become more and more *brittle*) whereas a MLM approach allows domain specific concepts to be defined at the type level which does not compromise reuse.

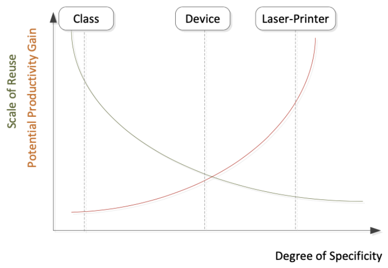


Figure 8.2: Range of Reuse and Productivity of Reuse (Frank et al. 2017)

In order to provide the benefits of MLM, Frank developed the modelling language FMMLx (Flexible Multi-Level Modeling and Execution Language) (Frank 2014). An example of FMMLx is shown in Figure 8.3. Each class is shown as being an instance of a particular meta-class. For example, Printer is an instance of the meta-class PeripheralDevice. In addition to its meta-class, a class is defined on a particular *level* of classification. For example, the class Printer is defined on level 2 and the class PX_66 is defined on level 1. The properties defined by a class are also given levels. The level of a property-definition must be strictly less than the level of its class and define at which point the property can be associated with a value. For example, the property `output` has the level 2 which means that it is associated with a value in an instance of PeripheralDevice at level 2. This allows the language of peripheral devices to specify whether types are associated with an output or not. In the case of Printer, the value is `true` meaning that a printer produces output. Compare the property-definition `output` with that of `serialNo` which has the level 0 meaning that only ground-instances (for example, X649Q) have a serial number. Together, the level numbers and type-relationships, allow FMMLx to achieve the aims of avoiding accidental complexity and reuse as shown in Figure 8.2.

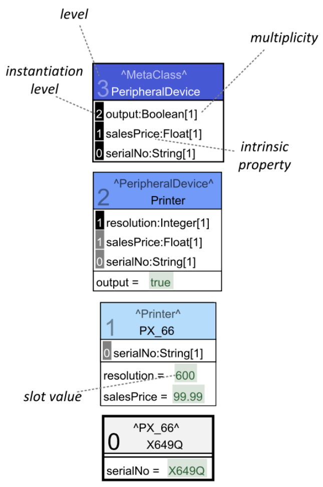


Figure 8.3: FMMLx Example

MLM can be favourably compared to traditional object-oriented modelling in terms of a framework of features such as *classification*, *generalisation*, *encapsulation*, *polymorphism* and *composition* (Frank 2022). Twenty years of research has led to several current approaches to MLM. Whilst all of these approaches have common features in terms of aiming to achieve abstraction and involving multiple type levels and representing types and instances, they differ in terms of the detail. Several researchers continue to aim to produce a unified foundation for the field (Jeusfeld and Frank 2021).

8.3 Execution and Relationship to Low-Code

Modelling approaches can be used to organise information about a system or a domain. Model Driven Engineering (MDE) introduced the idea of models being used as part of the development process leading to executable systems. Initially, this was achieved through code generation from models enabled by a significant research and development effort around model transformations. More recently, models have been viewed as being intrinsically executable through the use of model interpreters.

As argued in Frank (2022), these approaches lead to a disconnect between the model and the executable system. In the case of code generation through model transformation, the code becomes difficult to maintain and it is virtually impossible to link the code back to the models. The use of models-at-runtime is more attractive, however it requires a new model interpreter on a case-by-case basis.

These drawbacks can be addressed through the integration of MLM and code, *i. e.,* by making FMMLx an executable language. The details of this are described in Section 8.6. The essence of the approach is to provide classes at all levels with operations which are themselves modelled and open to execution through reflective services. This integration allows models to be directly executable without further extension, but domain-specific model interpreters can be defined as part of the model if required. The FMMLx environment therefore *becomes the system* together with a definition of its semantics. Any FMMLx application can reason about itself, present different views of its own definition, and even change the model if required.

Low-code systems present the user with an environment to create applications with minimal source code. MLM is a rich basis for designing low-code languages and applications (Frank et al. 2022). FMMLx is shown to go beyond standard low-code approaches, because it allows domain-specific modelling languages to be created that support low-code applications.

8.4 Methodology

There is a general lack of MLM methods since most research has addressed technical and tooling aspects. It is therefore important to investigate design principles that should be taken into account when undertaking MLM that achieves *reuse*, *integrity* and *maintainability* (Frank 2021) which proposes key design principles such as:

- Commonalities should be captured by an appropriate abstraction.
- Commonalities should be captured by an appropriate abstraction only, if the abstraction is likely to be invariant during the lifetime of a system.
- The higher the level of a class, the more invariant it should be.

- The design of a class at any level should aim at modification consistency.
- Assign properties of classes on levels higher than 1 to categories that indicate semantic differences.
- Avoid the introduction of 'fake' levels, *i. e.,* levels that could be expressed through generalization/specialization.

A design method for the MLM process is proposed in Frank (2021) and shown in Figure 8.4.

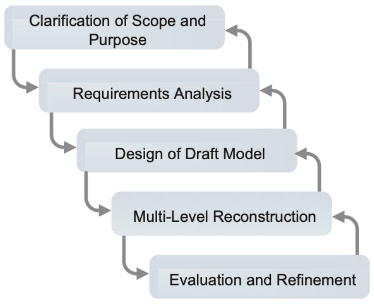

Clarification of Scope and Purpose
Requirements Analysis
Design of Draft Model
Multi-Level Reconstruction
Evaluation and Refinement

Figure 8.4: A Multi-Level Model Design Process (Frank 2021)

Briefly, the process phases involve: clarification of the purpose of the modelling activity; a definition of the requirements of the activity, with particular attention on the level of abstraction and domain-specific language development that is appropriate; a draft model that bridges the gap between requirements and a multi-level representation; the construction of a multi-level model including the definition of concrete syntax. Finally, the model is reviewed and design alternatives revisited, allowing earlier phases to be repeated.

Multi-level modelling can be applied in the context of digital business transformation where the need for methods is particularly acute (Frank 2019). This leads to a requirement for method engineering which is traditionally a meta-modelling activity. As noted earlier, there is a significant drawback using standard modelling languages in the context of activities that require highly expressive meta-concepts such as method engineering. Multi-level models are ideally suited to creating a dedicated methods language that can be integrated with the instances that represent particular processes as shown in Figure 8.5.

8.5 Pedagogy

Software and systems modelling is established as part of Computer Science and Information Systems undergraduate courses. More recently, MLM has been included in postgraduate degrees (Frank and Clark 2022b). FMMLx implemented in XModeler seems an ideal way to introduce students to the concepts of meta-modelling, abstraction, language engineering and the benefits of MLM. In particular, MLM allows students to instantiate classes and execute operations as part of an integrated environment. Constraints can be attached to language definitions that guide students in the construction of both models and correctly formed instances.

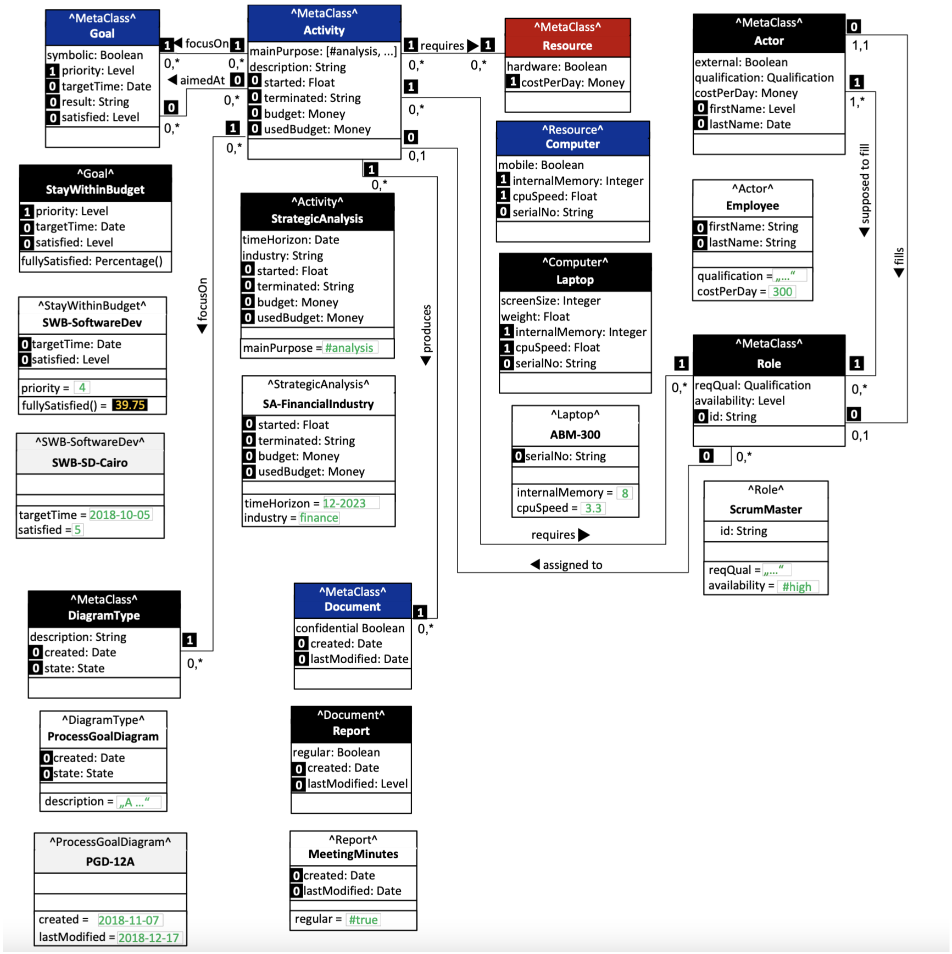
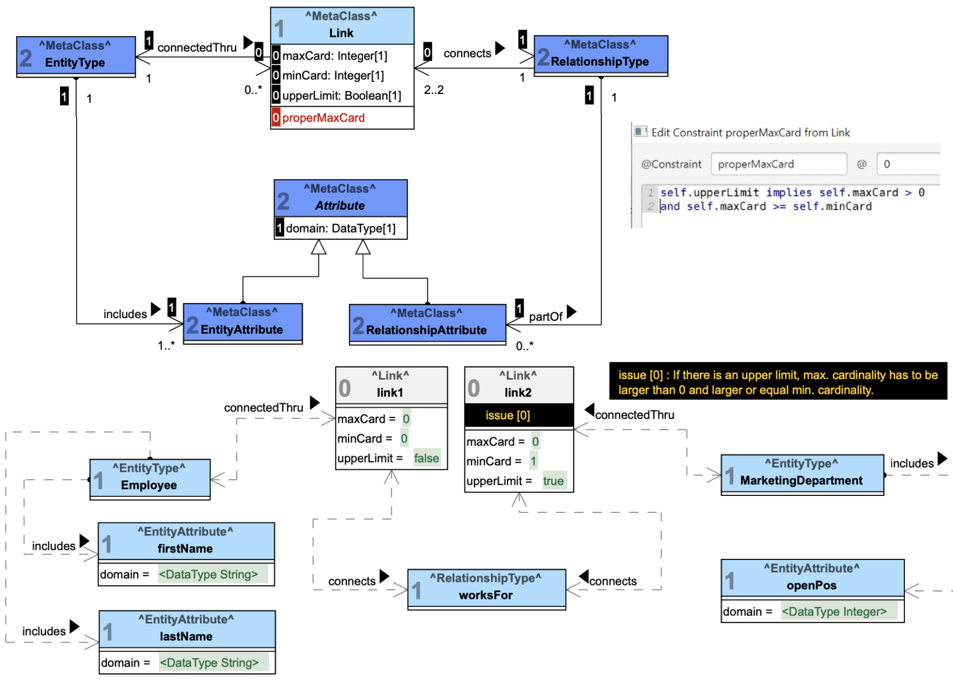


Figure 8.5: A Multi-Level Design Method (Frank 2021)

An example of this approach constructs a language for Entity-Relationship (ER) models together with the associated well-formedness constraints that guides student in the construction of ER-models and instances of the models – all on the same diagram panel as shown in Figure 8.6 (Frank and Clark 2022b).

The FMMLx and XModeler provide an ideal basis for such a pedagogical approach. However, a problem arises due to the abstraction and levels of reuse supported by the approach: raising definitions to the highest level of abstraction means that they can be reused across multiple languages and models, however this also means that these features become exposed at lower levels, particularly to novice users.

Several solutions are proposed (Frank and Clark 2022b): elimination by *hiding* where definitions at all levels are tagged so that tooling can elide all definitions with illegal tags; elimination by *constraints* where languages are associated with semantic definitions defining their legal features; elimination by *freezing* which acts as a special case of constraints just involving the names of the features.



```
self.upperLimit implies self.maxCard > 0
and self.maxCard >= self.minCard
```

issue [0] : If there is an upper limit, max. cardinality has to be larger than 0 and larger or equal min. cardinality.

Figure 8.6: An Entity-Relationship Language (Frank and Clark 2022b)

The MLM field is starting to consolidate to the extent that there have been several tutorials held at international conferences, e. g., Clark and Frank (2020). Further tutorial information about MLM, FMMLx and XModeler can be found at `https://www.wi-inf.uni-duisburg-essen.de/LE4MM/`.

8.6 XModeler and Tooling

XModeler (Clark et al. 2015; Clark and Willans 2014) is an environment for creating modelling languages and associated tooling. It was designed with the aim of having a single core executable reflective language called XCore that can be extended to create domain-specific languages. All features of XCore are represented in itself, meaning that it provides an ideal basis to implement FMMLx (Frank 2018).

The key features of XCore and the FMMLx extension are shown in Figure 8.7. Everything in XCore is a sub-class of Object including Class which is shown explicitly in the figure. Each object has the ability to access and update its slots. When a class is sent a `new` message it will create a new instance of itself by creating slots for all its attributes. Sending a message to an object inspects the class of the object (and the parents of the class) for an operation with the supplied name. When the operation is found, the default execution engine is supplied with the model of the operation (actually compiled to a sequence of machine instructions) and an appropriate context including the message arguments and the target of the message.

Figure 8.7 shows the additional properties required by FMMLx in green (as 'extended features'). XCore does not implement levels on classes and attributes, and so these are added

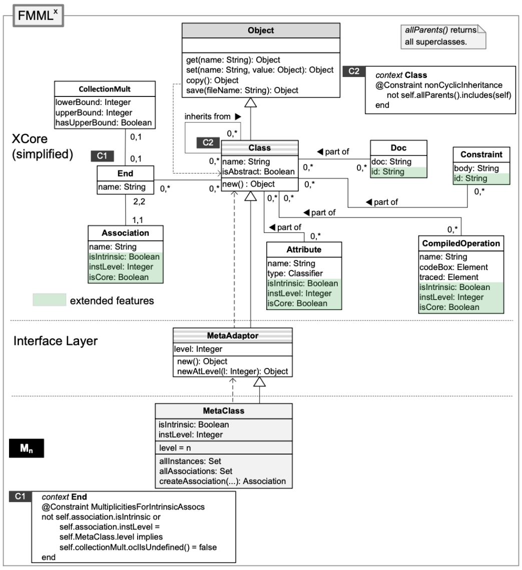


Figure 8.7: XCore and the FMML^X Extension

to XCore as a modification. As noted above, when an XCore class receives a new message, it will turn all available attributes into slots. However, this is not true for FMMLx which requires the level number of the attributes to be taken into account: attributes only turn into slots when the level number matches. Therefore, a new class is defined as a specialisation of Class called MetaAdaptor. All FMMLx classes are instances of MetaAdaptor and therefore use the redefined version of new. In order to encapsulate and hide these features, FMMLx models all contain the class MetaClass which extends MetaAdaptor with level numbering to classes. The resulting extension of XCore retains all of the meta-circularity and execution facilities of basic XCore, but adds level numbering to attributes and classes whose semantics are defined in terms of the redefined new operation leading to executable models as shown in Figure 8.8.

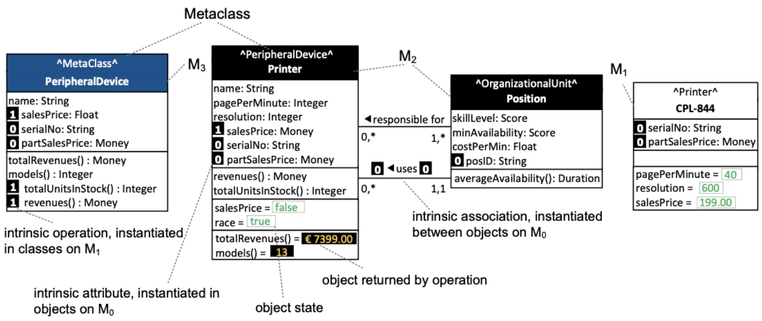


Figure 8.8: Example Use of FMMLx as an Extension of XCore

The reflective semantics of XCore, its meta-object protocol (MOP), and ability to create languages including multi-level languages is described in Clark 2020, where operations over models are used to express semantics in the form of meta-circular constraints.

8.7 Delegation

XModeler provides a meta-object protocol (MOP) where the system defined execution engine is parameterized. The XCore MOP parameters are the operations that perform slot access and update, and that send messages. A MOP must be defined at the meta-level, which means that executable MLM is an ideal framework to support a MOP. Delegation is an association between two classes, where the instances of one class serve as delegators, the instances of the other class as delegatee objects. A delegator object 'inherits' behaviour and state from the corresponding delegatee object (Frank 2018).

Figure 8.9 shows an example of delegation such that whenever a Student instance receives a message, if it cannot handle the message then the message is passed to the Person instance. This allows an instance of a class to have a temporal role which *inherits* the state and behaviour without having to permanently inherit these features.

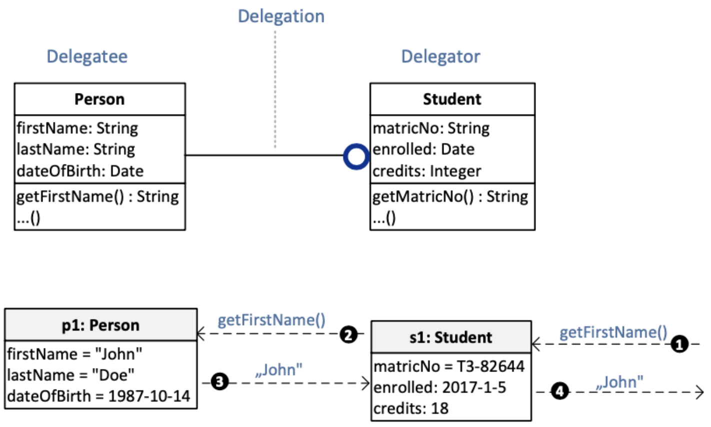


Figure 8.9: Delegation

FMMLx provides a MLM language with delegation via the XModeler XCore MOP. It does so by extending XCore with the appropriate features for linking all objects via delegation and by redefining the send operation that handles message passing.

8.8 Multi-Level Constraints

Model-based engineering involves hierarchy in two dimensions: inheritance and types, as shown in Figure 8.10 (Clark and Frank 2018). Traditional approaches tend to promote the former and ignore the latter. For example, when defining the semantics of a model, OCL can be used to attach model-specific constraints to classes at level 1; however, there is limited scope for traversal of the classification hierarchy within such constraints.

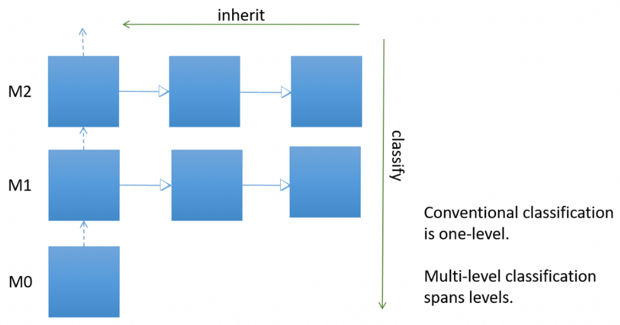

inherit
M2
M1
M0
classify
Conventional classification is one-level.
Multi-level classification spans levels.

Figure 8.10: MLM introduces two Dimensions to Classification

MLM involves *both* the inheritance and classification hierarchy. FMMLx introduces a new type relationship between classes and meta-class that merges the two dimensions with the aim of supporting the definition of domain specific modelling languages and the associated reuse of concepts.

Traditional OCL does not support the new features of FMMLx since it was not designed to take account of level numbers and the need to range across multiple classification levels. In XModeler, standard OCL classification can be defined as follows:

```
context Obj
  @Operation checkConstraints():Boolean
    self.of().classify(self)
  end
context Class
  @Operation classify(o:Obj):Boolean
    classifier.allConstraints()→forAll(c | c(o))
  end
context Obj
  @Constraint AllConstraintsSatisfied
    self.checkConstraints()
  end
```

The language-engineering facilities of XCore can be used to extend the standard OCL classification mechanism in order to take account of levels as defined by FMMLx. To do so, we introduce a new meta-class called `ClassWMC` which is the type of any new class (or meta-class) which supports levels. Any instance of `ClassWMC` is a class that can have `multiConstraints` which are predicates supplied with an implicit level number and a candidate object that is checked against the predicate only if the level number matches. The machinery for multi-level classification as required by FMMLx is defined as follows:

```
  @Package MultiConstraints extends XCore
    @Class ClassWMC extends Class
      @Attribute multiContraints : [Operation] (+,-) end
      @Operation classify(o:Obj):Boolean
        super(o) and self.metaClassify(o,0)
      end
      @Operation metaClassify(o:Obj,level:Integer):Boolean
        let C:[Operation] = self.allMultiConstraints()
        in C→forAll(c | c(o,level)) and self.guardedMetaClassifyUp(o,level)
      end
      @Operation allMultiConstraints():[Operation]
        parents→iterate(p C = multiConstraints |
          C + if p.isKindOf(ClassWMC) then p.allMultiConstraints() else [] end)
      end
      @Operation guardedMetaClassifyUp(candidate:Obj,level:Integer):Boolean
        if self.of().isKindOf(ClassWMC)
        then self.of().metaClassify(candidate,level+1)
        else true
      end
    end
  end
end
```

FMMLx opens up the two-dimensional space for classification. The multi-level constraint mechanism described above (and defined in more detail in Clark and Frank 2018) is just one option. This space offers up an interesting research area where languages can be defined that define the scope of constraints based on level ranges and constraints that can range over all the instances of classes at different levels.

8.9 Relationship to Product Lines

An important feature of systems engineering is the ability to describe product lines where a single model describes variation points and cross variation constraints. Product Line modelling languages have evolved to be able to describe such variations through configuration (Pohl et al. 2005) where features are defined and combined using logical operators such as *and* and *or*. It can be argued that while feature modeling may serve as a versatile tool for representing variability in requirements, they are limited in specifying and designing software product lines in terms of *ambiguity* and their support for *abstraction* (Frank et al. 2017).

MLM can address these shortcomings (Frank et al. 2017) because classification can be used as a powerful concept to express variability. A class defines the properties that are shared by all its instances and represents a set of constraints that need to be satisfied by its instances. MLM abstraction can be used to capture high-level variations whilst specialisation allows constraints to be expressed at a local level that is not otherwise possible in feature models.

A feature model expressed using FMMLx is shown in Figure 8.11 where the upper levels are considered as models of products and corresponding software systems at different levels of abstraction. Each level reduces the number of valid configurations of its upper levels by introducing constraints or assigning values to attributes. For example, a racing bike can be restricted as not being suited for tough terrains but rather for races *etc.* Further details of the use of FMMLx for feature models is given in (Frank et al. 2017).

8.10 Domain Specific Modelling Languages

MLM provides a basis for Domain Specific Modelling Languages (DSML) (Frank 2022). This is because the definition of languages is intrinsic to the MLM approach through type-level abstraction. This is further enhanced by the conflation of models and code as described in Section 8.3.

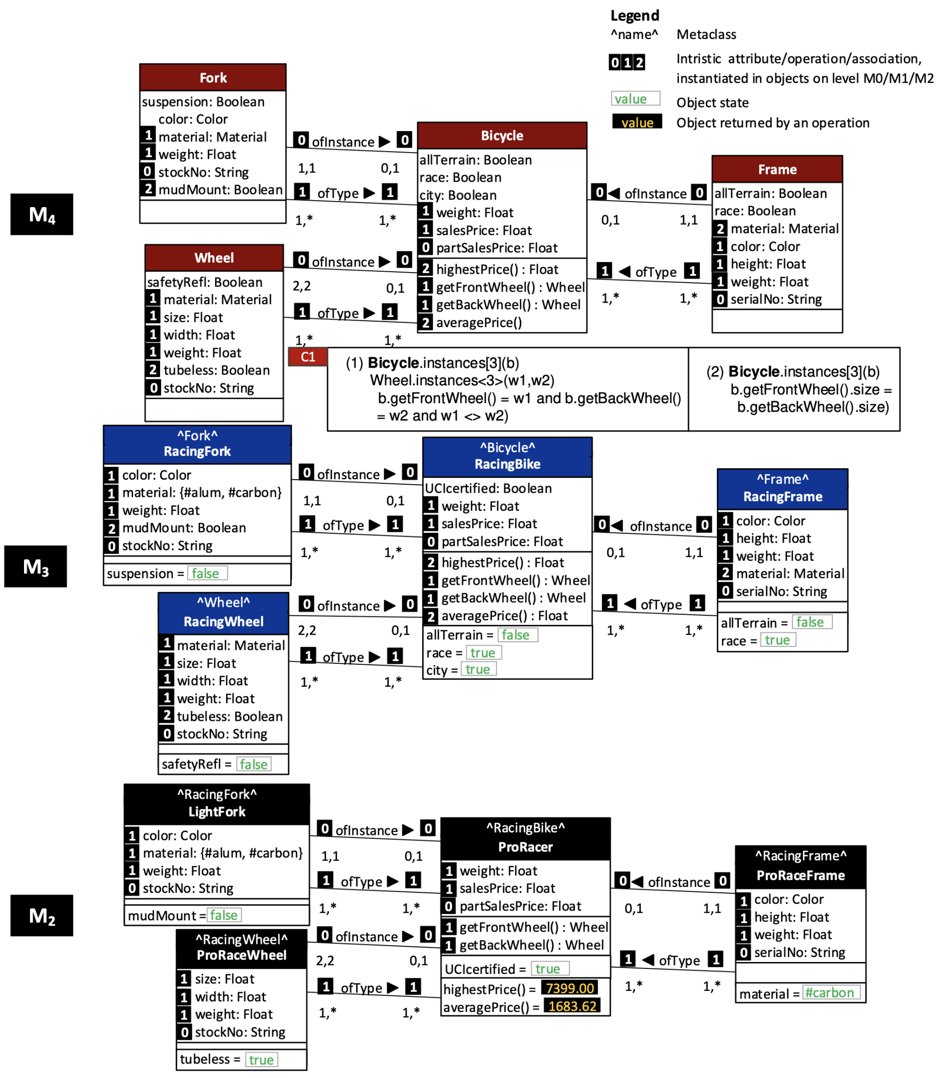


Figure 8.11: A Feature Model Expressed as a MLM (Frank et al. 2017)

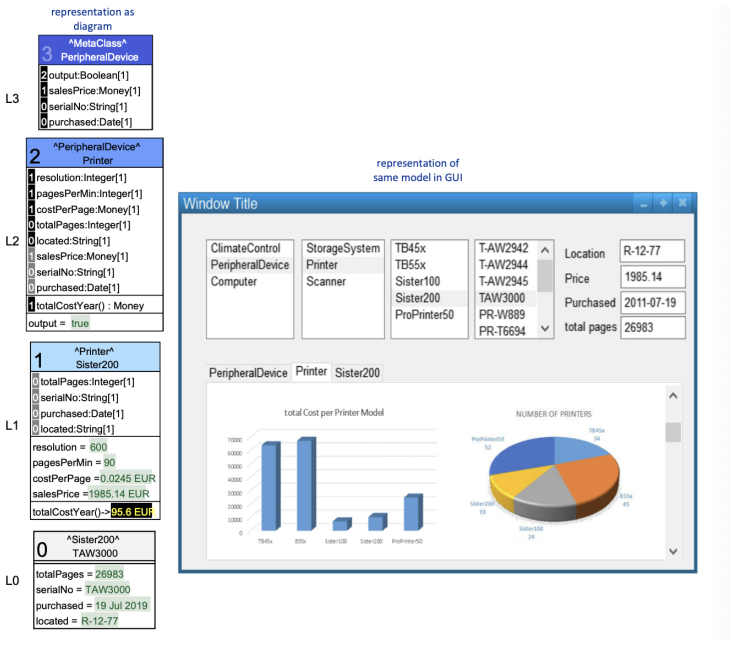


Figure 8.12: Domain Specific Representation (Frank 2022)

Figure 8.12 (taken from Frank 2022) shows an FMMLx model using two different representations. The first presents the model using a variation of standard class diagrams. The second presents the same model using a domain-specific format. It should be noted that such a representation arising from a UML class model (perhaps through the use of stereotypes) would have no dynamic features. In the case of FMMLx this is not a static representation precisely, because FMMLx conflates models and code and because of FMMLx class properties. The window on the right of Figure s8.12 represents the current state of the system which changes due to system execution and due to user interactions. Further, the underlying language of the model can be presented to, and changed by, the user.

8.11 Research Challenges

The field of executable MLM has established a number of contributions towards the goal of improving abstraction and reuse for information systems engineering. The recent Dagstuhl Seminar (Almeida et al. 2018) und the MULTI workshop series demonstrate the vitality of the field and its relationship to other fields such as Software Engineering, Ontologies and Process Engineering. The MULTI workshop series was established in 2014 as a forum for researchers and practitioners to discuss the field.

The series of workshops (see `https://jku-win-dke.github.io/MULTI2022/` for links to past events) is an excellent resource to review alternative approaches to MLM and to see recent advances. In 2017, the workshop was co-organised by Ulrich Frank Clark et al. 2017 and introduced a *challenge* that encourages participants to use MLM methods, technologies and tools to demonstrate the benefits when applied to a real-world problem. The following is a non-exhaustive list of key outstanding research challenges:

Unification: As noted in Section 8.2, there are a number of competing approaches which broadly agree that meta-concepts and levels are required, but differ in terms of the technical details. A key challenge remains to unify these as described in Jeusfeld and Frank (2021).

Refactoring: Multi-level models introduce pan-model constraints that must be maintained when the model is updated. Furthermore, MLM supports a wider range of *update* operations compared to standard modelling.

Type-checking: Executable MLM includes code along with types. The code includes references to types. Conventional modelling languages that can include expressions or code as part of the models can check the type references as through they were *static* definitions. However, a multi-level model can effectively represent constraints that define that additional types must exist *in the future* which raises an interesting challenge for any type checker.

Syntax: The activity of multi-level modelling is effectively *language engineering* in many cases which introduces the question of syntax. In addition, since the languages exist over multiple classification levels, the issue of syntax *abstraction* and *reuse* becomes relevant.

User views: A key opportunity for executable MLM is the ability to *run the model* because the distinction between te modelling environment and the end system becomes blurred. Further, since a MLM contains its own language and domain concepts, it becomes possible to service different user-roles by offering a different domain specific interface to the users whilst at the same time allowing users to modify the system with respect to business-facing concepts rather than technical features.

Processes: Whilst FMMLx employs executable modelling techniques based on XCore, the operations that are embedded within the models are at a low-level of abstraction. Conventional business process modelling languages have been developed to abstract from the implementation details of business operations. Something similar is required for multi-level models. This raises the challenge of how to abstract with respect to the type of a business process. The ideas described in Frank and Clark (2022a) provide some direction for this challenge.

Contingency: Most approaches to MLM make the levels explicit by labelling types and features with integers. This provides a clear indication as to how the concepts are intended to be used and prevents unintended consequences. However, this is not always appropriate since some concepts can be applied at several or across all levels. For example, the basic meta-type Class can be used at any level (including 0) – it is *contingent*. This raises a technology issue: how to retain the engineering benefits associated with both strict, non-strict and semi-strict level numbering whilst achieving the benefits of contingent types.

Constraints: As noted in Section 8.8, MLM introduces challenges for constraints with respect to the additional requirement for expressive power over type levels.

8.12 Closing comments

Executable Multi-Level Modelling has the potential to advance the field of Information Systems Engineering, especially in the era of *big data* where being able to make sense of large amounts of data requires an increasing amount of structure and abstraction. The ability to express information at the highest possible level of abstraction improves the potential for understanding existing systems and reusing information in future systems. The integration of type-level abstractions and type-based execution supports a language-driven approach to systems development that improves system resilience and accessibility for a wide range of stakeholders. The contributions made by Ulrich Frank to this field form the basis of many future innovations that will have impact across the IT industry and beyond.

References

Almeida, J. P. A., Frank, U. and Kühne, T. (2018). 'Multi-level modelling (Dagstuhl Seminar 17492)'. In: *Dagstuhl Reports*. Vol. 7. 12. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Atkinson, C. and Kühne, T. (2001). 'The essence of multilevel metamodeling'. In: *International Conference on the Unified Modeling Language*. Springer, pp. 19–33.

Clark, T. (2020). 'A Meta-Circular Basis for Model-Based Language Engineering'. In: *Journal of Object Technology* 19.3, pp. 1–18.

Clark, T. and Frank, U. (2018). 'Multi-Level Constraints'. In: *MoDELS (Workshops)*. Vol. 2245. CEUR Workshop Proceedings. CEUR-WS.org, pp. 103–117.

Clark, T. and Frank, U. (2020). 'Multi-level Modelling with the FMMLx and the XModelerML'. In: *Modellierung 2020*. Ed. by D. Bork, D. Karagiannis and H. C. Mayr. Bonn: Gesellschaft für Informatik e.V., pp. 191–192.

Clark, T., Frank, U. and Wimmer, M. (2017). 'Preface to the 4th International Workshop on Multi-Level Modelling (MULTI 2017)'. In: *CEUR Workshop Proceedings*. Vol. 2019, pp. 210–212.

Clark, T., Sammut, P. and Willans, J. S. (2015). 'Applied Metamodelling: A Foundation for Language Driven Development (Third Edition)'. In: *CoRR* abs/1505.00149. arXiv: 1505.00149. URL: http://arxiv.org/abs/1505.00149.

Clark, T. and Willans, J. (2014). 'Software language engineering with XMF and XModeler'. In: *Computational Linguistics: Concepts, Methodologies, Tools, and Applications*. IGI Global, pp. 866–896.

Frank, U. (2014). 'Multilevel modeling'. In: *Business & Information Systems Engineering* 6.6, pp. 319–337.

Frank, U. (Dec. 2018). *The Flexible Modelling and Execution Language (FMMLx), Version 2.0: Analysis of Requirements and Technical Terminology. ICB Research Report*. Tech. rep. 66. Essen. DOI: doi:10.17185/duepublico/47506. URL: https://duepublico.uni-duisburg-essen.de/servlets/DerivateServlet/Derivate-46850/ICB%5C_Report%5C_66.pdf.

Frank, U. (2019). 'Specification and Management of Methods-A Case for Multi-Level Modelling'. In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, pp. 311–325.

Frank, U. (2021). 'Prolegomena of a Multi-Level Modeling Method Illustrated with the FMML x'. In: *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, pp. 521–530.

Frank, U. (2022). 'Multi-level modeling: cornerstones of a rationale'. In: *Software and Systems Modeling* 21.2, pp. 451–480.

Frank, U. and Clark, T. (Sept. 2022a). 'Multi-Level Design of Process-Oriented Enterprise Information Systems'. In: *Enterprise Modelling and Information Systems Architectures (EMISAJ)* (17), pp. 1–50. doi: doi:10.18417/EMISA.17.10.

Frank, U. and Clark, T. (2022b). 'Peculiarities of language engineering in multi-level environments or: Design by elimination: a contribution to the further development of multi-level modeling methods'. In: *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, pp. 424–433.

Frank, U., Mattei, L. L., Clark, T. and Töpel, D. (2022). 'Beyond Low Code Platforms: The XModelerML — an Integrated Multi-Level Modeling and Execution Environment'. In: *Modellierung 2022 Satellite Events*. Ed. by J. Michael, Pfeiffer, Jérôme and A. Wortmann. Bonn: Gesellschaft für Informatik e.V., pp. 235–244. doi: 10.18420/modellierung2022ws-032.

Frank, U., Reinhartz-Berger, I., Sturm, A. and Clark, T. (2017). 'A multi-level approach for supporting configurations: A new perspective on software product line engineering'. In: *CEUR Workshop Proceedings*. Vol. 1979. CEUR-WS, pp. 170–178.

Jeusfeld, M. A. and Frank, U. (2021). 'Unifying multi-level modeling: A position paper'. In: *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, pp. 536–540.

Pohl, K., Böckle, G. and Van Der Linden, F. (2005). *Software product line engineering*. Springer.

Chapter 9

Projective Multi-Level Enterprise Architecture Modeling with MEMO

Colin Atkinson, Christian Tunjic and Arne Lange

At the heart of the current drive for ubiquitous digitisation and pervasive cyber-physical systems are enterprise architectures that comprehensively describe the interplay between organisations' goals, plans, and IT solutions. However, traditional enterprise architecture modelling approaches are not fit for purpose since they are unable to cope with the scales and dynamics of modern enterprises and systems. This paper explores how the full and synergistic integration of three forward-looking modelling principles can help address this problem and provide a sound foundation for the enterprise architecture models needed in the future.

9.1 Introduction

To cope with the increasing size, complexity, and scope of Enterprise Architectures (EAs), there is a growing consensus that Enterprise Architecture Modeling (EAM) environments need to be built on three core principles. The first is that they should be fundamentally *view-centric* so that their many stakeholders can be provided with precisely the information they need at the right time and in the right form (Frank 2002). For this reason, EAM approaches, from Zachman (1987) onward, have been structured around a viewpoint framework that describes what view types are available, from what perspectives (i. e., viewpoints), and how they are organised. The second is the principle that EAM approaches need to be founded on *multi-level modelling* approaches rather than on the traditional two-level modelling approaches supported by the UML and most other EAM languages (Frank 2014b). Two-level approaches struggle to adequately support the deep characterisation patterns frequently occurring in EAs or the flexible definition of new languages for new view types. Moreover, when the scope of EAMs includes the 'run-time' phase of an enterprise's operation, multi-level modelling approaches allow all required information (or data), at all levels of abstraction to be represented in a seamless and unified way. The third is the principle that EAM environments need to be fundamentally *'projective'* (ISO/IEC/IEEE 2011), in the sense that all views seen by stakeholders should be explicitly projected on demand from a Single Underlying Model (SUM) (Atkinson et al. 2010) using model transformation defined by end users. The alternative 'synthetic' approach, in which views actually hold primary information and need to be kept mutually consistent by a multitude of pairwise consistency-preservation rules, simply does not scale up to large EAMs.

Clark and Frank (2020) have written extensively about the synergy between the first two principles in EAM modelling and have led the way in developing suitable modelling platforms to demonstrate its benefits. Most recently, they have shown how their XModelerML multi-level modelling tool and FMML^X multi-level modelling language can provide a natural and efficient platform for using Frank's MEMO EAM language (Frank 2002). As its name implies, MEMO (Multi-Perspective Enterprise Modelling) is a fundamentally view-centric approach that offers different stakeholder types individual views of an EA customized to their needs and skills. However, while the XModellerML tool uses a single underlying store of data, the XModelerML/FMML^X environment does not fully support the projective view concept as characterised in the standard for architecture description in systems engineering (ISO/IEC/IEEE 2011), because this store is hidden and the supported viewpoints/view types are not readily customisable and extensible by end users.

In this paper, we, therefore, explore the extension of this synergy to cover all three principles by presenting a platform designed from the ground up to support them and showing how such a platform can be customized to support the MEMO view types. The platform, therefore, extends the view-centric and multi-level characteristics demonstrated by the XModelerML/FMML^X environment with full support for viewpoint framework customization and view type definition. The approach, called DOSM, and the tool, called DOREEN, were developed and first presented in the Ph.D. thesis of Christian Tunjic (Tunjić 2021). Much of the material in this paper is therefore reproduced from that manuscript.

The rest of this paper is structured as follows. The next section presents the background to the DOSM approach and the existing approaches/tools that have been integrated into the prototype DOREEN modelling environment: the OSM modelling metaphor and the deep modelling approach supported by the MELANEE tool. Section 9.3 then shows how this environment can be customized to support the viewpoint framework underpinning Frank's MEMO EAM language, while Section 9.4 shows how this customized environment can be used to model a simple EA example. Finally, Section 9.5 discusses the insights gained and concludes with some closing remarks.

9.2 Deep Orthographic System Modelling

The Deep Orthographic System Modelling (DOSM) approach and supporting tool, DOREEN (Deep, Orthographic System Engineering Environment), is built from the ground up to support and integrate the three aforementioned principles in a seamless and synergistic way. Each is discussed in a separate subsection below.

9.2.1 Multi-Dimensional Viewpoint Framework

Although the notion that views should play a key role in the depiction and development of EAs is generally accepted in most EAM approaches, there is little consensus on how view types should be defined and how they should be organised. Most view-based modelling approaches just delineate a fixed set of views that are essentially organised in a linear viewpoint framework, such as RM-ODP (ISO/IEC 1997) and MDA (Miller and Mukerji 2003). Others, such as Zachman (Zachman 1987), MEMO (Frank 2002) and ArchiMate (Iacob et al. 2012), organise views in a two-dimensional framework in which viewpoints are characterised by two basic properties (or facets), while a small number of EAM modelling approaches,

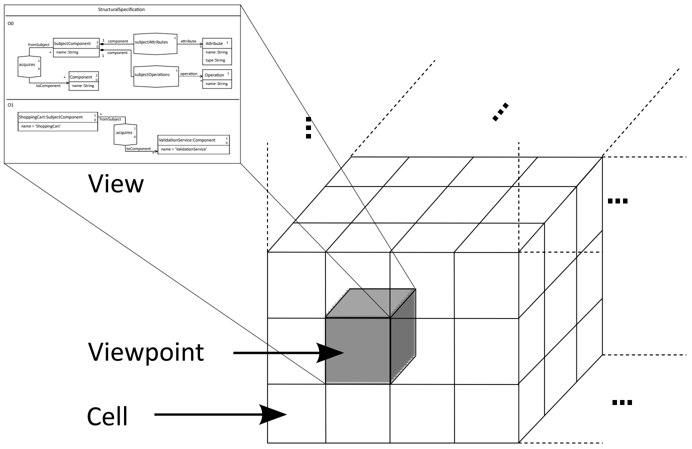


Figure 9.1: View Selection based on Dimensions

such as a FEAF (Urbaczewski and Mrdalj 2006; Schlieter et al. 2015), organise viewpoints within a three-dimensional framework.

The DOSM approach takes this principle further by allowing views to be organised in viewpoints frameworks with unlimited numbers of dimensions which are governed by the Orthographic System Modeling (OSM) approach inspired by the orthographic projection principles used in CAD tools. This multi-faceted way of categorising and organising view types provide the basis for an intuitive, multi-dimensional 'navigation' interface for selecting and creating views. This is analogous to the OLAP approach, Codd (1993) used in the field of data warehousing in which a multi-dimensional space is spanned by orthogonal dimensions, each containing multiple, unique dimension values. Views are situated within this space, which is conceptually a hyper-cube (i. e., a multi-dimensional cube), at the position whose coordinates exactly characterise the view. Figure 9.1 illustrates this multi-dimensional space as a three-dimensional cube whose cells identify concrete views.

The dimensions of the cube are derived from concerns often defined by the chosen method, where each independently-selectable choice for a concern represents a dimension value. Dimensions can be static, in which case the number of selectable values is fixed, or they can be dynamic, in which case the number of values is dynamic (i. e., derived from the enterprise model) and changes over time. In order to select a view, a user has to select one choice for each dimension. The goal is to have dimensions that are 'orthogonal' to each other in order to span the multi-dimension space with a minimum number of concerns.

9.2.2 Deep Modeling

DOSM supports the second principle by allowing all views to be represented as multi-level models of arbitrary depth. These could easily be represented in the FMML^{x} flavour of multi-level modelling developed by Frank Clark and Frank (2020), but DOSM adopts the LML multi-level modelling language supported by the Melanee tool (Gerbig 2017) since this is based on the so-called Orthogonal Classification Architecture (OCA) (Atkinson and Kühne 2001; Atkinson and Kühne 2002), which separates linguistic classification from ontological classification (Kühne 2006) into two orthogonal dimensions. As shown in Figure 9.2, an

OCA-based environment usually has three linguistic levels (L_2–L_0), where L_2 contains the linguistic (meta)model (i. e., the basic set of concepts which are used to build the deep model), L_1 contains the domain content (i. e., the deep model containing the user data) and L_0 contains the 'real world' objects that are described by the deep model.

Users usually only work with the L_1 linguistic level which contains the domain content. Figure 9.2 shows three ontological levels (O_0–O_2) in the L_1 linguistic level, but generally, the number of levels is unlimited and can be adapted according to the needs of the domain to be modelled. The O_0 ontological level is the most abstract and the O_2 level is the most concrete. Generally, the O_n ontological level contains the instances of the O_{n-1} ontological level. Furthermore, all user data is modelled in a unified way, using the basic set of model elements defined in L_2.

9.2.3 Projected Views from an Essential SUM

DOSM supports the third principle outlined above by fully adopting the notion of projective views as outlined in the standard for architecture description in systems engineering (ISO/IEC/IEEE 2011). Figure 9.3 depicts this idea schematically by showing how different views are 'projected' from a Single Underlying Model (SUM) by model transformations that can be written by end users. Since it is an extension of the OSM approach, DOSM assumes the SUM is completely free of internal redundancy (i. e., that it is an essential SUM) (Atkinson et al. 2015). In contrast, other projective SUM-based approaches like Vitruvius

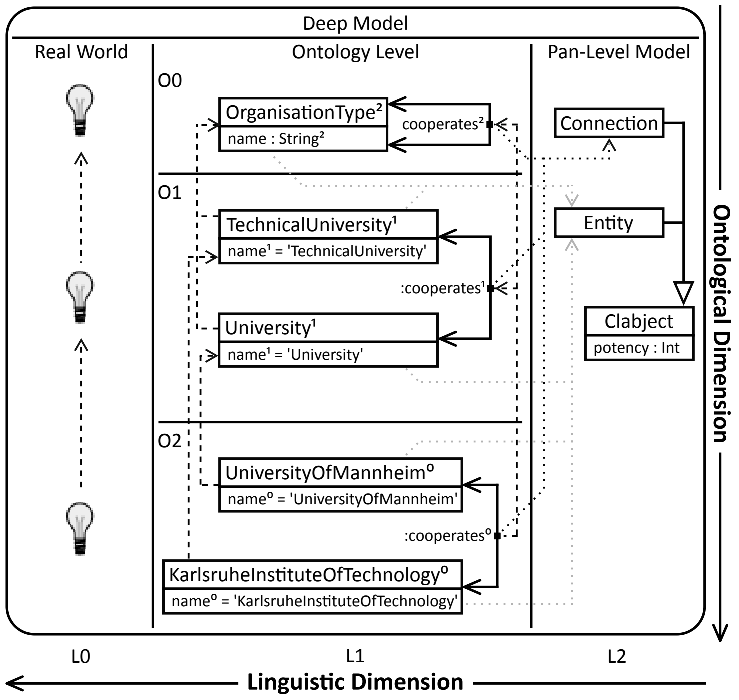


Figure 9.2: OCA Deep Model Example

(Kramer et al. 2013) allow the SUM to have multiple internal models that are kept consistent in a synthetic way using explicit consistency preservation rules.

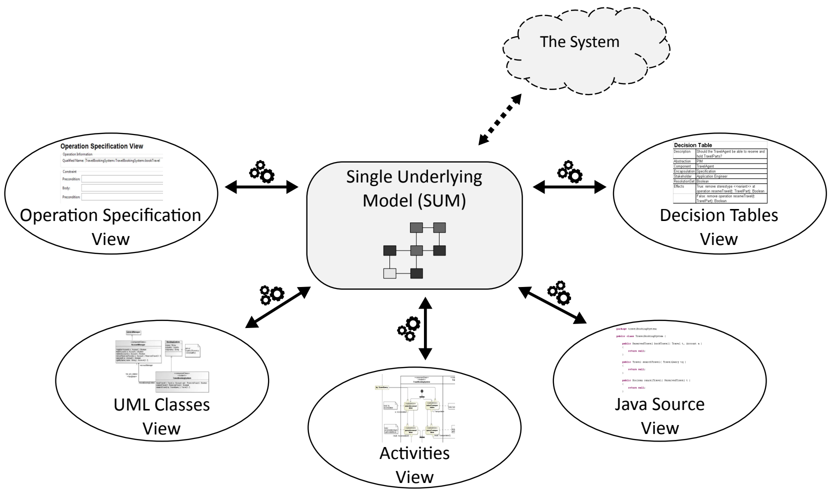


Figure 9.3: Views Project from an Essential SUM (Tunjic et al. 2018)

In essence, a view type is defined by two things – one or more predefined ontological levels which capture the ‘language’ that can be used to represent view content, and the VPRs which define the mappings between information in the SUM and in the view. Since these are both accessible to certain users (i. e., methodologists) and are represented using well-known modelling conventions (i. e., based on the UML and ATL), new view types can easily be added. View types can also easily be extended using the inherent specialisation mechanism of the deep modelling language.

9.3 MEMO Viewpoint Framework Configuration

The Multi-Perspective Enterprise Modelling (MEMO) Framework developed by Frank (2002) is a view-based method for EAM. It provides several modelling languages tailored for different stakeholder types such as the Strategy Modelling Language (MEMO SML), the Organization Modelling Language (MEMO OrgML), and the Object-oriented Modelling Language (MEMO OML). Its viewpoint framework defines three main *perspectives* on an enterprise – strategy, organization, and information system – and describes them in terms of four *aspects* – resource, structure, process, and goal. Figure 9.4 shows the usual 2D representation of the framework used by Frank (2014a) to show the two orthogonal sets of concerns.

9.3.1 MEMO Dimension Space

To configure a DOSM environment such as DOREEN to support MEMO’s viewpoint framework the ‘static’, predefined facets of views need to be mapped to dimensions in a viewpoint hypercube – namely, a Perspective dimension and an Aspects dimension.

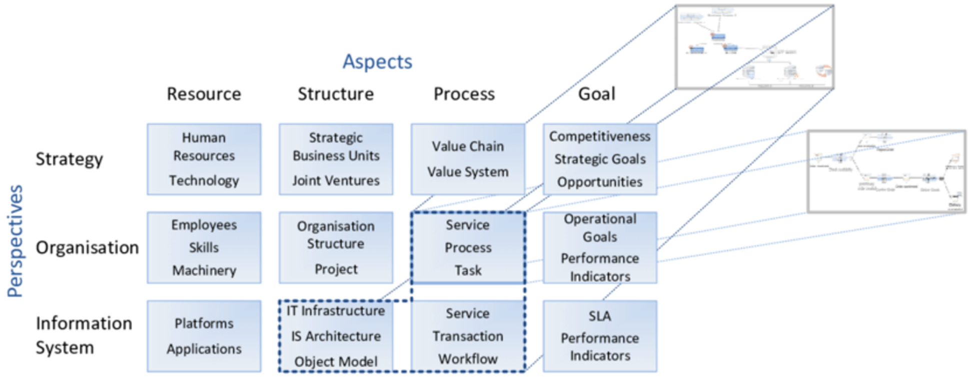

Aspects
Resource
Structure
Process
Goal
Perspectives
Strategy
Organisation
Information System
Human Resources Technology
Strategic Business Units Joint Ventures
Value Chain Value System
Competitiveness Strategic Goals Opportunities
Employees Skills Machinery
Organisation Structure Project
Service Process Task
Operational Goals Performance Indicators
Platforms Applications
IT Infrastructure IS Architecture Object Model
Service Transaction Workflow
SLA Performance Indicators

Figure 9.4: MEMO Viewpoint Framework (Frank 2014a)

Perspective. The Perspective dimension is a static dimension used to provide an abstraction mechanism over an enterprise. It ranges from *strategy*, which is the most abstract, over *organization* to the most concrete, *information system*.

Aspect. The Aspect dimension is a static dimension whose values (*resource*, *structure*, *process*, and *goal*) represent a view of an enterprise based on given concerns.

In addition to these two static dimensions, to allow views to portray human-tractable amounts of information focused on a recognisable ingredient of an EA model, rather than information about the whole system, as the basic viewpoint framework shown in Figure 9.4 implies, some dynamic dimensions are also needed. These represent fundamental ingredients of a MEMO EA model that can serve as the focus of a view. In this example, we add two new 'dynamic' dimensions to identify activities and business processes.

Activity. The Activity dimension is a dynamic dimension that provides values to identify the activities of the enterprise. It is dynamic since the activities to be represented depend on the particular enterprise being portrayed.

Business Process. The Business Process dimension is a dynamic dimension that provides values to describe the business processes of the enterprise. Again, it is dynamic since the business process depends on the particular enterprise model being portrayed. In addition to the dynamic dimension values corresponding to individual business processes, the dimension contains the static value *All*, which is used when a view shows all business processes of a particular activity.

Overall, therefore, a four-dimensional viewpoint framework is used to customize (i. e., instantiate) DOREEN to support MEMO. This has two static dimensions, *Aspect* and *Perspective*, and two dynamic dimensions, *Activity* and *BusinessProcess*, which are used to identify the Activity and BusinessProcess model elements contained in the SUM. The left hand side of Figure 9.5 shows the data structure (i. e., model) used by the DOREEN tool to represent this viewpoint hypercube. Note that each dimension is characterised as an instance of SD (static dimension) or DD (dynamic dimension) respectively and that the model shows all the static dimensions elements (SDEs) contained by the dimensions. Although dynamic dimensions can have some static dimension elements (i. e., values), by definition they must also have some dynamic elements which have to be derived from the SUM.

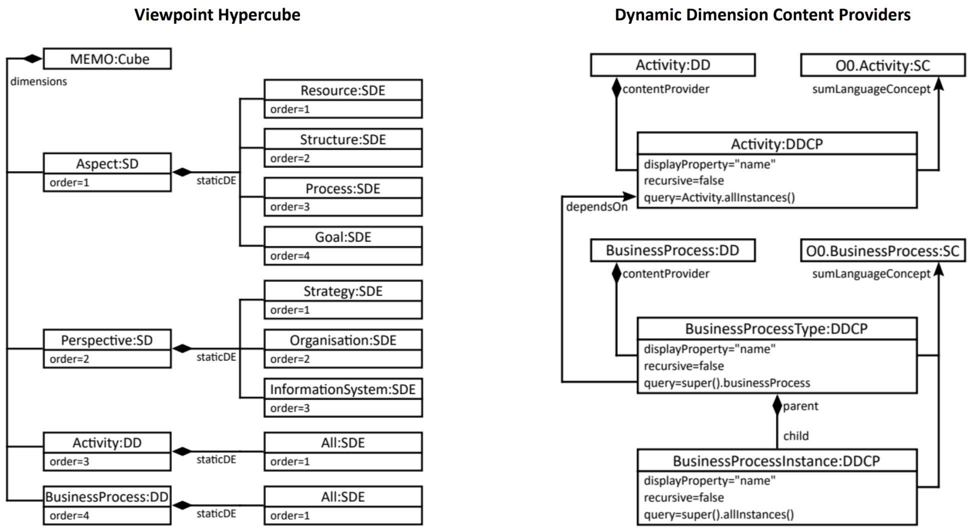


Figure 9.5: Hypercube and Dynamic Dimension Content Providers

The right-hand side of Figure 9.5 shows how this is achieved using so-called ‘dynamic dimension content providers’ (DDCP), which provide an API for the DOREEN tool to obtain current dimension elements from the SUM. These elements can change over time as the model of the EA stored in the SUM evolves.

Figure 9.5 shows how the Activity and BusinessProcess dimensions contain appropriate DDCPs to support queries for the current dimension elements when needed. The BusinessProcessType DDCP contains the BusinessProcessInstance DDCP in a parent /child relationship, to reflect the fact that business processes can be further described using sub-processes. The sub-processes of a business process can, for example, be displayed in the DynamicDimension *BusinessProcess* by adding a child DynamicDimensionContentProvider that provides the DynamicDimensionElements for the sub-processes.

9.3.2 SUM Language

In order to provide a single consistent description of the enterprise under discussion, all types of information (i. e., classes) representable in the available view types must be integrated into an ‘essential’ SUM language that can support the full specification of an enterprise. The SUM language for the MEMO DSL relevant to this example is shown in Figure 9.6. This is a simplified version of the full meta-model described in Frank (2014a). The language includes concepts to capture the activities in a value chain using the `Activity` model element. The realization of the activities at the organizational level is represented by the `implementedBy` model element, linking an activity to one or more `BusinessProcesses`. Business processes are further described using the model elements `Process` and `Event`. The transition between the `Processes` to the `Events` and vice versa is realized by the `linkPE` model element.

The model elements used to define the SUM language are enhanced with `potency` and `durability` properties which describe the possible influence range of the model elements. Since `BusinessProcesses` can be executed in a running enterprise, the `BusinessProcess`

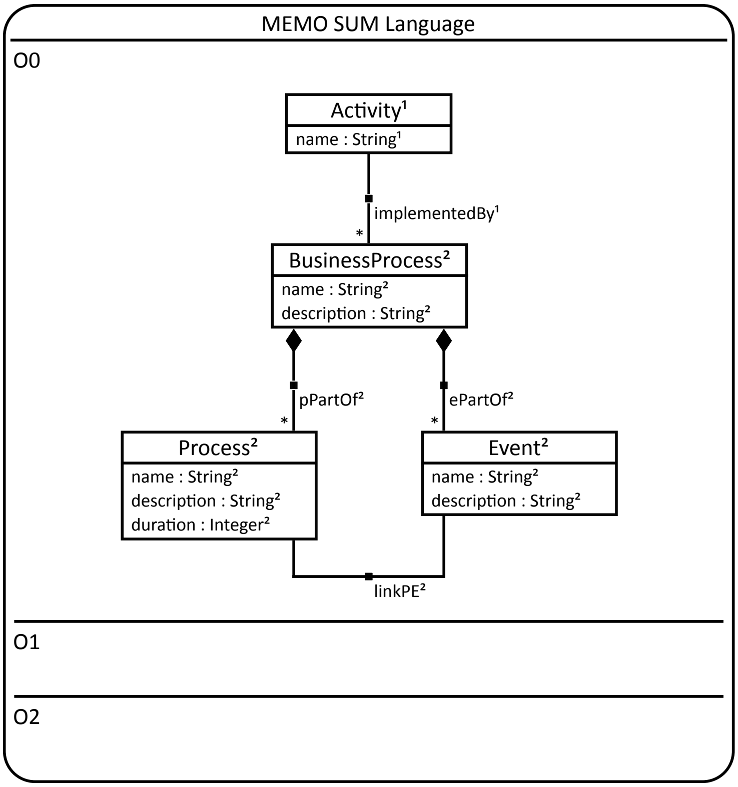


Figure 9.6: MEMO – SUM Language

model element has a potency value of '2' so that instances of BusinessProcess can exist at O_1 where they would have a potency value of '1'. These clabjects at O_1, therefore, play the role of types for the BusinessProcesses instances situated at O_2 which are executed in running enterprises and have a potency value of '0'. The same is true for the model elements which belong to a BusinessProcess (Process and Event) and the corresponding connections (pPartOf, ePartOf and linkPE). The attributes of the model elements belonging to BusinessProcesses are defined with the same influence range as their owning model elements. This makes it possible to use the attributes at the ontological levels O_1 and O_2. The duration attribute of the model element Process is used at O_2 to store the execution duration of a Process in the running enterprise.

9.3.3 View Type Definition

The final step in the process of configuring a DOSM-based modelling environment like DOREEN for MEMO is to define the supported view types. Based on the dimensions described previously, and the kind of information that can be represented using the MEMO OrgML modelling language, two basic view types are needed, activity view types and business process view types. In this section, for space reasons, we only outline the latter.

The Business Process View Type is designed to facilitate the description of business processes using concepts like `Process` and `Event` and the relationships between them. The business process view is portrayed by cells with the aspect *process* and the perspective *organization*. The business process view type elaborates on the structure of business processes by defining the actions and events used to realize them.

The left-hand side of Figure 9.7 shows the view language for the business process view type. The BusinessProcess model element in this view type serves as the *subject* of an instance of the view type since each instance portrays (i. e., is a view of) a specific business process. The *subject*-oriented approach for defining views avoids the creation of views that convey information about arbitrary parts of an enterprise and simplifies the structure of the overall enterprise specification.

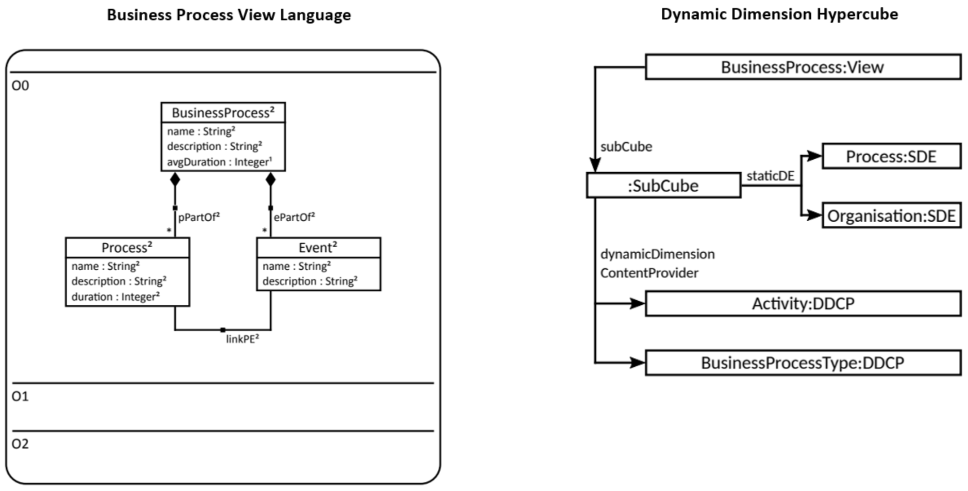


Figure 9.7: Business Process View Language and Dynamic Dimension Subcube

Once the language for a view type has been defined, the next step is to define the model transformation that creates instances of the view type from SUM on demand, and updates the SUM when changes are made to model elements in a view instance. In DOREEN, such transformations are defined using deep ATL (Tunjić 2021), since they need to cope with the fact that both the SUM and views are deep models. An excerpt of the transformation for the business process view types is shown below in Listing 9.1.

The right-hand side of Figure 9.7 shows the `SubCube` of the 'Business Process View Type' from Figure 9.7, based on the hyper-cube shown in Figure 9.5. The Subcube is defined by the *Process* and *Organisation* `StaticDimensionElements` (`SDE`) and the *Activity* and *BusinessProcessType* `DynamicDimensionContentProvider` (`DDCP`). Because of the dependency between the two dynamic dimensions, the content of the business process dimension depends on the selected *Activity* from the activity dimension.

Note that the view content part of the view shown on the right-hand side of Figure 9.7 is empty, since no concrete value for the dynamic dimension is available at this point.

```
rule BusinessProcess2BusinessProcess {
  from
   s : SUM!O0.BusinessProcess 1..2 (thisModule.isSubject(s))
  to
    t : VIEW!O0.BusinessProcess 1..2 (
      name <- s.name,
      _l_.name <- s._l_.name,
      description <- s.description,
      duration <- thisModule.sumDurationOfProcesses(s),
      avgDuration <- thisModule.avgDurationOfInstances(s)
  )
}
rule Process2Process {
from
    s : SUM!O0.Process 1..2 (thisModule.processBelongToSubject(s))
  to
    t : VIEW!O0.Process 1..2 (
      name <- s.name,
      _l_.name <- s._l_.name,
      description <- s.description,
      duration <- s.duration
  )
}
```

Listing 9.1: Extract of the Business Process View Projection Transformation

9.4 Example EA Model (Insurance Enterprise)

In this section, we show how a DOSM environment configured for MEMO in the way described in the previous section can be used to model a small example EA. More specifically, we show how the business process view type can be used to model behavioural aspects of the enterprise architecture of an insurance company example. This is achieved primarily using portrayals of business processes depicted using views corresponding to the *process* element of the *Aspect* dimension. This provides a strategic perceptive on a system in order to describe how the implementation of an enterprise's behavioural properties impacts the underlying information system in terms of descriptions of the transactions and workflows.

In MEMO, value chains are composed of activities at the strategic level which are further used in the description of business processes situated at the organizational level. The mapping of the activities to business processes shows how the activities are realized by the underlying business. Business process views show the concrete structure of individual business processes in a language inspired by the MEMO OrgML language. The left-hand side of Figure 9.8 shows an example of the use of the DOREEN dimension explorer to select a particular type of view of a specific business process. More specifically, the hyper-cube selections (i. e., coordinates) on the left-hand side identify an instance of the 'Business Process View Type', with the values *'Process – Organisation – ClaimProcessing – ClaimProcessingCars'*. *ClaimProcessingCars* is the name of a particular process modeled in the SUM and is the 'subject' of the view. The right-hand side of Figure 9.8 shows the contents of the view that is created by execution of the view projection transformation when the *'Process – Organisation – ClaimProcessing – ClaimProcessingCars'* view is selected. DOREEN 'knows' it has to execute the model transformation for the business process view type, with *ClaimProcessingCars* as

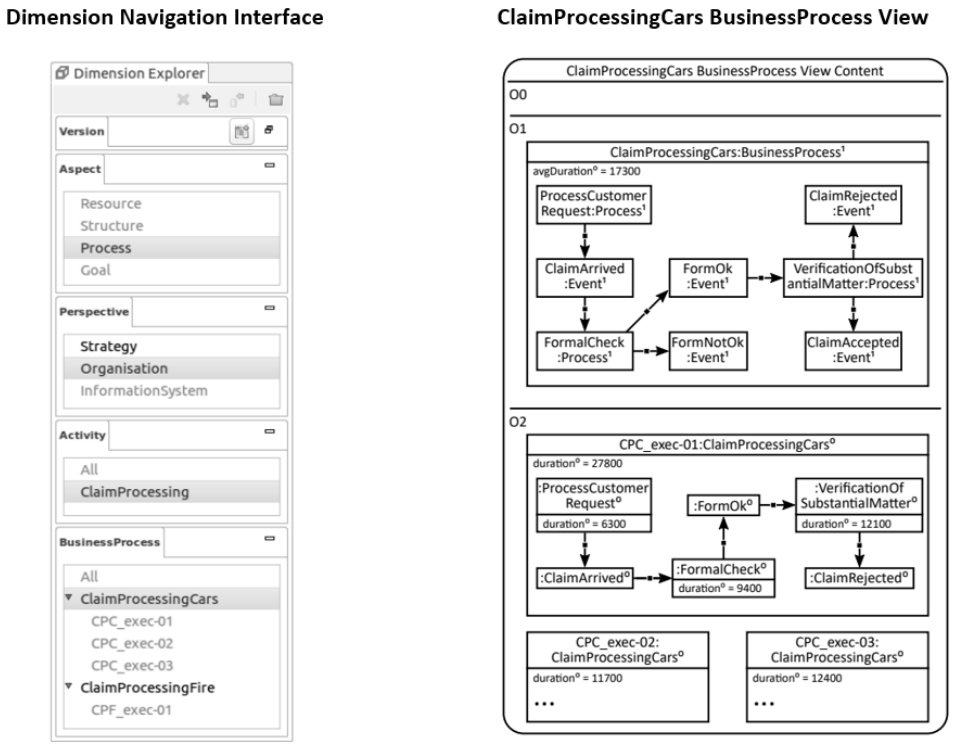


Figure 9.8: Dimension Navigatior and ClaimProcessingCars BusinessProcess View

the subject, when this cell is selected because it lies within the subcube visualised by this view type (as specified by the model on the right-hand side of Figure 9.7).

The static dimension values 'Process' and 'Organisation' define the static type of the view and the dynamic dimension values 'ClaimProcessing' and 'ClaimProcessingCars' define the dynamic content of the view. The view content contains the business process identified by the view's cell (i. e., the 'ClaimProcessingCars' business process). As defined by the source and target rule potencies in the view projection transformation in Listing 9.1, the instances of business processes from the O_1 and O_2 ontological levels are projected along with their inner events and processes to the same ontological levels into the view's content.

The view shown on the right-hand side of Figure 9.8 also shows two properties calculated during the view projection process. The first, the *duration* property of the 'CPC_exec-01' model element contains the SUM of all duration values of the owned Process instances in milliseconds. The second aggregated value in the view, the *avgDuration* of the 'ClaimProcessingCars' model element, is derived from all the duration property values of all instances of the 'ClaimProcessingCars' model element (at O_2). Due to the durability property value (of '1') of the *avgDuration* Attribute (see Figure 9.7), the property only exists at O_1 and not in the following levels. This value shows that the execution duration of the 'CPC_exec-01' business process instance is above the average execution duration of all business process instances.

9.5 Conclusion

This paper has argued the case for building future EAM approaches around the three fundamental principles of view-centricity, multi-level models and projective view generation. Essentially, from the point of view of users, this means that future EAM environments need to ensure that EAs are represented as multi-level models which can only be seen and edited via multi-level views projected on demand from a SUM via user-defined model transformations. To demonstrate the feasibility of this vision, the paper gave a brief overview of a prototype platform (DOREEN) that supports such an approach (DOSM) and showed how it could be used to support projected views of a simple EA represented using Frank's MEMO method. Although the multi-level modeling language used in DOREEN is the LML language supported by the Melanee tool, any multi-level modelling language optimized from domain modeling, such as Frank's FMML^x language, could be used just as well.

The importance of these three principles in EAM is likely to grow in the future, as the boundaries between EAs and cyber-physical systems (CPSs) continue to blur. More specifically, as companies' infrastructures, products and networks are increasingly bound up in CPSs, and all their physical artifacts, employees and customers gain digital twins, enterprise architectures will essentially be indistinguishable from CPSs. This means that they will have to (a) span all phases of an enterprise's life cycle including system construction, deployment, operation and evolution (i. e., support information across multiple levels of abstraction), (b) ensure that the many different kinds of stakeholders can access all the information they need (but no more) at the right time and in the right form (i. e., support view-only access to the underlying enterprise model) and (c) preserve consistency in a projective rather synthetic way to avoid an exponential explosion in the number of pairwise consistency relationships that have to be maintained (i. e., allow views, of user-defined views types, to be projected from a SUM by user-defined model transformations). By demonstrating the feasibility of such an approach, and showing how it can be realised using EA modeling languages (e. g., the MEMO DSLs) and existing multi-level modeling language/tools (e. g., Melanee/LML or XModelerML/FMML^x), we hope this paper will motivate future work on the synergy between these principles and open the way for future commercial products based on such a platform.

References

Atkinson, C. and Kühne, T. (2001). 'The Essence of Multilevel Metamodeling'. In: *Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools.* London: Springer, pp. 19–33.

Atkinson, C. and Kühne, T. (Oct. 2002). 'Rearchitecting the UML infrastructure'. In: *ACM Transactions on Modeling and Computer Simulation* 12.4, pp. 290–321.

Atkinson, C., Stoll, D. and Bostan, P. (2010). 'Orthographic Software Modeling: A Practical Approach to View-Based Development'. In: *Evaluation of Novel Approaches to Software Engineering 3rd and 4th International Conference, ENASE 2008 / 2009, Funchal, Madeira, Portugal, May 4–7, 2008 / Milan, Italy, May 9–10, 2009, Revised Selected Papers.* Ed. by L. A. Maciaszek, C. González-Pérez and S. Jablonski. Vol. 69. Communications in Computer and Information Science (CCIS). Berlin, Heidelberg: Springer, pp. 206–219.

Atkinson, C., Tunjic, C. and Möller, T. (2015). ‘Fundamental Realization Strategies for Multi-view Specification Environments’. In: *19th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2015, Adelaide, Australia, September 21–25, 2015*. Ed. by S. Hallé, W. Mayer, A. K. Ghose and G. Grossmann. IEEE Computer Society, pp. 40–49.

Clark, T. and Frank, U. (2020). ‘Multi-level Modelling with the FMMLx and the XModelerML’. In: *Modellierung 2020, 19.–21. Februar 2020, Wien, Österreich*. Ed. by D. Bork, D. Karagiannis and H. C. Mayr. Vol. P-302. LNI. Gesellschaft für Informatik, pp. 191–192.

Codd, E. F. (1993). *Providing OLAP to User-Analysts: An IT Mandate.*

Frank, U. (2002). ‘Multi-perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages’. In: *Proceedings of the 35 Annual Hawaii International Conference on System Sciences (HICSS)*, pp. 1258–1267.

Frank, U. (2014a). ‘Multi-perspective enterprise modeling: Foundational concepts, prospects and future research challenges’. In: *Software and Systems Modeling* 13.3, pp. 941–962.

Frank, U. (2014b). ‘Multilevel modeling: toward a new paradigm of conceptual modeling and information systems design’. In: *Business & Information Systems Engineering* 6, pp. 319–337.

Gerbig, R. (2017). ‘Deep, Seamless, Multi-format, Multi-notation Definition and Use of Domain-specific Languages’. PhD thesis. University of Mannheim.

Iacob, M.-E., Jonkers, H., Lankhorst, M. M., Proper, E. and Quartel, D. A. (2012). *ArchiMate 2.0 Specification: The Open Group*. Van Haren Publishing. URL: http://doc.utwente.nl/82972/.

ISO/IEC (1997). ‘RM-ODP. Reference Model for Open Distributed Processing’. In: *ISO/IEC 10746, ITU-T Rec. X.901-X.904.*

ISO/IEC/IEEE (Jan. 2011). ‘Systems and software engineering – Architecture description’. In: *ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000)*, pp. 1–46.

Kramer, M. E., Burger, E. and Langhammer, M. (2013). ‘View-Centric Engineering with Synchronized Heterogeneous Models’. In: *Proceedings of the 1st Workshop on View-Based, Aspect-Oriented and Orthographic Software Modelling*. VAO ’13. Montpellier, France: Association for Computing Machinery, pp. 1–6.

Kühne, T. (2006). ‘Matters of (Meta-)Modeling’. In: *Software and Systems Modeling* 5.4, pp. 369–385.

Miller, J. and Mukerji, J. (2003). *MDA Guide Version 1.0.1*. Tech. rep. Object Management Group (OMG).

Schlieter, H., Burwitz, M., Schönherr, O. and Benedict, M. (2015). ‘Towards Model Driven Architecture in Health Care Information System Development’. In: *Smart Enterprise Engineering: 12. Internationale Tagung Wirtschaftsinformatik, WI 2015, Osnabrück, Germany, March 4-6, 2015*. Ed. by O. Thomas and F. Teuteberg, pp. 497–511.

Tunjic, C., Atkinson, C. and Draheim, D. (2018). ‘Supporting the Model-Driven Organization Vision through Deep, Orthographic Modeling’. In: *Enterprise Modelling and Information Systems Architectures : An International Journal* 13, pp. 1–39.

Tunjić, C. V. (2021). ‘A deep orthographic modelling environment’. PhD thesis. University of Mannheim.

Urbaczewski, L. and Mrdalj, S. (2006). ‘A comparison of enterprise architecture frameworks’. In: *Issues in information systems* 7.2, pp. 18–23.

Zachman, J. A. (1987). ‘A framework for information systems architecture’. In: *IBM Systems Journal* 26.3, pp. 276–292.

Chapter 10

The Product Perspective in Multi-Perspective Enterprise Modelling

What Information about Products is Relevant for Enterprise Models?

Kurt Sandkuhl

Most enterprise modelling approaches use multiple perspectives to provide guidance in the modelling process and to improve manageability of the resulting enterprise model. The perspectives are supposed to address particular concerns of an enterprise and to provide adequate description and modelling practices. The focus of this paper is on the product perspective, i. e., on constructs, structures and rules for modelling products and services of an enterprise. The contributions of the paper are (a) an analysis of product information relevant for enterprise modelling, (b) an overview to prominent enterprise modelling approaches and how they support product modelling, and (c) selected practices for product modelling. We argue that the trend of digital transformation of enterprises motivates more support for product modelling in the context of enterprise modelling.

10.1 Introduction

Many enterprise modelling (EM) approaches use multiple perspectives when analysing the current situation of an organisation or designing the future situation. The motivation for multiple perspectives in general is to reduce complexity of the modelling process and to improve manageability of the resulting enterprise model (cf. Section 10.3.1). Each perspective is supposed to address particular concerns of an enterprise and to provide adequate constructs, structures and rules. This supports modellers in understanding, analysing, capturing and representing what is relevant for different groups of stakeholders and/or different modelling purposes (Frank 2014). Examples of perspectives often visible in enterprise modelling approaches are the process perspective (i. e., the work processes of an enterprise with their decomposition in tasks, information flows and control flows) or the organisation structure perspective (i. e., the structure of organisation units with decision structures, roles and positions). The focus of this paper is on a perspective that is not explicitly mentioned in most EM approaches: the product perspective, i. e., the products offered by an enterprise with decomposition structure, features, target groups, and other product-related information.

From an economic viewpoint, the purpose of an enterprise is commonly the creation of value for its customers and profits for its owners by providing defined products or services meeting the customers' needs and fulfilling regulatory, environmental, ethical, and market standards. Consequently, a substantial part of an enterprise's operation has to focus on design, development, marketing and delivery of these products and services in congruence with the enterprise's business model and strategy. In this context, many application areas for enterprise modelling exist that motivate the inclusion of product information in enterprise models, for example, to capture or design the processes required for product/service delivery, roles and organisational units responsible for product/service components, or dependencies between products and services.

Given the importance of products and services for an enterprise, enterprise modelling languages and frameworks could be expected to offer modelling constructs for representing the product/service information relevant to a given modelling purpose. However, most EM languages do not offer explicit concepts for product and service modelling (cf. Section 10.3.2). There are many approaches for modelling products in the field of mechanical engineering and production systems (cf. Section 10.2), but these approaches are often addressing aspects of (physical) products (geometric shape, material specification, surface design, mechanical characteristics, etc.) that are outside the scope of EM projects. Integrating these approaches with EM methods has not been widely done and is considered challenging because of different methodological approaches. In the context of digital transformation (cf. Section 10.4), many companies change their business models with new variants of products and services. This transformation requires a detailed understanding of dependencies between products, services, business processes, and organisation structures, which would motivate the use of enterprise modelling techniques.

The focus of this paper is on the characteristics of the product perspective with a focus on four questions:

- What product information is relevant for enterprises and should potentially be included in enterprise models?
- How do prominent enterprise modelling methods support the product perspective?
- What practices for product modelling exist?
- What developments related to digital transformation motivate product modelling in enterprises?

The remainder of the paper is structured following the above questions: Section 10.2 investigates the different facets of product information in light of various application scenarios. Section 10.3 analyses multi-perspective enterprise modelling approaches and how they tackle product information. Section 10.3 also describes selected practices for product information in enterprise modelling. Section 10.4 gives an outlook on future challenges in the field resulting from digital transformation.

10.2 Product Information and its Relevance for Enterprise Modelling

This section investigates what product information is relevant in enterprises and what business challenges causing EM projects would motivate a product perspective in EM. The different facets of product information have been subject to research in industrial organisation, product and production engineering for many years. The following non-exhaustive collection originates from work on product lifecycle management (Stark 2020), standardisation activities (Chungoora et al. 2013), manufacturing environments (Lu et al. 2019), and the core product model (Fenves et al. 2008). Product information includes:

- *Product function* describes what the product does or provides (features, functional classifications), i. e., how the product is supposed to satisfy customer needs and engineering requirements.
- *Product form* specifies the selected solution for the design problem given by the product function. Characteristics of physical form are defined in terms of its geometry (2D, 3D, surface model, etc.) and material (material specification, recycling options, sources, etc.).
- *Product behaviour* specifies how the product's form implements the intended function, for example by defining a behavioural model. This includes interaction with other products or product functions and implies dependencies and configuration constraints.
- Information for *sales and distribution* includes illustrative pictures and drawings; descriptions of functionality, target groups, and application possibilities; customer-oriented technical specifications; weight, size and other data for logistics and delivery; pricing and distribution information.
- Information for *production management* includes customer orders; planned manufacturing; required resources and suppliers; items in stock with their configuration/-product variant, etc.
- Information for *product life-cycle management* including the different phases (from target setting to end-of-life recycling), stakeholder concerns (user, designer, producer, owner, etc.) and product structures (part families and features, part versions, manufacturing facilities and resources, manufacturing process and methods).
- Information about *manufacturing projects* for certain products including the bill of material; components, assembly or parts required for manufacturing; features of manufacturing methods needed; resources related to the features and providers of the resources.

Not all information included in the list above is suitable for representation in enterprise models. 3D geometry models, surface specifications, or behavioural models, to take only three examples, require other ways of representation than those offered by EM. However, much information in the above list has already been subject to conceptual modelling, for example, product functions, sales and distribution, or product management information, and could be integrated into EM meta-models. Furthermore, much of the product information also is relevant for strategic or operational activities in an enterprise. Depending on the modelling purpose this information can be a relevant part of enterprise models, for example in the following scenarios:

- *Process innovation*: changes in the business model of an organisation often cause the re-design of existing work processes or the introduction of substantially changed processes. If the business model change concerns the product and services offered to the customer, the product life-cycle processes (target setting, product design, production design, manufacturing, maintenance) are affected and might have to be adapted.

- *Information system development*: most enterprises nowadays use information systems in all business functions and also aim at an enterprise-wide integration of available information for support of operational planning and strategic decision making. Information systems supporting the product life-cycle and the economic management need to be integrated, which often requires developing new information systems supporting this purpose.

- *Product innovation*: enterprises usually assign the responsibility for certain products, their components or variants to organisational roles. When new products or variants are introduced, the responsibility may have to change, which also can have an impact on accountability or responsibilities for the product-related processes.

In all the above scenarios, an understanding of the current situation in an organisation, the evaluation of potential change options, and the definition of the future situation are required. Due to mutual dependencies between organisation structure, process, IT support and product information, all these perspectives would have to be considered in EM projects.

10.3 Product Information in Multi-Perspective Enterprise Modelling

10.3.1 Terminology

Enterprises are complex socio-technical systems with their different organisational units and people working in the enterprise, with workflows and production processes, products and services offered to different customer groups, suppliers and business partners, IT systems and production resources. Enterprise modelling is defined by Bubenko et al. (2001) as follows:

> *'Enterprise Modelling (EM) is the process of creating an integrated enterprise model which captures the aspects of the enterprise required for the modelling purpose at hand. An enterprise in this context can be a private company, government department, academic institution, other kind of organisation, or part thereof. An enterprise model consists of a number of related sub-models, each focusing on a particular aspect of the enterprise, e. g., processes, business rules, concepts/information, vision/goals, and actors. An enterprise model describes the current or future state of an enterprise and contains the commonly shared enterprise knowledge of the stakeholders involved in the modelling process.'*

Given the complexity of enterprises, in the course of modelling an enterprise there is the need to understand, analyse, capture and represent what is relevant for different stakeholders and/or modelling purposes. In this context, there seems to be an agreement in the academic literature related to enterprise modelling that a key feature of an enterprise model is that it includes various perspectives. Frank (2014), e. g., states that *'a perspective as a psychological construct constitutes a conception of reality, comparable to a particular viewpoint in spatial perception [...], which helps to reduce complexity by constituting sense [...].'* Different perspectives also allow for some kind of triangulation, i. e., looking at the subject under investigation from different perspectives often reveals dependencies or relationships not visible from only one perspective. For this paper, *perspective* is defined as follows:

> In the context of enterprise modelling, perspective refers to the conventions for expressing a sub-model with respect to a set of concerns. Conventions include description practices (e. g., common terminology, concepts, abstraction level) and modelling practices (e. g., procedure, notation, tools). The set of concerns should be relevant to defined stakeholder groups.

This definition, by intention, connects to Frank's explanation of 'perspective' above and is inspired by the definition of ISO 42010's viewpoints (ISO/IEC/IEEE 2011) in architecture descriptions. In the definition of enterprise given above, aspect is used as a synonym for perspective.

10.3.2 Perspectives of Prominent Enterprise Modelling Approaches

The main focus of this section is to investigate how prominent enterprise modelling methods support the product perspective. As a starting point for investigating this question, we had to compile a list of existing EM methods. We used scientific articles and textbooks providing an overview of the field of enterprise modelling. As enterprise modelling has attracted much research in both, computer science and information systems, and also in industrial organisation and enterprise engineering, we included overview articles from both areas. More concretely, we used the survey articles by Bock et al. (2014), Vernadat (2020) and Sandkuhl et al. (2014). From the methods listed in the above survey articles, we only included methods in our investigation that use several perspectives. Many approaches do not explicitly use the term perspective, but, e. g., viewpoint, dimension or aspect, which we treated as synonyms if they were used with the same intention. For each approach, we extracted (a) the perspectives in the focus of the modelling approach and (b) if an explicit product perspective exists or how product modelling is supported. Following the definition in Section 10.3, the product perspective should provide description and modelling practices for product information relevant to stakeholders concerned with the organisation's product-related activities. The result is shown in Table 10.1. The EM approaches in Table 10.1 show different ways of support for modelling the product information as part of enterprise models. Only two methods, AKM and 4EM, include an explicitly defined product perspective. AKM emphasises the need to adapt the modelling constructs to the case at hand; 4EM only provides support for product structures and features. MEMO does not offer a product perspective, but a meta-meta-model approach that would allow for defining a product perspective or aspect that is integrated with the other MEMO aspects. GIM provides a product construct in the modelling language, which supports capturing of product meta-data, but no explicit product function, structure, behaviour or management. Additional information can be modelled in the physical system. All other EM approaches allow for capturing product information in another perspective, which is usually the one allowing for information modelling.

10.3.3 Examples for Description and Modelling Practices for Product Information in Enterprise Modelling

The intention of this section is to illustrate description and modelling practices for a product perspective. The examples are the POPS* modelling practice and the product model developed in the MAPPER project (Sandkuhl 2010). The roots of the POPS* (pronounced 'pop star') practice are the ATHENA POP* (Chen et al. 2009) and the AKM method (Lillehagen and Krogstie 2008). This practice is also used in the 4EM method (Sandkuhl et al. 2014). POPS is the abbreviation for four modelling perspectives:

- Process (P) – work processes including control or information flow, tasks to be performed and their description.
- Organisation (O) – organisation structure (organisation units, like business areas, departments or boards, and their relationships) including organisational roles and, if required for the modelling purpose, the competencies needed.
- Product (P) – the products and/or services of an enterprise including their product or service composition structure, the different elements, features and functions.
- Systems (S) – IT and information systems of the enterprise including IT architecture, information model and resources, tools and machinery.

Table 10.1: Multi-perspective EM approaches and their support of a product perspective

EM approach	Primary perspectives	Support for Product Modelling
Computer-Integrated Manufacturing Open Systems Architecture (CIMOSA) (Kosanke 1995)	Function, information, resource and organisation views	Information about products are supposed to be included in information view
GIM (GRAI Integrated Methodology) (Chen et al. 1997)	Three subsystems and modelling methods: decision system, physical system, information and functional system	Product has to be included in information system and, in some cases, physical system (stock/resource)
Active Knowledge Modelling (AKM) (Lillehagen and Krogstie 2008)	Process, organisation, product, (IT-) systems perspectives; complementary perspectives may be added	Product perspective
ArchiMate (Lankhorst et al. 2009)	Business, information, application and technology architecture	'Product' construct and viewpoint exist, but are meant only for products that consist of business, application, or technology services
Architecture of Integrated Information Systems (ARIS) (Scheer 2013)	Traditional: control (business process, event), data, organisation, and function	Information about products can be included in data
Multi-Perspective Enterprise Modelling (MEMO) (Frank 2014)	Combination of four aspects (resource, structure, process and goal) and three perspectives (strategy, organisation, information systems)	Information about products can be included in information systems perspective; meta-meta-model would also allow for definition of specific product perspective
Business Engineering (BE) (Österle and Winter 2013)	Strategy, process, systems; extensions also include data, leadership, user, and products and services	Only in extension: product and service perspective
Design and Engineering Methodology for Organisations (DEMO) (Dietz 2006)	InterAction Model (IAM), InterStriction Model (ISM), Business Process Model (BPM), Transaction Process Model, the Action Model (AM), and Fact Model (FM)	Information about products can be included in fact model
For Enterprise Modelling Method (4EM) (Sandkuhl et al. 2014)	Goal/problem, business process, actor, business rule, concept, technology, product/service perspectices	Product/service perspective

The * in POPS* is meant as a *wildcard* to add any other perspective required for the modelling purpose. The *relationship* oval in Figure 10.1 illustrates that the POPS perspectives are mutually dependent on each other as the relationships between elements of the different perspectives mean that changes in elements of one perspective also might affect the related elements in other perspectives. Examples: adding or removing process steps will result in changing the roles responsible for the process steps; merging two product components will affect the production process and the process responsible role; extending functionality of an information system might require adaptation of the process it supports and training of the role working with it. The practice can be summarised as the following mnemonic: *Independently of the modelling purpose, remember to always investigate for each of the POPS perspectives if and to what extent they have to be included in modelling because they are mutually dependent and you might only get the full picture if you understand the dependencies. Add additional perspectives depending on the modelling purpose. Always start the modelling with one of POPS perspectives; select the one to start with based on the modelling purpose.*

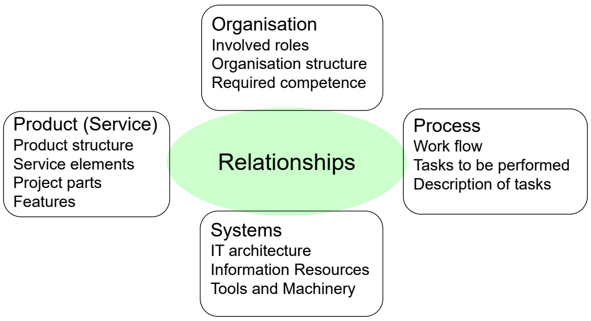


Figure 10.1: Perspectives of the POPS* best practice

This is a modelling practice relevant to the product perspective, because it makes sure that product concerns receive the same attention as the process, organisation structure and systems concerns. One result of the MAPPER project was a meta-model for enterprise modelling with a well-developed meta-model for modelling product information and the relevant relationships to processes, organisation structures, information systems and strategic objectives of an enterprise. This product model was developed for application cases in the automotive industry, electrical engineering and automotive supplier industries. Figure 10.2 shows an instantiation of this meta-model (not including the relationships to other modelling perspectives) for an automotive supplier. The model in Figure 10.2 includes the structure of products and their components (modelled as system and modules structures), the parameters used to configure them and their grouping into product families, the (customer) requirements, resulting product properties and functions, rules for platform integration and design, and production information.

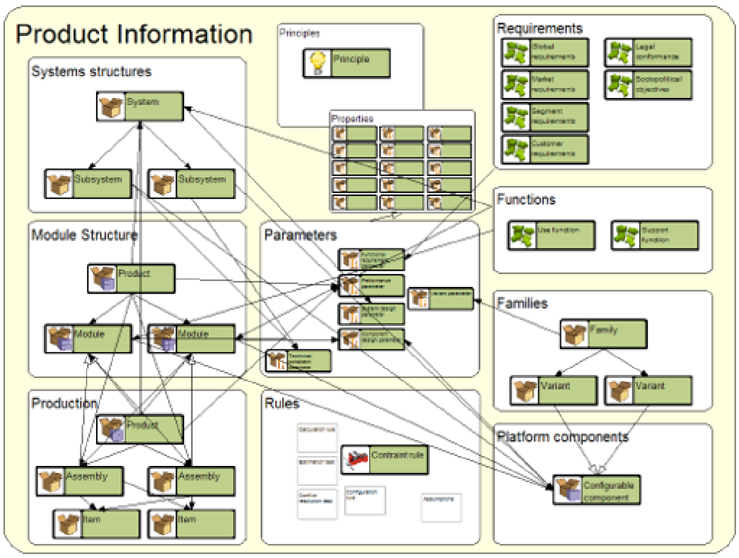


Figure 10.2: Example of a product model

10.4 Digital Transformation and Product Modelling

The analysis of the product perspective's visibility in multi-perspective EM approaches showed that the product perspective is under-represented compared to other perspectives (cf. Section 10.3.2). For approaches originating from IS engineering, one reason for this fact is that product information could be treated in the same way as any other resource-related information of an enterprise by representing them in the information model. Approaches from enterprise engineering often refer to existing product modelling approaches, like STEP (Pratt et al. 2001), BoM (Hegge and Wortmann 1991) or CPM2 (Fenves et al. 2008), as motivation for only including meta-data of products or for using the information model for capturing product information. Furthermore, meta-modelling approaches allow for development of a domain-specific modelling language for EM-inclusive product modelling. However, we argue that in particular the digital transformation of business models has caused changes in enterprises that call for more explicit description and modelling practices for product information in enterprise modelling, i. e., for an explicitly defined product perspective. The main reason is that the number of stakeholders concerned with product information and integrating products in enterprise activities steadily increases due to the digitalisation of products.

Digital transformation (DT) is *'concerned with the changes digital technologies can bring about in a company's business model, which result in changed products or organisational structures or in the automation of processes'* (Hess et al. 2016). Examples of developments closely related to DT are

- Products as *service frontline technologies*: in many traditional service domains, such as banking, tourism or healthcare, some of the human service desk employees have been replaced by a technological solution. Furthermore, also in traditional product domain, the product is combined with services and becomes the frontline of the service to the customer. Frontline service technologies (De Keyser et al. 2019) can be products on their own, nevertheless also have to be integrated into service delivery and management.
- *Product-IT integration in enterprise-IT*: the increasing number of cyber-physical systems (Baheti and Gill 2011) and smart connected products (Porter and Heppelmann 2014) have resulted in tighter integration of these systems and products into the enterprise architecture (Kaidalova et al. 2018). IT components built into products (also called 'product-IT') are increasingly part of IT service delivery and the corresponding business and service management processes, and, thus, have to be integrated into the enterprise-IT.
- *New product/service categories*, such as quantified products: Quantified products (QP) (Sandkuhl 2022) are a new product category that exploits data of individual product instances and fleets of instances. A quantified product is a product whose instances collect data about themselves that can be measured or, by design, leave traces of data. The QP design has to consider that modelling dependencies exist between the modelling of the actual product, services related to the product, and the digital ecosystem of the services.

References

Baheti, R. and Gill, H. (2011). ‘Cyber-physical systems’. In: *The impact of control technology* 12.1, pp. 161–166.

Bock, A., Kaczmarek, M., Overbeek, S. and Heß, M. (2014). ‘A comparative analysis of selected enterprise modeling approaches’. In: *IFIP Working Conference on The Practice of Enterprise Modeling*, pp. 148–163.

Bubenko, J. A., Persson, A. and Stirna, J. (2001). ‘User guide of the knowledge management approach using enterprise knowledge patterns’. In: *Stockholm (Sweden), Department of Computer and Systems Science, Royal Institute of Technology.*

Chen, D., Knothe, T. and Doumeings, G. (2009). ‘POP* meta-model for enterprise model interoperability’. In: *IFAC Proceedings Volumes* 42.4, pp. 175–180.

Chen, D., Vallespir, B. and Doumeingts, G. (1997). ‘GRAI integrated methodology and its mapping onto generic enterprise reference architecture and methodology’. In: *Computers in industry* 33.2-3, pp. 387–394.

Chungoora, N., Cutting-Decelle, A.-F., Young, R. I. M., Gunendran, G., Usman, Z., Harding, J. A. and Case, K. (2013). ‘Towards the ontology-based consolidation of production-centric standards’. In: *International Journal of Production Research* 51.2, pp. 327–345. DOI: 10.1080/00207543.2011.627885.

De Keyser, A., Köcher, S., Alkire, L., Verbeeck, C. and Kandampully, J. (2019). ‘Frontline service technology infusion: conceptual archetypes and future research directions’. In: *Journal of Service Management* 30.1, pp. 156–183. DOI: 10.1108/JOSM-03-2018-0082.

Dietz, J. L. G. (2006). *What is Enterprise ontology?* Springer.

Fenves, S. J., Foufou, S., Bock, C. and Sriram, R. D. (2008). ‘CPM2: A Core Model for Product Data’. In: *Journal of Computing and Information Science in Engineering* 8.1. DOI: 10.1115/1.2830842.

Frank, U. (2014). ‘Multi-perspective enterprise modeling: Foundational concepts, prospects and future research challenges’. In: *Software & Systems Modeling* 13.3, pp. 941–962.

Hegge, H. M. and Wortmann, J. (1991). ‘Generic bill-of-material: a new product model’. In: *International Journal of Production Economics* 23.1-3, pp. 117–128.

Hess, T., Matt, C., Benlian, A. and Wiesböck, F. (2016). ‘Options for formulating a digital transformation strategy’. In: *MIS Quarterly Executive* 15.2, pp. 123–139.

ISO/IEC/IEEE, ed. (2011). *ISO/IEC 42010: Systems and Software Engineering - Architecture description.*

Kaidalova, J., Sandkuhl, K. and Seigerroth, U. (2018). ‘How digital transformation affects enterprise architecture management–a case study’. In: *International Journal of Information Systems and Project Management* 6.3, pp. 5–18.

Kosanke, K. (1995). ‘CIMOSA—overview and status’. In: *Computers in Industry* 27.2, pp. 101–109.

Lankhorst, M. M., Proper, H. A. and Jonkers, H. (2009). ‘The architecture of the Archimate language’. In: *Enterprise, business-process and information systems modeling*. Springer, pp. 367–380.

Lillehagen, F. and Krogstie, J. (2008). *Active knowledge modeling of enterprises.* Springer.

Lu, Y., Wang, H. and Xu, X. (2019). ‘ManuService ontology: A product data model for service-oriented business interactions in a cloud manufacturing environment’. In: *Journal of Intelligent Manufacturing* 30.1, pp. 317–334. DOI: 10.1007/s10845-016-1250-x.

Österle, H. and Winter, R. (2013). *Business Engineering: Auf dem Weg zum Unternehmen des Informationszeitalters*. Springer.

Porter, M. E. and Heppelmann, J. E. (2014). 'How smart, connected products are transforming competition'. In: *Harvard business review* 92.11, pp. 64–88.

Pratt, M. J. et al. (2001). 'Introduction to ISO 10303—the STEP standard for product data exchange'. In: *Journal of Computing and Information Science in Engineering* 1.1, pp. 102–103.

Sandkuhl, K. (2010). 'Capturing product development knowledge with task patterns: evaluation of economic effects'. In: *Control and Cybernetics* 39.1, pp. 259–273.

Sandkuhl, K. (2022). 'Features of Quantified Products and Their Design Implications'. In: *International Baltic Conference on Digital Business and Intelligent Systems*. Springer, pp. 152–163.

Sandkuhl, K., Stirna, J., Persson, A. and Wißotzki, M. (2014). *Enterprise Modeling*. Springer.

Scheer, A.-W. (2013). *Architektur integrierter Informationssysteme – Grundlagen der Unternehmensmodellierung*. Springer.

Stark, J. (2020). 'Product lifecycle management'. In: *Product lifecycle management*. Vol. 1. Springer, pp. 1–33.

Vernadat, F. (Nov. 2020). 'Enterprise modelling: Research review and outlook'. In: *Computers in Industry* 122, p. 103265. doi: 10.1016/j.compind.2020.103265.

Chapter 11

Bridging the Mental and the Physical World: Conceptual Modeling and Augmented Reality

Hans-Georg Fill and Fabian Muff

Whereas conceptual modeling is today widely used for representing knowledge for the purpose of communication and understanding, the combination with augmented reality technologies permits for the first time to anchor this knowledge formally to objects in the physical world using electronic means. In addition, conceptual modeling may help to support the design of complex augmented reality applications and thus enable non-technical users to better engage with this technology. In this chapter, we thus explore the combination of conceptual modeling and augmented reality by focusing on the role of the subject and how its perception is augmented using augmented reality technologies. From this, we derive two directions in the form of a. *Augmented Reality-based Metamodeling and Modeling*, and b. *Knowledge-based Augmented Reality* and illustrate them with recent examples.

11.1 Introduction

The benefits of *models* are today widely known and largely undisputed. One of the probably most known examples includes architectural models that are an indispensable prerequisite for constructing houses (Zachman 1987). Thus, models not only stand for the *artistic* creation of an artifact (Thalheim 2012b). They also constitute a representation of the *knowledge* that is required for assembling a building's components in such a way that it meets the desired properties of its future owner (Karagiannis et al. 2017; Abazi et al. 2011; Johannsen and Fill 2015), e. g., in terms of stability, energy efficiency, comfort, or legal and safety requirements. For achieving this kind of knowledge representation, we will focus our discussion on the field of *conceptual modeling* that requires the application of a modeling language, which pre-defines the concepts to be used in models (Harel and Rumpe 2004; Thalheim 2012a; Stachowiak 1973). Conceptual modeling plays an important role in the fields of Computer Science and Business Informatics (Härer and Fill 2020), where it supports for example the elicitation of software system requirements, e. g., Yu et al. (2011), the creation of databases, e. g., Glässner et al. (2017), or the management of business processes and IT systems, e. g., Dumas et al. (2018) and Moser et al. (2022).

However, the adoption of conceptual modeling and its application in companies in the form of enterprise models is characterized by several challenges as recently laid out in Sandkuhl et al. (2016), Sandkuhl et al. (2018) and Frank (2014). Although conceptual modeling is common in many organizations, the complexity of the used modeling approaches often impedes their use by non-modeling experts. Thus, despite the benefits of modeling in everyday work practices, it is not sufficiently accessible by end-users. As a solution, various approaches have been proposed, including so-called *grass-roots modeling* where models are (semi-)automatically created from everyday office applications such as spreadsheet software (Sandkuhl et al. 2016) or multi-level modeling, where modeling languages can be more flexibly defined (Frank 2014). A further direction in this context is the use of novel interaction technologies such as virtual and augmented reality. Although 3D-based augmented and virtual reality approaches have been researched in the past, e. g., for generating models by immersing users in a virtual work environment (Leinenbach et al. 1999), such techniques have recently been taken up again due to the technological advancements and lower costs.

With the advancement of technologies in computer vision, hardware and software components for realizing so-called *augmented reality* applications have become widely affordable and easier to us (Cipresso et al. 2018). This is due to the evolution of the computing power of smartphones that permits today to conduct real-time calculations for processing image and other sensor information and overlaying the images with virtual objects by just using consumer-grade hardware (Chatzopoulos et al. 2017). Further, an international standard for web-based virtual and augmented reality (WebXR devices API) is being actively developed[1] with first implementations being available on Android, iOS or head mounted devices, e. g., the Microsoft HoloLens[2].

These developments led to a revival of research for investigating how augmented reality may be combined with conceptual modeling, e. g., Muff and Fill (2021a), Muff and Fill (2021b), Muff and Fill (2022b), Grambow et al. (2021), Seiger et al. (2021), Grum and Gronau (2018) and Havard et al. (2015). Although several applications of using augmented reality together with conceptual models have been demonstrated, a fundamental discussion of how conceptual modeling and augmented reality are related seems to be missing so far. Our working hypothesis for the following elaboration will be that the combination of conceptual modeling and augmented reality brings about a fundamental change, as the traditional way of using IT-based conceptual models as explicit – however, non-tangible – representations of knowledge is abandoned. With augmented reality, IT-based conceptual models can for the first time be *anchored formally* to objects in the real world by using electronic means.

In this chapter, we aim to derive some of the fundamental constituents of combining augmented reality with conceptual modeling. The goal is to contribute to a theory of augmented reality-based conceptual modeling. Such a theory should inform about the solution space of augmented reality in conceptual modeling and permit to derive and evaluate new types of applications. A further goal of this contribution is to lay the foundation for joining augmented reality with conceptual modeling on a technical basis in the form of an AR-based metamodeling platform for easing the realization of new AR-based modeling methods. The remainder of this chapter is structured as follows. We will first lay some foundations on our view on conceptual modeling and a brief characterization of augmented reality. Subsequently, we will advance to derive a fundamental framework for combining conceptual modeling with augmented reality and discuss some concrete examples.

1 See `https://www.w3.org/TR/webxr/`

2 See `https://www.microsoft.com/en-us/hololens/`

11.2 Foundations

In the following we introduce some foundations necessary for the subsequent discussion and for ensuring a common terminology. This includes a brief characterization of our view on conceptual modeling and metamodeling, the role of the subject in conceptual modeling, as well as a short characterization of augmented reality.

11.2.1 Conceptual Modeling

A central aspect of models in general is their capability of *reducing complexity* of any system under study. This is accomplished through *abstraction*, whereby certain attributes are *omitted* or *transformed* in a way that makes the resulting representation easier to understand and process. In addition, models may contain attributes that are not found in the original system but that may have been added in the course of creating the model for further enhancing the utility of the model. These attributes are denoted as *abundant* attributes (Stachowiak 1973). The relationships are depicted in Figure 11.1 below.

For giving a concrete example for these aspects, consider the building plan of a house that only shows the arrangement of the walls but that does not give details on their color or surface properties – see Figure 11.2 parts 1 and 2. In another iteration, the models may be extended with such properties, up to a level that already resembles to a large degree the original – see part 3 in Figure 11.2. Further, abundant attributes may be introduced in the models that do not directly correspond to attributes of the system under study. In the example, this would be the tape measure in part 3, indicated by the label A. Of course, such a tape measure would not be actually built. Rather, it provides information for better understanding the model and its contents – a *bridging-hypothetical function* as denoted by Stachowiak (1973), p. 156. As we will see later, also augmented reality can be regarded as such a function by augmenting the perception of the real world.

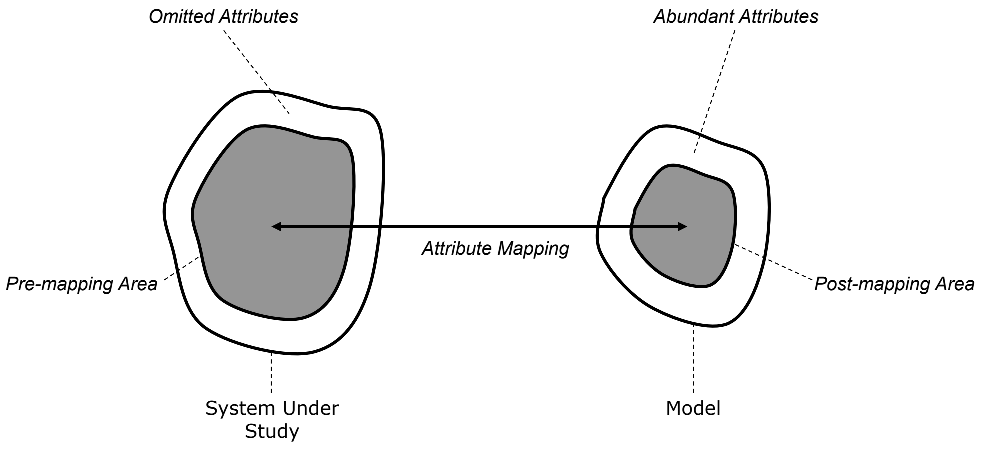


Figure 11.1: Mapping of attributes between a system under study and a model, adapted from Stachowiak (1973), p. 157.

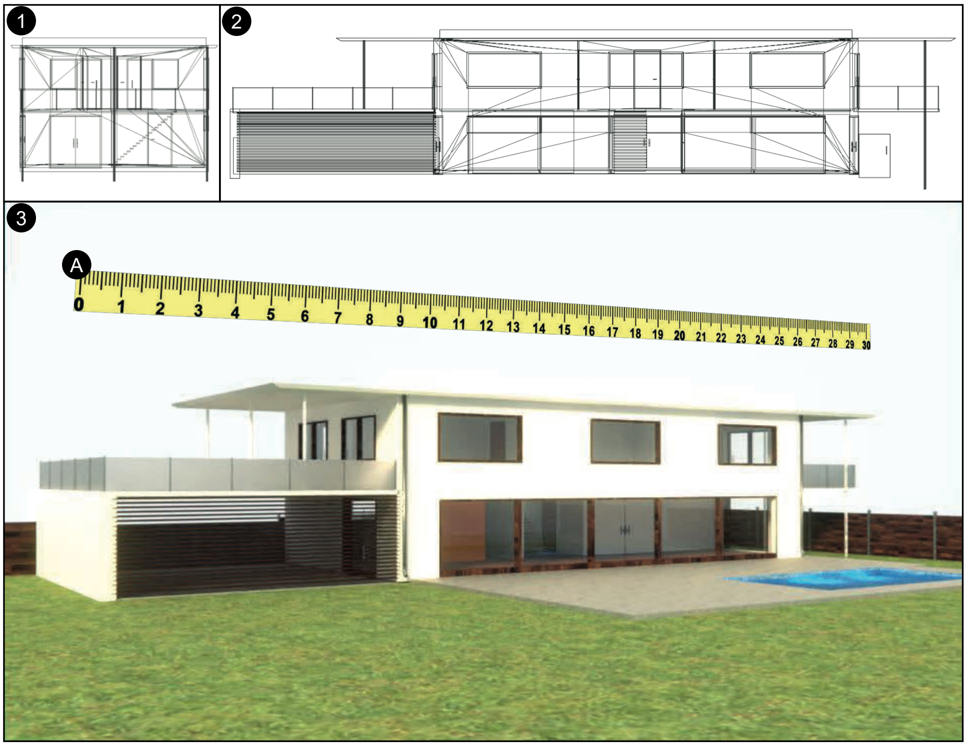


Figure 11.2: Models of a building with different abstraction levels and viewpoints

The example in Figure 11.2 features another aspect of models that we would like to highlight here, i. e., the different *viewpoints* that can be taken by a model. For traditional architectural models, the viewpoints show different two-dimensional perspectives as in parts 1 and 2 of Figure 11.2. Such viewpoints may not only show different geometric perspectives, e. g., as in Figure 11.2 from the left and the front side. They may even contain entirely different levels of abstraction or properties while referring to the same underlying original, Bork and Sinz (2013). In terms of model theory, this means that the attribute mapping is different, though the system under study stays the same. The viewpoints thus provide different insights for different groups of people or for different purposes according to the goals of the modeling activity, which we will take up later in the context of pragmatics and the role of the subject in conceptual modeling.

The field of *conceptual modeling* can be regarded as a sub-field of modeling that imposes further constraints on the way how models are created. In the following, we base our characterization of the field on prior work in Karagiannis and Kühn (2002), Ferstl and Sinz (2015) and Mylopoulos (1992), and complement it with our own notions. In contrast to the creation of models in general where it is not specified how the abstraction is accomplished, conceptual modeling pre-defines a set of concepts, i. e., *conceptual schemata* (Mylopoulos 1992), that have to be used for creating the models using a formal notation. These concepts have *semantics* assigned to them (Harel and Rumpe 2004), i. e., they carry a meaning which further constrains the form and content of the resulting models. Thereby, the concepts may

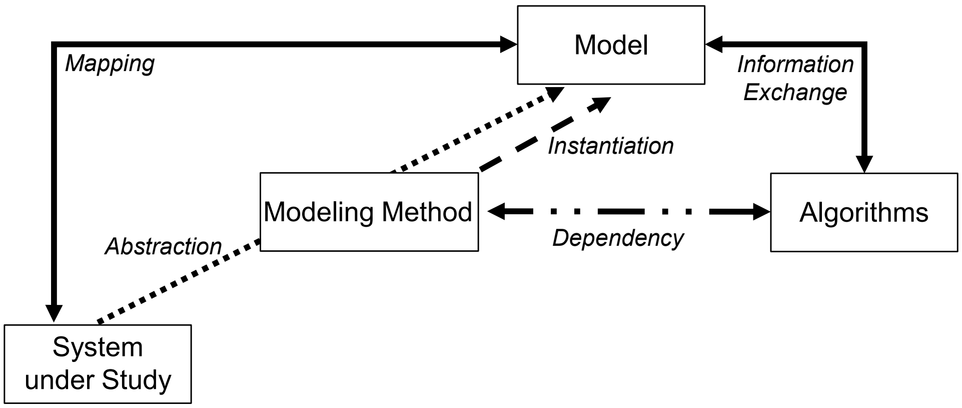


Figure 11.3: Conceptual modeling as a constraint-based abstraction where the modeling method constitutes the constraints for the instantiation of models

either be defined statically, or they may be changed at run-time as it is typical in multi-level modeling (Atkinson and Kühne 2001; Frank 2014).

Furthermore, these concepts may be complemented by a set of rules that define how to apply the concepts for creating the models. This is typically denoted as a *modeling procedure* (Karagiannis and Kühn 2002). We denote in the following the combination of the concepts and the modeling procedure as a *modeling method* – see Figure 11.3. The application of the modeling method for the creation of models is then the *instantiation* of the modeling method[3]. For the machine-based processing of the models, *algorithms* may be added (Karagiannis and Kühn 2002). These depend on the used modeling method as they have to consider the pre-defined semantics and rules of the modeling method and can exchange information with the models. Finally, the model itself features a mapping to the system under study as assumed also in general model theory (Stachowiak 1973; Ferstl and Sinz 2015).

Apart from these components and relationships, conceptual modeling features special pragmatic characteristics. The traditional goal of conceptual modeling is to describe aspects of the physical and social world for the purpose of human understanding and communication (Mylopoulos 1992). It has thus not been directed towards the interpretation of the model content by machines. However, the machine-based processing of model information can further increase the value of models as apparent in model-driven engineering (Brambilla et al. 2017) or recent approaches in enterprise modeling (Fill et al. 2021). This permits to apply a variety of technologies to models for interpreting their content, e. g., for semantic reasoning and data processing (Fill 2017; Smajevic et al. 2021), code generation (Kelly et al. 2013) or as direct interfaces to other systems and data representations (Fill and Johannsen 2016; Chis-Ratiu and Buchmann 2018). For these purposes, additional information may have to be added to the models, e. g., via extensions or annotations (Fill 2018).

3 It shall be denoted that this term is not to be confused with the instantiation of classes in a modeling language.

11.2.2 Metamodeling

When applying conceptual modeling in practice, the use of pen-and-paper approaches is obviously not adequate for dealing with large models or for the machine-based processing of information. Therefore, IT-based approaches are necessary. These can either come in the form of drawing tools or dedicated modeling tools. Whereas the former may include pre-defined shapes for more easily creating models, dedicated modeling tools permit a much easier creation and processing of models. In particular, these tools typically include mechanisms for enforcing the correct use of the conceptual schema and the modeling procedure and may provide data interfaces that can be accessed by algorithms for exchanging information.

The activity for designing suitable modeling tools is subsumed here under the term *metamodeling*, which includes their technical implementation in software. Metamodeling not only concerns the description of a conceptual schema in a formal notation and its technical implementation, but also decisions on how to implement a modeling procedure and algorithms. As a solution approach, language-based approaches as used in computer science have been found suitable and are today widely accepted. These comprise the definition of syntax, semantics, and pragmatics for the conceptual modeling languages (Harel and Rumpe 2004; Harel and Rumpe 2000). They revert to formal or semi-formal languages for describing the conceptual schema and the resulting models (Fraser et al. 1994; Bork and Fill 2014).

It is typically distinguished between two different language or semantic levels: an *object language* that is used for the description of something and a *meta language* that is used for describing the object language (Strahringer 1998). There may exist multiple levels of object and meta languages, e. g., also a meta-meta language that describes a meta language and a meta-meta-meta language describing the meta-meta language and so on. This corresponds to the hierarchy of formal (logic) languages as mentioned, e. g., by Russell (1922) who stated that one cannot talk about the structure of a language in a given language itself but you rather need another language that deals with the structure of the first language and that itself has another structure. This further relates to the infinite regress of semantics where a *definiendum* is described by a *definiens*, which is again the definiendum of another definiens et cetera (Messer 1999). Thereby, a meta language may either be *static* as found in traditional modeling approaches where the constructs of a language remain fixed; or, they may be *dynamic* as found in multi-level modeling approaches where constructs in a meta language may be changed flexibly and additional language levels can be introduced on the fly (Frank 2014).

The main advantage of using several hierarchies of meta languages in conceptual modeling is that those concepts that remain fixed over a longer period of time can be assigned a higher meta level, while concepts that need to be adapted more frequently are part of the lower levels. This brings benefits in terms of so-called *metamodeling platforms* that are software applications supporting the specification of languages and the instantiation of modeling methods. In an ideal case, such platforms only need to implement the highest required meta level that remains the same for a long time and foresee mechanisms for composing any lower language levels. Such an approach corresponds well to scientific research where fundamental, long-term concepts are sought after in the form of theories that can be applied to a large number of concrete cases.

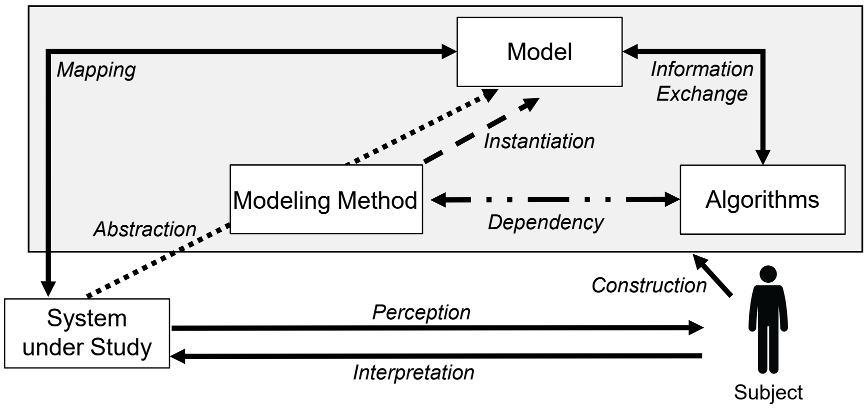


Figure 11.4: Role of the subject in conceptual modeling

11.2.3 The Role of the Subject in Conceptual Modeling

When we will later discuss the alignment of augmented reality and conceptual modeling, the role of humans will be an important part as augmented reality is inherently linked to human senses. Similarly, humans play an important part in conceptual modeling as conceptual models are intended to add to human understanding and communication. The perception of the system under study, its interpretation and the creation of a modeling method, the models, and the algorithms always involves a human subject (Ferstl and Sinz 2015) – see Figure 11.4. Further, despite the fact that the definition of modeling methods and their implementation in according tools has been greatly simplified throughout the last years, it is usually a collaborative effort of several people to engage in method design and even more so for the creation of models. Thus, the 'subject' may well involve several people.

The proposed view on the subject in conceptual modeling can even be extended to machine-created models as found for example in workflow and process mining (Herbst and Karagiannis 1998; Van Der Aalst 2012) or enterprise architecture mining (Perez-Castillo et al. 2019). Also, for these scenarios where models are created automatically, there is a subject involved for defining the modeling language, the algorithms and configuring the creation of the models from data sources as well as in the interpretation of the system under study.

11.2.4 Augmented Reality

Augmented reality is traditionally regarded as a technology to embed virtual images generated by a computer into the real world environment (Zhou et al. 2008). A widely used definition of AR has been proposed by Azuma (1997). He describes AR as a technology that merges the real and virtual worlds and is interactive in real time. This includes creating a three-dimensional mapping of virtual and real objects based on interactions.

From a technical point of view, we can classify AR technology by different sensors for the input and output of information and by components for processing this information (Dörner et al. 2019). For the input of information, different types of sensors are used, e. g., motion, position, audio or camera sensors. On a more conceptual level, not only sensor information is used as input by the device. As described in Muff and Fill (2022a), also explicit

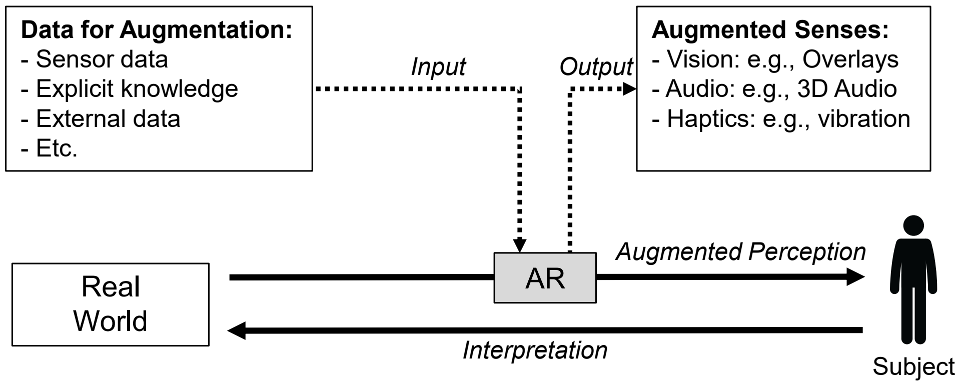


Figure 11.5: Concepts and relations in augmented reality

knowledge, such as model information or semantic information, as well as other external data, such as information on a web page, can serve as input – see upper part of Figure 11.5.

This information is processed by the AR device, which is today typically a smart phone, tablet or a head-mounted display. The AR device processes the input information and generates information that augments the user's perception of the real world. By augmenting the user's senses, e. g., vision, haptics, or audio, the perception can be augmented to allow for improved interpretation of the real world. Therefore, the nature of the concept is more precisely denoted as *augmented perception* rather than *augmented reality*, although the latter term is today more common.

11.3 Bridging the Mental and the Physical World

With the foundations presented above we can now advance to putting all parts together. In particular we are interested in how conceptual modeling and augmented reality may be combined on a meta level, i. e., without considering already concrete applications but rather aligning them from a fundamental perspective. Such a combination will then serve as a theoretical basis for deriving new types of applications and further serve for a technical reference implementation in the future.

For joining the two areas, we start from the *subject*, which plays a central role both in conceptual modeling and augmented reality. In conceptual modeling, the subject constructs the components of the model space including the modeling method, the models, and the algorithms. It does so by perceiving and interpreting a system under study. In augmented reality, the perception of the subject is augmented by data that comes from sensors, knowledge representations or other sources. We thus take the *augmented perception* of the subject as a foundation for joining it with the view developed for conceptual modeling. As shown in Figure 11.6, this results in two major directions based on the mapping between models and the real world through AR: (i.). *AR-assisted Metamodeling and Modeling* where functionalities of AR are used in the model space and (ii.). *Knowledge-based AR* where information from the model space is fueled into an AR application. In both cases, the perception of the subject is enhanced through AR, either through modeling-enhanced AR or for constructing parts in the model space.

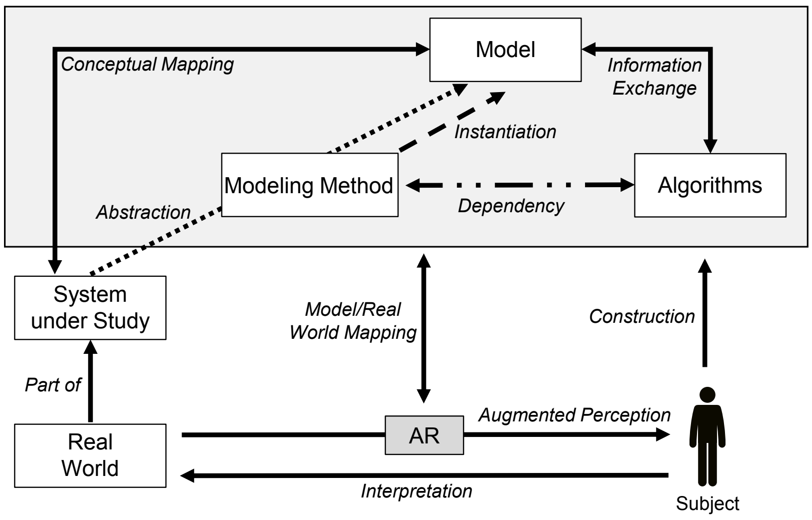


Figure 11.6: Join conceptual modeling & augmented reality from the viewpoint of the subject

11.3.1 Augmented Reality-assisted Metamodeling and Modeling

In this direction, the tasks for constructing modeling methods, algorithms, and models are enhanced using augmented reality techniques. It would include for example the elicitation of models using a head-mounted AR device together with gestures or voice commands that are recognized by the device and translated into modeling actions. This could be useful in environments where the use of traditional interfaces for modeling is not feasible or convenient, e. g., where manual interaction is not possible such as in dirty or hands-free scenarios, or where remote collaboration is necessary. It would also include metamodeling actions that are accomplished using augmented reality, e. g., when assembling real-world images via AR as input for a new modeling language.

Several challenges will need to be solved when taking this direction. This encompasses for example the definition of new types of interfaces for interacting with the components in the model space such as gesture-based input of information or the use of voice commands. Further, the representation of components of the model space will need to be adapted to the capabilities and scope of the AR environment. This could for example be new types of visual representations of models that can be anchored in the physical world via AR or three-dimensional views on models.

Figure 11.7 shows a mockup of an application following this direction (Muff and Fill 2022b). It features an AR-based application for the collaborative modeling of a *Business Model Canvas* (BMC) (Osterwalder et al. 2010). Thereby, a BMC is created using AR smart glasses together with voice commands or gestures. Additional information in context of the selected area of the BMC can be visualized dynamically as external knowledge source. Further, avatars of remote collaborators may be visualized and models may be synchronized to enable location independent collaborative modeling in AR. This illustrates that AR does not directly augment reality, but rather the subject's perception of reality and thus his or her interpretation of the real world when creating models.

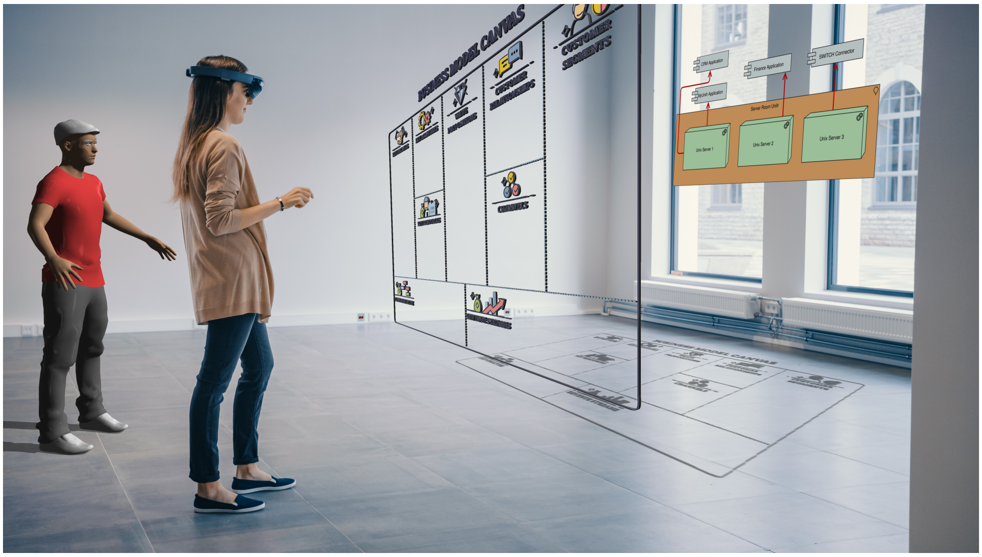


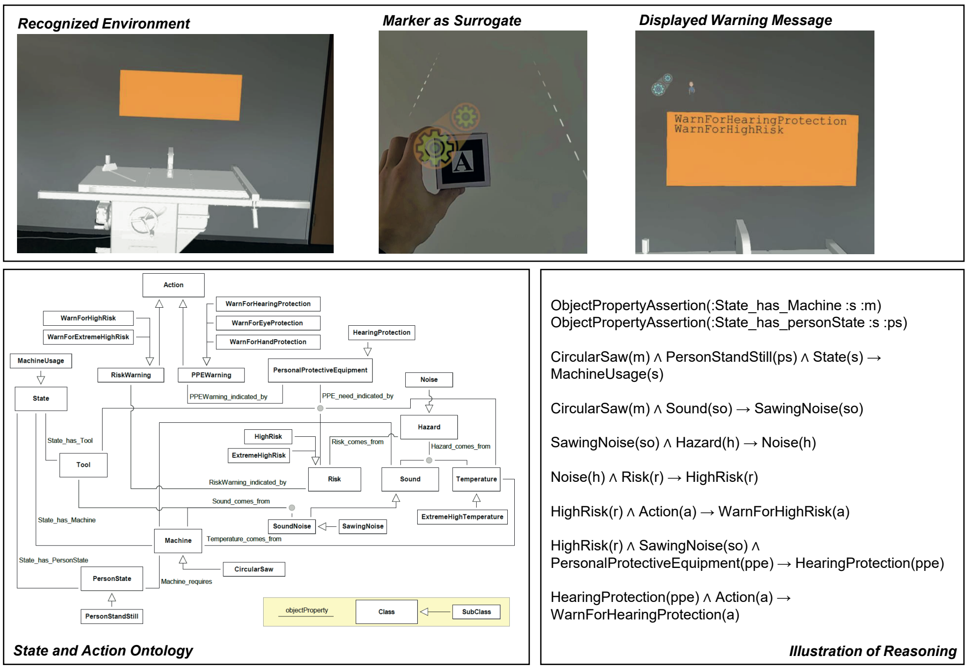
Figure 11.7: Example of a collaborative AR BMC modeling application. In the AR environment, users can work collaboratively on a BMC regardless of their location. The different parts of the BMC can be enhanced with relevant information from other models.

11.3.2 Knowledge-based Augmented Reality

The second direction focuses on the transfer of explicit knowledge in the form of models to an AR application. This comprises both design-time and run-time aspects, i. e., where models may be used for creating the *form* of AR applications, e. g., in the style of model-driven engineering, or where the information from models serves as data for augmentations, i. e., the *content* of AR applications. Despite the advancements in APIs and specialized development platforms, AR applications are still complex to develop. On the one hand, this is due to the numerous technologies involved in creating an AR application, including for example 3D design environments, sensors, or interaction devices. On the other hand, concrete usage scenarios need to be elaborated and the AR application has to be embedded in an enterprise's information system. For such tasks, the use of model-driven engineering could greatly help and ease the development.

Secondly, the data from models can be used for creating augmentations at run-time. Although it may look similar to the direction of AR-based modeling, the focus here is to use the information from the models for enhancing the AR experience. For example, an AR application may revert to a workflow defined through a model for deciding on which virtual objects to present to a user. Similarly, models may define complex scenarios over which it can be automatically reasoned by using sensor information from the AR device. This is shown in the example below where a process model for a carpentry process has been annotated with information from an ontology for reasoning over AR sensor information to display safety warnings. The markers thereby act as surrogates for a more advanced 3D object recognition (Muff and Fill 2022a).

The challenges in this direction include the mapping between information stemming from models to information sensed in the real world and in some cases even the real-time



ObjectPropertyAssertion(:State_has_Machine :s :m)
ObjectPropertyAssertion(:State_has_personState :s :ps)

CircularSaw(m) ∧ PersonStandStill(ps) ∧ State(s) → MachineUsage(s)

CircularSaw(m) ∧ Sound(so) → SawingNoise(so)

SawingNoise(so) ∧ Hazard(h) → Noise(h)

Noise(h) ∧ Risk(r) → HighRisk(r)

HighRisk(r) ∧ Action(a) → WarnForHighRisk(a)

HighRisk(r) ∧ SawingNoise(so) ∧ PersonalProtectiveEquipment(ppe) → HearingProtection(ppe)

HearingProtection(ppe) ∧ Action(a) → WarnForHearingProtection(a)

Figure 11.8: Example of knowledge-based augmented reality using ontology-based reasoning over sensor information collected from an AR device showing the recognized environment, markers as surrogates for 3D object recognition, the resulting warning message, the used state and action Ontology and an illustration of the reasoning process.

requirements in AR applications when processing model content. As AR applications typically tie information to some coordinates in the physical space, respectively to some recognized three-dimensional object, this needs to be aligned with information stemming from models, which is at best in a two-dimensional format or does not contain any position information at all so far. When the contents of models have to be processed in order to be suitable for the AR application, this has to be accomplished near real-time so that no lag in the user experience becomes visible. This is a particular challenge, e. g., for reasoners on mobile platforms – see Van Woensel and Abidi (2019).

11.4 Directions and Challenges for Future Research

From the insights discussed above we can derive several directions and corresponding challenges for advancing the combination of conceptual modeling and augmented reality. Despite some first elaborations on possible use cases for augmented reality and conceptual modeling – cf. Muff and Fill (2022b) – it will have to be further investigated which scenarios would be most beneficial for the combination of these technologies. Further, also the conceptual foundations for integrating AR in modeling and metamodeling frameworks will have to be further researched. Due to the high complexity of computer vision technologies and how they are effectively applied in augmented reality, this combination is a major challenge. For example, just the investigation of new forms of human-computer interaction in augmented reality-based modeling or the formal alignment of models on different abstraction levels – from purely conceptual to models of the physical world – would easily require a multi-year research project. This concerns in particular the implementation of such concepts in modeling tools that go beyond research prototypes but that can be effectively used in practice and teaching. Finally, the exploration of these topics should be conducted in close cooperation with industry to ensure the relevance of such approaches and benefit from the high maturity of commercial implementations in augmented reality.

11.5 Conclusion and Outlook

The combination of conceptual modeling and augmented reality offers for the first time an explicit, formal relation between the real, physical world and that of models in an electronic and machine-processable way. In contrast to prior approaches where models may have been created with pen and paper and attached to physical objects, the technologies of augmented reality permit to go much further. They not only permit to dynamically integrate models, or the information derived from models in the physical space. They also offer new ways for interacting with models or their content.

We plan to investigate these relations in more detail in the future by exploring new scenarios and evaluating them using prototypical implementations. In particular, we aim for the development of new metamodeling approaches that can consider augmented reality in a fundamental way and ease the support of AR-based metamodeling and modeling as well as model-based augmented reality.

References

Abazi, F., Fill, H., Grossmann, W. and Karagiannis, D. (2011). ‘Formalising Knowledge-Intensive Nuclear Fuel Process Models Using Pattern Theory’. In: *Knowledge Science, Engineering and Management - 5th International Conference*. Ed. by H. Xiong and W. B. Lee. Springer, pp. 353–364.

Atkinson, C. and Kühne, T. (2001). ‘The essence of multilevel metamodeling’. In: *International Conference on the Unified Modeling Language*. Springer, pp. 19–33.

Azuma, R. T. (1997). ‘A survey of augmented reality’. In: *Presence: teleoperators & virtual environments* 6.4, pp. 355–385.

Bork, D. and Sinz, E. J. (2013). ‘Bridging the gap from a multi-view modelling method to the design of a multi-view modelling tool’. In: *Enterprise Modelling and Information Systems Architectures (EMISAJ)* 8.2, pp. 25–41.

Bork, D. and Fill, H. (2014). ‘Formal Aspects of Enterprise Modeling Methods: A Comparison Framework’. In: *47th Hawaii International Conference on System Sciences, HICSS 2014, Waikoloa, HI, USA, January 6-9, 2014*. IEEE Computer Society, pp. 3400–3409.

Brambilla, M., Cabot, J. and Wimmer, M. (2017). *Model-Driven Software Engineering in Practice, Second Edition*. Synthesis Lectures on Software Engineering. Morgan & Claypool Publishers.

Chatzopoulos, D., Bermejo, C., Huang, Z. and Hui, P. (2017). ‘Mobile Augmented Reality Survey: From Where We Are to Where We Go’. In: *IEEE Access* 5, pp. 6917–6950. doi: 10.1109/ACCESS.2017.2698164.

Chis-Ratiu, A. and Buchmann, R. A. (2018). 'Design and Implementation of a Diagrammatic Tool for Creating RDF graphs'. In: *Proceedings of the 2nd International Workshop on Practicing Open Enterprise Modelling within OMiLAB (PrOse) co-located with 11th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling (PoEM 2018)*. Ed. by D. Bork, J. Grabis and B. Lantow. Vol. 2238. CEUR Workshop Proceedings. CEUR-WS.org, pp. 37–48.

Cipresso, P., Giglioli, I. A. C., Raya, M. L. A. and Riva, G. (2018). 'The Past, Present, and Future of Virtual and Augmented Reality Research: A Network and Cluster Analysis of the Literature'. In: *Frontiers in Psychology* 9. DOI: 10.3389/fpsyg.2018.02086.

Dörner, R., Broll, W., Grimm, P. and Jung, B. (2019). *Virtual und Augmented Reality (VR/AR) - Grundlagen und Methoden der Virtuellen und Augmentierten Realität*. 2nd ed. Berlin Heidelberg: Springer.

Dumas, M., Rosa, M. L., Mendling, J. and Reijers, H. A. (2018). *Fundamentals of Business Process Management, Second Edition*. Springer.

Ferstl, O. K. and Sinz, E. J. (2015). 'Grundlagen der Wirtschaftsinformatik'. In: *Grundlagen der Wirtschaftsinformatik*. Oldenbourg Wissenschaftsverlag.

Fill, H. (2017). 'SeMFIS: A flexible engineering platform for semantic annotations of conceptual models'. In: *Semantic Web* 8.5, pp. 747–763.

Fill, H. (2018). 'Semantic Annotations of Enterprise Models for Supporting the Evolution of Model-Driven Organizations'. In: *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* 13, 5:1–5:25.

Fill, H., Härer, F., Muff, F. and Curty, S. (2021). 'Towards Augmented Enterprise Models as Low-Code Interfaces to Digital Systems'. In: *Business Modeling and Software Design - 11th International Symposium*. Ed. by B. Shishkov. Vol. 422. LNBIP. Springer, pp. 343–352.

Fill, H. and Johannsen, F. (2016). 'A Knowledge Perspective on Big Data by Joining Enterprise Modeling and Data Analyses'. In: *49th Hawaii International Conference on System Sciences, HICSS*. Ed. by T. X. Bui and R. H. S. Jr. IEEE Computer Society, pp. 4052–4061.

Frank, U. (2014). 'Multilevel Modeling - Toward a New Paradigm of Conceptual Modeling and Information Systems Design'. In: *Bus. Inf. Syst. Eng.* 6.6, pp. 319–337.

Fraser, M. D., Kumar, K. and Vaishnavi, V. K. (1994). 'Strategies for Incorporating Formal Specifications in Software Development'. In: *Commun. ACM* 37.10, pp. 74–86.

Glässner, T. M., Heumann, F., Keßler, L., Härer, F., Steffan, A. and Fill, H. (2017). 'Experiences from the Implementation of a Structured-Entity-Relationship Modeling Method in a Student Project'. In: *Proceedings of the 1st International Workshop on Practicing Open Enterprise Modeling within OMiLAB (PrOse 2017) co-located with 10th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling (PoEM 2017), Leuven, Belgium, November 22, 2017*. Ed. by D. Bork, D. Karagiannis and J. Vanthienen. Vol. 1999. CEUR Workshop Proceedings. CEUR-WS.org. URL: http://ceur-ws.org/Vol-1999/paper4.pdf.

Grambow, G., Hieber, D., Oberhauser, R. and Pogolski, C. (2021). 'A Context and Augmented Reality BPMN and BPMS Extension for Industrial Internet of Things Processes'. In: *Business Process Management Workshops*. Springer, pp. 379–390.

Grum, M. and Gronau, N. (2018). 'Process Modeling Within Augmented Reality - The Bidirectional Interplay of Two Worlds'. In: *Business Modeling and Software Design - 8th International Symposium, BMSD 2018*. Ed. by B. Shishkov. Vol. 319. Springer, pp. 98–115.

Harel, D. and Rumpe, B. (2000). *Modeling languages: Syntax, semantics and all that stuff*. Tech. rep. MCS00-16. The Weizmann Institute of Science, Rehovot, Israel.

Harel, D. and Rumpe, B. (2004). 'Meaningful Modeling: What's the Semantics of 'Semantics'?' In: *Computer* 37.10, pp. 64–72.

Härer, F. and Fill, H. (2020). 'Past Trends and Future Prospects in Conceptual Modeling - A Bibliometric Analysis'. In: *Conceptual Modeling - 39th International Conference*. Ed. by G. Dobbie, U. Frank, G. Kappel, S. W. Liddle and H. C. Mayr. Springer, pp. 34–47.

Havard, V., Baudry, D., Louis, A. and Mazari, B. (Mar. 2015). 'Augmented reality maintenance demonstrator and associated modelling'. In: *2015 IEEE Virtual Reality (VR)*. Arles, Camargue, Provence, France: IEEE, pp. 329–330. doi: 10.1109/VR.2015.7223429.

Herbst, J. and Karagiannis, D. (1998). 'Integrating Machine Learning and Workflow Management to Support Acquisition and Adaptation of Workflow Models'. In: *Ninth International Workshop on Database and Expert Systems Applications, Vienna, Austria, August 24-28, 1998, Proceedings*. Ed. by R. R. Wagner. IEEE Computer Society, pp. 745–752.

Johannsen, F. and Fill, H. (2015). 'Supporting Knowledge Elicitation and Analysis for Business Process Improvement through a Modeling Tool'. In: *Smart Enterprise Engineering: 12. Internationale Tagung Wirtschaftsinformatik*, pp. 752–766.

Karagiannis, D., Buchmann, R. and Walch, M. (2017). 'How can Diagrammatic Conceptual modelling Support Knowledge Management?' In: *25th European Conference on Information Systems, ECIS 2017*. Ed. by I. Ramos, V. Tuunainen and H. Krcmar, p. 101.

Karagiannis, D. and Kühn, H. (2002). 'Metamodelling Platforms'. In: *E-Commerce and Web Technologies, Third International Conference, EC-Web 2002, Aix-en-Provence, France, September 2-6, 2002, Proceedings*. Ed. by K. Bauknecht, A. M. Tjoa and G. Quirchmayr. Vol. 2455. Lecture Notes in Computer Science. Springer, p. 182.

Kelly, S., Lyytinen, K. and Rossi, M. (2013). 'MetaEdit+ A Fully Configurable Multi-User and Multi-Tool CASE and CAME Environment'. In: *Seminal Contributions to Information Systems Engineering, 25 Years of CAiSE*. Ed. by J. A. B. Jr., J. Krogstie, O. Pastor, B. Pernici, C. Rolland and A. Sølvberg. Springer, pp. 109–129.

Leinenbach, S., Seel, C. and Scheer, A. (1999). 'Interaktive Prozessmodellierung in einer Virtual Reality-gestüzten Unternehmungsvisualisierung'. In: *Modellierung 1999, Workshop der Gesellschaft für Informatik*. Ed. by J. Desel, K. Pohl and A. Schürr. Teubner, pp. 11–26.

Messer, B. (1999). 'Zur Interpretation formaler Geschäftsprozess- und Workflow-Modelle'. In: *Wirtschaftsinformatik und Wissenschaftstheorie: Bestandsaufnahme und Perspektiven*. Ed. by J. Becker, W. König, R. Schütte, O. Wendt and S. Zelewski. Springer, pp. 95–124.

Moser, C., Kannengiesser, U. and Elstermann, M. (2022). 'Examining the PASS Approach to Process Modelling for Digitalised Manufacturing Results from Three Industry Case Studies'. In: *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* 17. doi: 10.18417/emisa.17.1.

Muff, F. and Fill, H. (2021a). 'Initial Concepts for Augmented and Virtual Reality-based Enterprise Modeling'. In: *Proceedings of the ER Demos and Posters 2021 co-located with 40th International Conference on Conceptual Modeling (ER 2021), St. John's, NL, Canada, October 18-21, 2021*. Ed. by R. Lukyanenko, B. M. Samuel and A. Sturm. Vol. 2958. CEUR Workshop Proceedings. CEUR-WS.org, pp. 49–54. url: http://ceur-ws.org/Vol-2958/paper9.pdf.

Muff, F. and Fill, H.-G. (2021b). 'Towards Embedding Legal Visualizations in Work Practices by Using Augmented Reality'. In: *Jusletter-IT* 27-Mai-2021.

Muff, F. and Fill, H. (2022a). 'A Framework for Context-Dependent Augmented Reality Applications Using Machine Learning and Ontological Reasoning'. In: *AAAI Spring*

Symposium on Machine Learning and Knowledge Engineering for Hybrid Intelligence. Vol. 3121. CEUR-WS.

Muff, F. and Fill, H. (2022b). 'Use Cases for Augmented Reality Applications in Enterprise Modeling: A Morphological Analysis'. In: *Business Modeling and Software Design - 12th International Symposium, BMSD 2022.* Ed. by B. Shishkov. Vol. 453. Springer, pp. 230–239.

Mylopoulos, J. (1992). 'Conceptual modelling and Telos'. In: *Conceptual Modeling, Databases, and Case: An Integrated View of Information Systems Development.* Ed. by P. Loucopoulos and R. Zicari. John Wiley & Sons, Inc., pp. 49–68.

Osterwalder, A., Pigneur, Y. and Clark, T. (2010). *Business model generation: a handbook for visionaries, game changers, and challengers.* Wiley.

Perez-Castillo, R., Ruiz-Gonzalez, F., Genero, M. and Piattini, M. (2019). 'A systematic mapping study on enterprise architecture mining'. In: *Enterprise Information Systems* 13.5, pp. 675–718.

Russell, B. (1922). 'Introduction'. In: *Tractatus Logico-Philosophicus Logisch-philosophische Abhandlung.* Ed. by L. Wittgenstein. Kegan Paul.

Sandkuhl, K., Fill, H., Hoppenbrouwers, S., Krogstie, J., Leue, A., Matthes, F., Opdahl, A. L., Schwabe, G., Uludag, Ö. and Winter, R. (2016). 'Enterprise Modelling for the Masses - From Elitist Discipline to Common Practice'. In: *The Practice of Enterprise Modeling - 9th IFIP WG 8.1. Working Conference, PoEM 2016, Skövde, Sweden, November 8-10, 2016, Proceedings.* Ed. by J. Horkoff, M. A. Jeusfeld and A. Persson. Vol. 267. Lecture Notes in Business Information Processing. Springer, pp. 225–240.

Sandkuhl, K., Fill, H., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A. L., Schwabe, G., Uludag, Ö. and Winter, R. (2018). 'From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling'. In: *Bus. Inf. Syst. Eng.* 60.1, pp. 69–80.

Seiger, R., Kühn, R., Korzetz, M. and Aßmann, U. (Oct. 2021). 'HoloFlows: modelling of processes for the Internet of Things in mixed reality'. In: *Software and Systems Modeling* 20.5, pp. 1465–1489. DOI: 10.1007/s10270-020-00859-6.

Smajevic, M., Hacks, S. and Bork, D. (2021). 'Using Knowledge Graphs to Detect Enterprise Architecture Smells'. In: *The Practice of Enterprise Modeling - 14th IFIP WG 8.1 Working Conference, PoEM 2021.* Ed. by E. Serral, J. Stirna, J. Ralyté and J. Grabis. Vol. 432. LNBIP. Springer, pp. 48–63.

Stachowiak, H. (1973). *Allgemeine Modelltheorie.* Springer.

Strahringer, S. (1998). 'Ein sprachbasierter Metamodellbegriff und seine Verallgemeinerung durch das Konzept des Metaisierungsprinzips'. In: *Modellierung '98, Proceedings des GI-Workshops in Münster, 11.-13. März 1998.* Ed. by K. Pohl, A. Schürr and G. Vossen. Vol. 9. CEUR Workshop Proceedings. CEUR-WS.org.

Thalheim, B. (2012a). 'Syntax, Semantics and Pragmatics of Conceptual Modelling'. In: *Natural Language Processing and Information Systems - 17th International Conference on Applications of Natural Language to Information Systems.* Springer, pp. 1–10.

Thalheim, B. (2012b). 'The Science and Art of Conceptual Modelling'. In: *Trans. Large Scale Data Knowl. Centered Syst.* 6, pp. 76–105.

Van Der Aalst, W. (2012). 'Process mining'. In: *Communications of the ACM* 55.8, pp. 76–83.

Van Woensel, W. and Abidi, S. S. R. (May 2019). 'Benchmarking semantic reasoning on mobile platforms: Towards optimization using OWL2 RL'. In: *Semantic Web* 10.4, pp. 637–663. DOI: 10.3233/SW-180315.

Yu, E. S. K., Giorgini, P., Maiden, N. A. M. and Mylopoulos, J., eds. (2011). *Social Modeling for Requirements Engineering*. Cooperative information systems. MIT Press.

Zachman, J. A. (1987). 'A framework for information systems architecture'. In: *IBM Systems Journal* 26.3, pp. 276–292. doi: `10.1147/sj.263.0276`.

Zhou, F., Duh, H. B.-L. and Billinghurst, M. (2008). 'Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR'. In: *International Symposium on Mixed and Augmented Reality*. IEEE, pp. 193–202.

Chapter 12

Towards Tailored Support for Design Practices in Collaborative Modeling Sessions

Tobias Kautz and Robert Winter

Especially in early phases of digital business innovation endeavors, collaborative design sessions are critical for creating a shared understanding, aligning diverse perspectives and objectives, and making fundamental design decisions. As knowledge about the characteristics of these sessions is scarce, existing support artefacts (e. g., methods, techniques, or software tools) cannot be easily selected, let alone tailored to the specific characteristics of such a session resulting in only generic guidance and missed effectivity potentials. This paper reports intermediate results of a design science research project aimed at developing tailored support artefacts for such sessions. As a foundation to achieve higher effectivity, a theory-informed conceptual model is proposed from which the classification dimensions for collaborative design sessions are derived. Classification dimensions focus on involved stakeholders, topics, i. e., compositions of focal entity types, and information characteristics derived from abstraction principles. Based on interviews and surveys, fifteen session types along four phases are identified. A concrete plan for future research is provided to investigate the explored session types and develop tailored support artefacts for them.

12.1 Introduction

For years and most recently intensified through the Covid-19 pandemic, organizations are designing and implementing, sometimes voluntarily, sometimes not, digital technologies. These change endeavors impact, e. g., customer behaviors, value propositions, value chains, and sometimes entire business models (Dąbrowska et al. 2022). On an organizational level, we will refer to them as *digital business innovation* (DBI) endeavors. DBI endeavors can be focused on a specific business unit (e. g., a certain component of the product and service portfolio), but also across certain business units or ultimately on the entire enterprise. The latter type of DBI endeavors is also known as digital (enterprise) transformation. Depending on its focus and also the specific wording used in the respective enterprise, a DBI endeavor can be organized as a project, a program or even a strategic initiative (Klingebiel and De Meyer 2013; Pellegrinelli 2011). However, even when implemented in one business unit only, the value proposition and IT landscape changing character has consequences for the entire enterprise. Even the introduction of a single new service may compete with offerings of other business units or at least influence the expectations of their customers. In a similar way, the implementation of isolated new IT-related entities can have an impact on enterprise-wide security and compliance. However, local changes cannot only have negative, but also desirable side effects: experiences in building up new capabilities – or even the capabilities themselves – can be leveraged for the rest of the organization, financial benefits strengthen the entire organization, or new types of customers and employees can be attracted. Against this background, we analyze DBI always at the enterprise-level – even if the underlying change focus is local.

Adopting the 'Strategy-as-Practice' view from management research, which focuses on *'the actual day-to-day activities, contexts, processes and content [of strategy development work] that relate to strategic outcomes'* (Peppard et al. 2014, p. 1) and which is also seen as beneficial in the information systems (IS) field (Whittington 2014), activities (on various aggregation levels, i. e., from micro-level behavior and interactions to overall processes) are central to DBI endeavors. In these activities, people use a variety of artefacts such as methods, techniques, or software – which we will refer to for reasons of simplicity subsumed as 'support artefacts' in the following.

Prominent examples might be enterprise architecture development *methods* such as TOGAF® (`https://www.opengroup.org/togaf`), canvas-based ideation *techniques* or graphical editing *software* such as PowerPoint. Adopting the view of enterprises as systems (Lukyanenko et al. 2022; Rouse 2005), one important source of those support artefacts comes from the domain of enterprise modeling (EM) which supports the analysis and design of those systems (Sandkuhl et al. 2014) and can also occur in the form of a more informal 'grass-roots-modeling' (Sandkuhl et al. 2018, p. 69).

Noteworthy examples for more 'lightweight' modeling approaches in the context of strategy and digital transformation design are: *Fractal Enterprise Modeling* (e. g., Bider et al. 2021) establishing a connection between an enterprise's internal structure and environmental influences; de Kinderen et al. (2021)'s modelling method for facilitating strategic analysis; Bergmann and Strecker (2018)'s modeling method for supporting strategic planning; the *LiteStrat Modelling Method* (Pastor et al. 2022) with an emphasis on the hierarchy of organizational goals and associated concepts; *4EM* (Sandkuhl et al. 2014) as a comprehensive EM method which also has been specifically used for capability modeling (Koutsopoulos 2021; Stirna and Sandkuhl 2018); the *Work System Modeling Method* (Bork and Alter 2020) for product/service delivery aspects; the *Digital Transformation-Oriented Model* (Hafsi and Assar 2021) which focuses on mapping flexibility-enhancing entities in the context of digital transformation; and the *Process Goal Alignment Technique* (Roelens 2022) aimed at enhancing alignment between strategy and business processes.

In practice, support artefacts are developed by software vendors, consultants, or organization-internal coaches/centers of excellence. Strongly generalized, it can be said that all of these want to provide effective and efficient design support artefacts to users, in order to, ultimately, e. g., generate revenue with selling their support artefacts (for artefact developers outside the organizations) or increase endeavor performance (with regards to the internal ones).

To do so, they need to understand concretely in which situations users (could) leverage modeling support artefacts, i. e., understand modeling practitioners regarding *'when they model, what the role of modeling artifacts really is, how several actors collaborate in modeling or using models'* (Sandkuhl et al. 2018, p. 73). In other words, they need to understand the (possibly grouped) activities occurring in DBI endeavors.

12.2 Research aims and questions

In our *overall research*, we plan to tailor existing or develop new modeling support for the design phase in DBI endeavors, i. e., using models to approach, conceptually develop, and document the most promising future state of the organization (see also Section 12.6). *Tailoring* in this paper should mean potentially adapting and/or combining existing support artefacts based on a more differentiated understanding on the requirements they should fulfill. In context of this paper, tailoring is only investigated with regards to *cross-organizational* requirements. Thus, in a first step, we aim to create a foundation to structure the *existing* supply-side of modeling-based support artefacts based on the requirements these artefacts fulfill. For the 'demand-' and requirement-side we use the lens of 'collaborative session' types (CSTs) to analyze (part of) the activities conducted in DBI endeavors. In theoretical terms, these sessions can be seen as episodes (Hendry and Seidl 2003) in the context of the management of DBI endeavors in which several persons from *different* domains (e. g., corporate functions) come together (thus, 'collaborative') to make decisions on, design, plan, and communicate the endeavor (e. g., meetings, workshops). Following from this understanding, we exclude adjacent sessions (e. g., to collect the information necessary for the discussions in the sessions, such as customer interviews), sessions on the 'meta-level' (e. g., sessions setting up the general DBI process with its roles and applicable methods), and collaborative sessions in later stages of DBI (e. g., detailed design during implementation). Thus, our first research question *for this paper* is:

RQ1: Which dimensions allow to describe collaborative sessions to identify design support artefacts that are suitable for the respective session on a cross-organizational level?

Dimensions should indicate that for this paper, we primarily consider the different aspects of collaborative sessions that are relevant, while we only provide partial and preliminary suggestions for characteristics (i. e., expressions, manifestation possibilities) 'underneath' those dimensions (as outlined in Section 12.5.2). *Cross-organizational* should emphasize that we are interested in identifying design support artefacts that provide the functionality to solve a certain design problem, deliberately neglecting organization-related dimensions at this stage that might create, e. g., cost-benefit-tradeoffs. In other words, we are only interested in the different 'nature' of design problems but neglect organization-specific contingency factors further 'shaping' those problems (Bucher et al. 2007). We elaborate on this aspect in Section 12.3.1. Second, to provide an overview of the sessions for which support is needed, we ask:

RQ2: Which types of collaborative sessions occur on the management-level of DBI endeavors?

Based on the previous section, we define DBI endeavors as *purposeful, structured, and steered activities bringing about an organizational change that holistically, in the case of digital (enterprise) transformations also fundamentally, alters the social and technical constituting entities on enterprise level, especially the value proposition, and that relies on the alignment of digital technologies, business/organizational designs, and human ambitions and perceptions* (Dąbrowska et al. 2022; Demirkan et al. 2016; Hartl and Hess 2017; Hinings et al. 2018; Li et al. 2018; Markus 2004; Matt et al. 2015; Project Management Institute 2008). As a consequence, we expect that all DBI design and implementation activities follow a similar course (i. e., structure), even if the digital (enterprise) transformations are much more complex and require longer time spans. Based on a similar logic, we exclude DBIs that are not implemented in existing organizations (instead, e. g., in a spin-off) and/or where realization is accomplished via a joint venture or acquisition, expecting that in these cases, especially at a later stage, CSTs with other topics and stakeholders involved (i. e., other CSTs) occur. Examples for endeavors that we do not consider under the definition of DBI are the routine development of new products (having no or only a slight impact on organizational designs as they for example leverage existing processes and skills) or process innovations (as long as they do not have a significant impact on the value proposition). In line with other researchers (Hund et al. 2021), our understanding of a digital innovation does not differentiate whether the innovation is just novel for the respective organization or for the industry as such, recognizing and consciously aiming at extending the scope of phenomena relevant for the research. For the overall research project, we follow the design science research (DSR) approach by Peffers et al. (2007), where we currently are in the phase of problem identification. In the remainder of the paper, the conceptual foundations and related literature for the research problem are presented, before our research method is described in more detail. Explorative results are presented and discussed, including a plan for further research.

12.3 Literature review

12.3.1 Conceptual foundations

To approach the identification and development of (modeling-based) support artefacts for design activities in DBI endeavors from a DSR perspective, we developed a theory-informed conceptual model illustrated in Figure 12.1. It is intended to justify and support data collection as well as identify white spots regarding, e. g., activity-procedure combinations for which no solution knowledge exists. In the following, we explain how the different areas (i. e., frames) and concepts (i. e., the individual boxes) were derived. The model also draws on the considerations made in Kautz and Winter (2023).

To set the 'boundaries' of our conceptual model, i. e., its scope of coverage, we chose to use an overall separation in the *problem and solution space* common in DSR (vom Brocke et al. 2020). Within the problem space, *situational artefact design* (Winter 2011) was chosen as a further structuring dimension because it provides a connection from a neutral phenomenon description (as a general starting point for every research) to a problematization and *situation-dependent* solution approach. To conceptualize different situations, we use the term 'endeavor types'. The endeavor we defined in the introduction can be conceived as belonging to a different type of endeavor compared to, e. g., a more incrementally oriented process optimization. The types are a result of different *characteristics* of these endeavors, one exemplary being whether the value proposition of an organization is changed or not. As a consequence, organizations face different *design problems* based on those characteristics, in which organizations want to come from an as-is-state to the targeted state (Hevner et al. 2004), e. g., how the enterprise is configured before the value proposition change and afterwards. As pointed out above, activities are a central aspect in our understanding of the research problem, and a focus on those is explicitly demanded in the EM literature (Sandkuhl et al. 2018), following the user centricity paradigm already common in other disciplines (Brenner et al. 2014). Activities can be conceived as occurring on various levels until a not further decomposable one, denoted by the round arrow in the upper right corner (Gutzwiller 1994). In reaction to the design problem, an organization could have developed

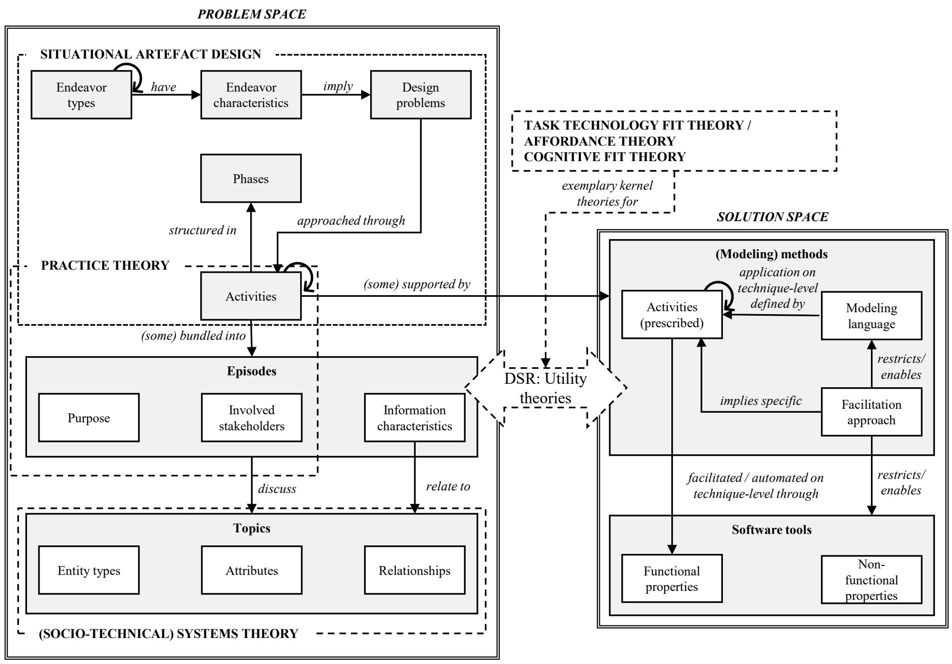


Figure 12.1: Conceptual model to prepare tailoring of modeling support for collaborative sessions in DBI endeavors

an own, not public set of activities (maybe explicitly called and documented as a method). It could also have adopted a method from a consultancy or another reference method base. The challenge here is that the actual, observable activities conducted are influenced by the existing support artefacts in the solution space. However, to avoid complexity, we have not separated between an existing and potential solution space in our model.

Activities are also the focus of practice theory (Feldman and Orlikowski 2011). As the analytical lens within practice theory, we use episodes, defined as a *'sequence of communications structured in terms of its beginning and ending'* (Hendry and Seidl 2003, p. 176), characterized by conscious goal- and time-restrictions, their spontaneous or routine occurrence, a change from the pre- and succeeding discursive structures, and being a means to reflectively facilitate strategic change (Hendry and Seidl 2003). This definition appeared to fit well to our research phenomenon and is also a common term in the literature (Section 12.3.2). The *purpose* and *involved stakeholders* are also based on Hendry and Seidl (2003). Other constituting elements of episodes proposed by these authors (e. g., location), however, were not integrated into the model as the above mentioned are expected to have a greater impact on the selection of (modeling) methods and techniques, while they might be important for the selection of software tools. Based on abstraction mechanisms for modeling, we derived *information characteristics* to describe how topics are discussed in the sessions (Section 12.5.2). It should be noted that the exact characteristics will most likely vary across organizations. For example, other types of stakeholders might be participating, or topics will be discussed with a different intensity. However, we assume that, on the one hand, there are some characteristics of the episodes and topics that follow logically from the activity. On the other hand, we expect to find at least groups of similar organizations or even a majority of organizations having episodes with the same characteristics. Not achieving total uniformity appears to be acceptable given the aim of finding session *types*. As stakeholders in these sessions work on the future state of the enterprise, a conceptualization is needed what enterprises actually are. For this, we adopted the view common in the EM literature that enterprises can be conceived as (*socio-technical*) *systems* (Lukyanenko et al. 2022; Rouse 2005). A *topic* then can be seen – similar to a viewpoint (Steen et al. 2004) – as a description of parts of the system for specific stakeholders for a specific purpose (i. e., set of concerns). The *entity types*, *attributes*, and *relationships* were included in the conceptual model as they are central modeling constructs and aspects to describe a system (Lukyanenko et al. 2022).

The solution space is first comprised of *(modeling) methods*. Our understanding of the constituting elements of a method is based on (Gutzwiller 1994) and comprises – besides the activities – roles, an information model, results, and techniques, where the latter support creation of the results. However, for reasons of simplicity, we have chosen to show only the activities in the model as they provide the necessary link to the problem space. Our definition of a modeling technique follows Karagiannis and Kühn 2002. While the *modeling language* is shown separately, the *modeling procedure* is included in the prescribed activities. We have chosen this simplification as, again, the already applied techniques might be observable as (very fine-grained) activities in the problem space. *Software tools* facilitate or automate activities on technique-level through mechanisms and algorithms (Karagiannis and Kühn 2002). It should be noted that an important previous effort to research the relationship between modeling methods and supporting software tools was done early in EM research (Stirna 2001). In this work, situational factors such as EM method usage maturity & stability, tool usage & development maturity, organizational commitment as well as factors relating to the complexity and resources of the EM project, as well as intentional aspects such as consultant support, the purpose of EM, attitude towards model maintenance, and stability of intentions were identified. While these factors have an important influence on the choice for a specific software tool in a concrete organization, this has no impact on which *functional properties* a certain software tool (or its respective type) provides to support a specific design activity and also no impact on the non-functional properties it offers. With respect to our aim of preparing tailoring of support artefacts, this also implies that these artefacts still might need further adaptions to be used in an organization.

Modeling approach recognizes the important distinction between participatory and conventional forms of modeling (Sandkuhl and Seigerroth 2020). Participatory modeling means that a facilitating modeling expert supports content-providing domain experts with the formally correct creation of the model, while the domain experts are actually developing the model (Stirna and Persson 2018), whereas conventional modeling implies that domain experts, if at all, only share their knowledge without being involved in the model development activity itself (Sandkuhl and Seigerroth 2020). The collaboration of modeling experts when developing the model also falls under this category. When thinking about lightweight modeling approaches, this might cause that domain and modeling experts are the same, if the modeling approach is sufficiently easy to be used efficiently and effectively without expertise. Separating between different modeling approaches will most likely have an impact on the suitable support artefacts as they allow for more advanced (i. e., formal, complex) artefacts to be used if a modeling expert for these artefacts is involved. That means, the creation of models can become more difficult, while there might still be limits to the artefact used in terms of understandability of the developed model, i. e., its notation (Gutschmidt et al. 2022).

Also common in the DS literature, the 'bridge' between the problem and solution space are utility theories (Venable 2006), which are the ultimate aim of our research as defined in Section 12.2 above. The utility can be justified by *kernel theories* (Gregor and Jones 2007). For these, we identified three well-known and broadly applied mid-range theories (Gregor 2006) that provide fundamental prescriptions for achieving alignment between (and also within for the latter) design activities and supporting artefacts. Of course, this list should not be considered to be conclusive. The first two concern the alignment between activities (i. e., tasks) with (modeling) techniques and software tools that offer corresponding functional and non-functional properties and are the *task-technology-fit theory* (Goodhue and Thompson 1995) and affordance theory (Maier and Fadel 2009). The first provides a quite comprehensive overview about the areas (i. e., characteristics of the task, technology, and individual, influences on utilization beliefs, as well as feedback) to be considered when considering performance impacts of deployed technology (in this case, the support artefacts) (Goodhue and Thompson 1995). The latter defines affordances as *'possibilities for goal-oriented action afforded to specified user groups by technical objects'* (Markus 2004, p. 622). Among other differences and advantages to solely investigation functions only, affordances allow to describe how well an artifact affords a specific goal-oriented use (Maier and Fadel 2009), which can for example be used to compare simple graphical editors and sophisticated modeling software with regards to how well they enable creating custom notations, although they do not share the same functions. The third one regards the alignment between the 'nature' of the topics (i. e., information characteristics) and their graphical representation (i. e., through the modeling technique) inspired by the *cognitive fit theory*, which, in brief, postulates that if a problem representation (in this case, a model or software tool) emphasizes the same type of information as relevant for solving a task, this will increase performance for solving the task (Vessey and Galletta 1991). It has been often applied in the context of (conceptual) modeling (e. g., Bera et al. 2019; Malinova and Mendling 2021). The dimensions of the classification of CSTs are given by the entities *involved stakeholders*, *information characteristics*, and *entity types*. Based on a developed long-list of possible dimensions, we have selected these ones as they both allow cross-organizational classification of sessions/ session types and can be attributed with characteristics that are defined non-ambiguously (see Section 12.5.2 for more details). While the conceptual entities in the model are conceived to be comprehensive (enough), many more relationships could be shown in the model. However, we intended to show one central path from the phenomenon to the proposed solution artefact types. Moreover, we decided to use a bold-bidirectional arrow to show alignment, which covers multiple relationships of the individual boxes.

12.3.2 Prior research

Modeling in collaborative sessions is well studied in the EM field (Stirna and Persson 2018), which speaks in favor of the relevance of the research presented in this paper. Experiences are shared (Renger et al. 2008; Stirna et al. 2007) and guidance is proposed on when to choose participatory or conventional modeling (Sandkuhl and Seigerroth 2020). Research also addresses various aspects of the actual content of such sessions, e. g., how they should be structured (Fellmann et al. 2020), how distributed real-time collaboration can be conducted over the web (Nicolaescu et al. 2016), how multi-touch tables can be used (Gutschmidt

2019). Studies also cover concrete conversations (Hoppenbrouwers and Wilmont 2010), psychological issues (Gutschmidt 2022; Gutschmidt and Richter 2021), or how the quality of collaborative modeling can be assessed (Ssebuggwawo et al. 2010). While the literature just outlined only provides limited insights at this stage of the research project (i. e., the identification of CSTs), it will be highly useful phenomena and solution design knowledge (Drechsler and Hevner 2018) for developing the support artefacts later (see Section 12.6). Quite recently, van Gils et al. (2022) propose to do research on 'enterprise design dialogues' (van Gils et al. 2022, p. 4), i. e., the concrete discussions in which the design of the enterprise is taking place and in which models are used, in order to ultimately address the challenges of the diversity of stakeholders involved in an enterprise transformation and to include non-experts in modeling. We also leveraged the search results from another, comprehensive search for DBI modeling methods in the EM field (Kautz and Winter 2023) (see search strings published under `https://doi.org/10.5281/zenodo.7318057`) and identified two papers that were dealing with groupings of activities in DBI endeavors similar to the ones in our focus: Wißotzki and Sandkuhl (2017) provide a procedure model, the so called 'Digital Innovation and Transformation Process' (p. 352), which seems to be consistent with the one we derived. However, they do not identify concrete collaborative sessions along this procedure model. Stirna and Persson (2018) (based on their earlier work in Persson and Stirna 2001) and Sandkuhl et al. (2014) identify some generic business challenges which EM is expected to support, however these have not been derived empirically and/or are not focused on DBI endeavors.

To identify prior research on collaborative sessions from the general IS and management field, we searched the titles and abstracts in the Web of Science™ and the AIS eLibrary with combinations of the terms *transformation/strategy/innovation* and *episodes* (limiting the Web of Science™ Categories on business and information systems related ones due to the high number of search results) and performed a search for articles citing (Peppard et al. 2014) and (Whittington 2014). We identified the following articles: (Chanias et al. 2019) investigate the development of digital transformation strategies at a pre-digital financial service provider, but they provide only a description of the episodes encountered (i. e., no detailed analysis), which is also limited to one single organization. However, the examples for collaborative sessions mentioned in the case description appear to be consistent with our CSTs. Berghaus and Back (2017) looked at the fuzzy front end of digital transformation, but although adopting a practice perspective, focused on activity systems instead of single episodes. Morton et al. (2016) look at three different types of episodes in the context of open strategy initiatives, which are quite different from the scope of our research. To conclude, there is already some research dealing with similar phenomena like the collaborative sessions in our research, the closest being Chanias et al. (2019) and Wißotzki and Sandkuhl (2017). However, these do not fulfill the requirements of being *observed* (rather than prescribed) in *multiple* organizations. Most importantly, however, they do not provide necessary details on the attributes relevant to characterize collaborative sessions shown in the model above.

We have also found a quite often cited classification of design activities in the context of engineering design (Ostergaard and Summers 2009). However, their classification appears to be more focused on enhancing the performance of design teams through identifying challenges and support needs on the 'soft' aspects of collaborative design (e. g., interpersonal relations), than on content-related aspects of the design task. In other words, their focus seems to be on efficiency- rather than effectivity-related aspects. Thus, we have chosen to derive and validate a set of CSTs using a process outlined in the next section.

12.4 Method

12.4.1 Literature-based derivation

As indicated in the conceptual model, to derive the CSTs, an indication about the logical (and temporal) sequence of the activities is needed. Process models for digital transformation, which is close to our understanding of DBI, are considered to be appropriate to provide this necessary structure. Searching Web of Science™ articles that contain process or phase models for digital transformation, we opt for the consolidated model of stages and activities of Ubiparipovic et al. (2022), because it is based on a literature review of the anyhow scarce literature specific for this topic and consistent with the other relevant sources we identified (Klos et al. 2021; Rummel et al. 2022; Wessel et al. 2021). This model was then supplemented on sub-activity level with the activities presented in (Labusch et al. 2013). We chose this set of activities as it was derived empirically (e. g., as opposed to prescriptive methods), formulated with identifying (enterprise architecture) modeling support in mind, but still validated and extended by another comprehensive set of activities empirically derived by Labusch and Winter (2013) as well as underpinned by dynamic capabilities theory. Overall, this implied a set of activities that appeared balanced and documented at the right level of detail. After sorting the activities into the phases based on an information input and output logic, another grouping logic *within* the phases was needed. For that, we have chosen thematical similarity of the information needs satisfied in the sessions. Ex- ante, the nature of models as a dedicated form of displaying information as well as the emphasis on 'information flows' and 'information demand' (Sandkuhl et al. 2018, p. 73) in the call for more practice-oriented research on EM was the rationale for this choice. Ex-post, i. e., after the derivation, our ability to derive information characteristics and involved stakeholders justified this decision. Overall, this seemed sufficient for a discussion with practitioners, presented in the next section.

12.4.2 Empirical analysis

As a pre-test to a survey, four interviews lasting about half an hour were conducted with practitioners (an enterprise architect, a head of corporate IT, a head of IT & Digital, and a former executive assistant) involved in the management of DBI endeavors. The interviews were recorded. The preparatory presentation including the upfront sent questionnaire as well as more information about the interview participants can be found at `https://doi.org/10.5281/zenodo.7317907` and `https://doi.org/10.5281/zenodo.7317921`, respectively. In general, they assessed the list of CSTs to be quite comprehensive, only one interviewee proposed supplementing 'team-level communication'. Further remarks on the description of the CSTs for the survey were related to renaming of ambiguous words, clarification of the research scope, supplementation of job titles, simplifications, provisioning of examples, and a bilingual setup of the survey. A survey, accessible at `https://ww2.unipark.de/uc/survey_anonymized/`, was then distributed mainly through the professional social networks LinkedIn and Xing and complementary via e-mail. In total, we received six responses for this pilot survey round. We attribute this low number of responses to the length of the questionnaire, which lasted minimum ten minutes and included a lot of text. Except for one participant who was not confident with placing the 'rough concept development' before the 'business case development' (see Figure 12.2 in the next section), all survey participants as well indicated the general relevance of the CSTs. Moreover, the participants confirmed the relevance of the CSTs individually, besides three CSTs for which no responses have been received. Suggestions included supplementing a ‘strategy alignment session’ and a ‘communication to personnel representatives’. The first suggestion was additionally explained in the free text field and thus comprehensible, and accordingly added. The second one was just named, and it was not clear how this was different from including personnel representatives in the activities before implementation and the ‘townhall-style communication’. Thus, we have chosen not to include it. To triangulate the small data set of our first survey, we decided to conduct a second, non-anonymous survey ($n = 18$) in context of an executive MBA program that is specialized on business innovation and digital transformation. Participants’ backgrounds differ with regard to management level, corporate function, size of organization, and industry (see `https://doi.org/10.5281/zenodo.7317949`), but all participants have an academic background and at least five years of management experience. In their roles they are both specifically responsible for DBIs or steering them in a managerial role. Thus, we consider them to be a (first) representative sample for validation. For each CST, at least one executive has already been involved, speaking for the general relevance of the CSTs.

12.5 First results

12.5.1 Collaborative session types

Given that nearly all consulted persons indicated the relevance of the CSTs, we include an overview of those along DBI phases in Figure 12.2 below. Additional descriptions as well as results can be found in the survey at `https://ww2.unipark.de/uc/survey_anonymized/` and survey results (`https://doi.org/10.5281/zenodo.7317956`), respectively. The circles in Figure 12.2 should indicate that CSTs can occur multiple times with back-and-forth iterations (also between phases), while the implementation/scaling-up occurs after a rather ‘hard’ decision to do so.

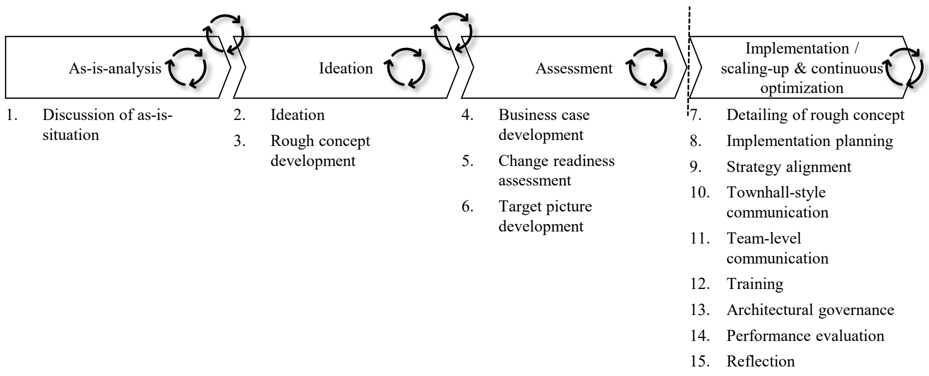

As-is-analysis
Ideation
Assessment
Implementation / scaling-up & continuous optimization
1. Discussion of as-is-situation
2. Ideation
3. Rough concept development
4. Business case development
5. Change readiness assessment
6. Target picture development
7. Detailing of rough concept
8. Implementation planning
9. Strategy alignment
10. Townhall-style communication
11. Team-level communication
12. Training
13. Architectural governance
14. Performance evaluation
15. Reflection

Figure 12.2: Types of collaborative sessions in DBI endeavors

12.5.2 Dimensions and characteristics to classify CSTs

To identify candidates for possible dimensions and characteristics that can be used to classify collaborative sessions, we used existing constituting elements of collaborative sessions (see Section 12.3.1) as well as the generic interrogatives (e. g., who, what, how), similar to, e. g., Yoshioka et al. (2001), as a basis. The long list and its assessment based on evaluation criteria for appropriate classification dimensions can be found under `https://doi.org/10.5281/zenodo.7759577` (criteria adapted from Sneath and Sokal 1973). Our final selection of dimensions (e. g., the ones we expect to have similar characteristics across organizations (regarding a specific session type) and that can be defined unambiguously) is presented in the subsequent paragraphs.

Involved stakeholders were derived based on common business functions (e. g., strategy, marketing). Based on feedback from the conducted pre-tests, we also supplied exemplary job titles for these roles based on an internet search (on websites such as `https://www.theladders.com/job-titles/process-management`). The involved stakeholders are likely to be relevant for tailoring modeling support tools as they have a different experience and familiarity with modeling and specific modeling languages/visualization options, which has for example been found to influence the attitude towards a certain modeling language (Recker 2017).

Exemplary entity types were derived on the basis of the literature we already used to define the CSTs (Labusch et al. 2013). For reasons of simplicity, they were presented in the survey as bundles, i. e., topics (see Section 12.3.1) with only some illustrative entity types given for each topic. The need to consider these to tailoring support artefacts is quite obvious, as the addressed entity types are a core feature of a modeling language.

Information characteristics are based on fundamental abstraction mechanisms and modeling constructs (Lukyanenko et al. 2022; Mylopoulos 1998; Proper and Guizzardi 2021; Roth et al. 2014) and summarized in Table 12.1.

The first information characteristic, *completeness of entities discussed* (*of a specific type*) considers whether it is relevant for a session to identify ‘all’ the entities of a certain type (e. g., competitors, processes, IT systems) or whether focusing on the essentials/‘game

Information characteristics	*Based on ...*
Completeness of entities discussed (of a specific type)	Selection mechanism
Completeness of relationships between the entities discussed	Selection mechanism, relationship construct
Level of granulatiry for discussing the entities	Aggregation mechanism
Heterogeneity of entity types relevant for the session	Selection and classification mechanisms
Importance of a characterization of the entities discussed	Attribute construct
Importance of thinking about alternatives/variants	*Notion of divergent/convergent thinking*

Table 12.1: Information characteristics to classify collaborative sessions where abstraction mechanisms and modeling constructs originate from Mylopoulos (1998) and Lukyanenko et al. (2022), respectively

changing' entities is enough. Analogously, *completeness of relationships between the entities discussed* asks the same for relationships between entities: Is it important to identify 'all' of the relationships or only the essential ones? *Level of granularity for discussing the entities* refers to whether a rather aggregated or more detailed perspective is taken, for example if processes or their respective steps are in focus, respectively. *Heterogeneity of the entity types relevant for the session* characterizes whether the entity types of relevance rather come from a similar domain (e. g., IT with IT systems and data objects) or whether they come from more heterogeneous domains (e. g., competitors and products from the strategy domain and the just mentioned ones for IT are considered in the same session). *Importance of a characterization of the entities discussed* asks whether entities need to be characterized precisely with their attributes (e. g., size, strength, etc. of a competitor) or whether having the entities named at all is enough. Last, the *importance of thinking about alternatives/ variants* considers whether alternatives/versions of a model are needed or one version is enough. A graphical illustration of the information characteristics can be found in the survey under the questions for the respective session type. Given that different modeling languages afford/emphasize different information characteristics, this dimension should also be relevant within a classification used to prepare tailoring of support artefacts.

12.6 Discussion and conclusions

Clearly, the above presented explorative results have their limitations. The conceptual model presented in Section 12.3.1 has – besides discussion in the research team and the feedback from the conference reviewers – not undergone additional (formative and summative) evaluation yet. Having established this basis, most of the potential for research lays in further investigating the collaborative sessions and tailoring support artefacts to them. Concerning those, the biggest limitations of this paper were the limited empirical validation of the sessions as well as providing only preliminary and incomplete characteristics for the classification of collaborative sessions. Thus, to overcome these limitations, Table 12.2 provides an overview about the proposed next research activities and positions the expected contribution of the results into DSR knowledge types (Drechsler and Hevner 2018).

For each activity, we now outline success factors and proposed means to realize them. Concerning (1) *validation of CSTs*, it will be key to increase the sample size, ensured through personalized requests to participate in the research and to people who can recommend lower-hierarchy employees to participate. Also providing enough time and space for asking more 'advanced' questions (e. g., to the information-characteristics) should be enabled through an appropriate length (for the interviews) and inclusion of an explanatory video (for the survey). Besides an extension/revision of the proposed characteristics in general, we see development potential also in the 'slicing' of the sessions. While they should be mutually exclusive and collectively exhaustive (under the defined scope), one could investigate what is the appropriate number of session types (to still be useful) and whether a further hierarchization is needed. This could also be used to ensure that the CSTs are homogenous with regards to the information characteristics. Regarding (2) *elaboration of activities*, several success factors were identified: First, finding the right level of detail and comprehensiveness ('breadth and depth') is important, through early testing for usefulness in light of the overall research aims. Second, identification of *generalizable* activities is key, achieved through observations in multiple organizations and triangulation with existing

Research activity	**Purpose**	**Data collection techniques**	**Deliverables' expected contribution to knowledge type**		
			Propositional (Ω)	**Prescriptive (λ)**	
			Phenomena knowledge	***Solution design knowledge***	***Solution design entities***
(1) Validation of CSTs	Develop point of reference, attempt generalizability	Interviews, focus group, targeted survey	*Cataloging* and *classification* of CSTs	High-level-*requirements* to support CSTs	-
(2) Elaboration of activities	Enable artefact design, establish connection to non-collaborative activities	Observations, desk research / literature review (for triangulation)	*Observations* of, e.g., design problems, activities and their interrelations, challenges, applied support artefacts	Observation protocol (i.e., contributing to *research methods*), more detailed (esp. functional) *requirements*	Overview of (prioritized) existing *models* and *methods* for DBI
(3) Review of existing support artefacts	Identification of potential gaps (i.e., no support existing, new artefact needed)	Desk research / literature review, coverage / capability analysis	*Classification* of existing support artefacts, *cataloging* of gaps (incl. focal ones for our research)	-	-
(4) Collaboration with consulting roles	Filling potential gaps with artefacts on method-to-technique-level	Action design research (e.g., interviews, observations)	-	-	(Modeling) *methods*, meta-*models* to fill potential gaps
(5) Collaboration with software developers	Filling potential gaps with artefacts on technique-to-software-level	System development (e.g., requirements gathering, demonstrations)	-	Most detailed *requirements*, design *principles* and necessary *features*	Software *system* (e.g., a business-user-oriented modeling software)

Table 12.2: Proposed research plan to further study modeling support for collaborative sessions in DBI endeavors, where knowledge types are based on Drechsler and Hevner (2018)

methods from the solution space (e. g., from leading consultancies). Third, the handling of data gaps will be crucial as existing consulting knowledge might not be freely accessible, which should be remedied through approaching consultancies directly and making data gaps explicit. Fourth, regarding methods coming from the academic literature, a scoping is needed, that can be done, for example, based on quality of the outlet. Last, terminological differences within and between knowledge from practice and science (i. e., mean the same and use different words) should be addressed by an own meta-model/ontology as kind of a 'dictionary'. With regards to (3) *review of existing support artefacts*, identifying the most pressing gaps needs to be accomplished based on impact (e. g., using the criteria summarized in Rosemann and Vessey (2008) and capability considerations. In (4) *collaboration with consulting roles*, the general drawbacks of action design research need to be remedied based on the existing methodology literature. Last, for (5) *collaboration with software developers*, it will be essential to maintain the relationships between DBI-specific method aspects and the technical implementation (i. e., not getting too domain-unspecific) through appropriate and regular communication in a potential software development project.

Based on the argumentation in the sections above, the tailoring enabled through a more differentiated problem understanding can be expected to enable the design of support artefacts, that are more effective and efficient and thus bring benefits for researchers and practitioners. Moreover, through a step towards a common language, analyzing existing research results for each session should be better facilitated. Last, the CSTs should enable researchers to understand existing sessions more easily they encounter in an organization, i. e., putting them into different 'buckets'.

Acknowledgment

We thank four anonymous conference reviewers for their fruitful comments on an earlier version of this article and the participants of the 1st European Workshop on Research for Impact in Digital Innovation for their feedback on the CSTs.

References

Bera, P., Soffer, P. and Parsons, J. (2019). 'Using Eye Tracking to Expose Cognitive Processes in Understanding Conceptual Models'. In: *MIS Quarterly*. DOI: 10.25300/MISQ/2019/14163.

Berghaus, S. and Back, A. (2017). 'Disentangling the Fuzzy Front End of Digital Transformation: Activities and Approaches'. In: *International Conference on Information Systems, Seoul, South Korea*. URL: https://aisel.aisnet.org/icis2017/PracticeOriented/Presentations/4.

Bergmann, A. and Strecker, S. (2018). 'A Modeling Method in Support of Strategic Planning: Language Design and Application'. In: *International Conference on Information Systems, San Francisco, California, USA*. URL: https://aisel.aisnet.org/icis2018/modeling/Presentations/8.

Bider, I., Perjons, E. and Bork, D. (2021). 'Context is King: an Enterprise Model that Connects the Internal Structure with the Business Environment'. In: *IEEE International Enterprise Distributed Object Computing Workshop, Gold Coast, Australia*. URL: https://doi.org/10.1109/EDOCW52865.2021.00065.

Bork, D. and Alter, S. (2020). ‘Satisfying Four Requirements for More Flexible Modeling Methods: Theory and Test Case’. In: *Enterprise Modelling and Information Systems Architectures Journal* 15, pp. 1–25. URL: https://doi.org/10.18417/emisa.15.3.

Brenner, W., Karagiannis, D., Kolbe, L., Krüger, J., Leifer, L., Lamberti, H.-J., Leimeister, J. M., Österle, H., Petrie, C., Plattner, H., Schwabe, G., Uebernickel, F., Winter, R. and Zarnekow, R. (2014). ‘User, Use & Utility Research’. In: *Business & Information Systems Engineering* 6.1, pp. 55–61. URL: https://doi.org/10.1007/s12599-013-0302-4.

Bucher, T., Klesse, M., Kurpjuweit, S. and Winter, R. (2007). ‘Situational Method Engineering’. In: *Situational Method Engineering: Fundamentals and Experiences.* Ed. by J. Ralyté, S. Brinkkemper and A. Henderson-Sellers. Springer, pp. 33–48. URL: https://doi.org/10.1007/978-0-387-73947-2_5.

Chanias, S., Myers, M. D. and Hess, T. (2019). ‘Digital transformation strategy making in pre-digital organizations: The case of a financial services provider’. In: *Journal of Strategic Information Systems* 28.1, pp. 17–33. URL: https://doi.org/10.1016/j.jsis.2018.11.003.

Dąbrowska, J., Almpanopoulou, A., Brem, A., Chesbrough, H., Cucino, V., Di Minin, A., Giones, F., Hakala, H., Marullo, C., Mention, A. L., Mortara, L., Nørskov, S., Nylund, P. A., Oddo, C. M., Radziwon, A. and Ritala, P. (2022). ‘Digital transformation, for better or worse a critical multi- level research agenda’. In: *R&D Management* 52.5, pp. 930–954. URL: https://doi.org/10.1111/radm.12531.

de Kinderen, S., Kaczmarek-Heß, M., Ma, Q. and Razo-Zapata, I. S. (2021). ‘A Modeling Method in Support of Strategic Analysis in the Realm of Enterprise Modeling: On the Example of Blockchain-Based Initiatives for the Electricity Sector’. In: *Enterprise Modelling and Information System Architectures Journal*, pp. 1–36. URL: https://doi.org/10.18417/emisa.16.2.

Demirkan, H., Spohrer, J. C. and Welser, J. J. (2016). ‘Digital Innovation and Strategic Transformation’. In: *IT Professional* 18.6, pp. 14–18. URL: https://doi.org/10.1109/Mitp.2016.115.

Drechsler, A. and Hevner, A. R. (2018). ‘Utilizing, Producing, and Contributing Design Knowledge in DSR Projects’. In: *Designing for a Digital and Globalized World.* Ed. by S. Chatterjee, K. Dutta and R. P. Sundarraj. Vol. 10844. LNCS. Springer, pp. 82–97. URL: https://doi.org/10.1007/978-3-319-91800-6_6.

Feldman, M. S. and Orlikowski, W. J. (2011). ‘Theorizing Practice and Practicing Theory’. In: *Organization Science* 22.5, pp. 1240–1253. URL: https://doi.org/10.1287/orsc.1100.0612.

Fellmann, M., Sandkuhl, K., Gutschmidt, A. and Poppe, M. (2020). ‘Structuring Participatory Enterprise Modelling Sessions’. In: *The Practice of Enterprise Modeling.* Ed. by G. Janis and D. Bork, pp. 58–72. URL: https://doi.org/10.1007/978-3-030-63479-7_5.

Goodhue, D. L. and Thompson, R. L. (1995). ‘Task-Technology Fit and Individual Performance’. In: *MIS Quarterly* 19.2, pp. 213–236. URL: https://doi.org/10.2307/249689.

Gregor, S. (2006). ‘The Nature of Theory in Information Systems’. In: *MIS Quarterly* 30.3, pp. 611–642. URL: https://doi.org/10.2307/25148742.

Gregor, S. and Jones, D. (2007). ‘The anatomy of a design theory’. In: *Journal of the Association for Information Systems* 8.5, pp. 312–335. URL: https://doi.org/10.17705/1jais.00129.

Gutschmidt, A. (2019). ‘On the Influence of Tools on Collaboration in Participative Enterprise Modeling—An Experimental Comparison Between Whiteboard and Multi-touch Table’. In: *Advances in Information Systems Development.* Ed. by B. Andersson, B. Johansson,

C. Barry, M. Lang, H. Linger and C. Schneider, pp. 151–168. URL: https://doi.org/10.1007/978-3-030-22993-1_9.

Gutschmidt, A. (2022). 'Advantages and Limitations of Experiments for Researching Participatory Enterprise Modeling and Recommendations for Their Implementation'. In: *The Practice of Enterprise Modeling*, pp. 216–231. URL: https://doi.org/10.1007/978-3-031-21488-2_14.

Gutschmidt, A., Lantow, B., Hellmanzik, B., Ramforth, B., Wiese, M. and Martins, E. (2022). 'Participatory modeling from a stakeholder perspective: On the influence of collaboration and revisions on psychological ownership and perceived model quality'. In: *Software and Systems Modeling*, pp. 1–17. URL: https://doi.org/10.1007/s10270-022-01036-7.

Gutschmidt, A. and Richter, H. D. (2021). 'Personal Space and Territorial Behavior – Sharing a Tabletop in Collaborative Enterprise Modeling'. In: *Advances in Enterprise Engineering XIV*. Ed. by D. Aveiro, G. Guizzardi, R. Pergl and H. A. Proper, pp. 111–130. URL: https://doi.org/10.1007/978-3-%20030-74196-9_7.

Gutzwiller, T. A. (1994). *Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen.* Heidelberg: Physica. URL: https://doi.org/10.1007/978-3-642-52405-9.

Hafsi, M. and Assar, S. (2021). 'Proposal of a visual impact analysis approach based on enterprise modeling: case of customer experience projects in the banking sector'. In: *PoEM 2021 Workshops, Riga, Latvia.* URL: https://ceur-ws.org/Vol-3031/paper_3.pdf.

Hartl, E. and Hess, T. (2017). 'The Role of Cultural Values for Digital Transformation: Insights from a Delphi Study'. In: *Americas Conference on Information Systems, Boston, Massachusetts, USA.* URL: https://aisel.aisnet.org/amcis2017/Global/Presentations/8.

Hendry, J. and Seidl, D. (2003). 'The structure and significance of strategic episodes: Social systems theory and the routine practices of strategic change'. In: *Journal of Management Studies* 40.1, pp. 175–196. URL: https://doi.org/10.1111/1467-6486.00008.

Hevner, A. R., March, S. T., Park, J. and Ram, S. (2004). 'Design science in Information Systems research'. In: *MIS Quarterly* 28.1, pp. 75–105. URL: https://doi.org/10.2307/25148625.

Hinings, B., Gegenhuber, T. and Greenwood, R. (2018). 'Digital innovation and transformation: An institutional perspective'. In: *Information and Organization* 28.1, pp. 52–61. URL: https://doi.org/10.1016/j.infoandorg.2018.02.004.

Hoppenbrouwers, S. and Wilmont, I. (2010). 'Focused Conceptualisation: Framing Questioning and Answering in Model-Oriented Dialogue Games'. In: *IFIP Working Conference on The Practice of Enterprise Modeling*. Ed. by P. van Bommel, S. Hoppenbrouwers, S. Overbeek, E. Proper and J. Barjis, pp. 190–204. URL: https://doi.org/10.1007/978-3-642-16782-9_14.

Hund, A., Wagner, H.-T., Beimborn, D. and Weitzel, T. (2021). 'Digital innovation: Review and novel perspective'. In: *The Journal of Strategic Information Systems* 30.4. URL: https://doi.org/10.1016/j.jsis.2021.101695.

Karagiannis, D. and Kühn, H. (2002). 'Metamodelling Platforms'. In: *EC Web 2002: E-Commerce and Web Technologies*. Ed. by K. Bauknecht, A. Min Tjoa and G. Quirchmayer. Springer. URL: https://doi.org/10.1007/3-540-45705-4_19.

Kautz, T. and Winter, R. (2023). 'Digital Transformation Designer: Towards a Comprehensive, Collaborative and Easy-to-Use Modeling Support for Enterprise-wide Change Programs'. In: *Enterprise Modelling and Information Systems Architectures Journal.* (in press).

Klingebiel, R. and De Meyer, A. (2013). 'Becoming Aware of the Unknown: Decision Making During the Implementation of a Strategic Initiative'. In: *Organization Science* 24.1, pp. 133–153. URL: https://doi.org/10.1287/orsc.1110.0726.

Klos, C., Spieth, P., Clauss, T. and Klusmann, C. (2021). 'Digital Transformation of Incumbent Firms: A Business Model Innovation Perspective'. In: *IEEE Transactions on Engineering Management*, pp. 1–17. URL: https://doi.org/10.1109/Tem.2021.3075502.

Koutsopoulos, G. (2021). 'Capabilities in Crisis: A Case Study Using Enterprise Modeling for Change Analysis'. In: *Perspectives in Business Informatics Research. BIR 2021.* Ed. by R. A. Buchmann, a. Polini, B. Johannsson and D. Karagiannis. Springer, pp. 100–114. URL: https://doi.org/10.1007/978-3-030-87205-2_7.

Labusch, N., Aier, S. and Winter, R. (2013). 'Beyond Enterprise Architecture Modeling – What are the Essentials to Support Enterprise Transformations?' In: *International Workshop on Enterprise Modelling and Information Systems Architectures, St. Gallen, Switzerland.* URL: https://dl.gi.de/bitstream/handle/20.500.12116/17239/13.pdf?sequence=1&isAllowed=y.

Labusch, N. and Winter, R. (2013). 'Towards a Conceptualization of Architectural Support for Enterprise Transformation'. In: *European Conference on Information Systems, Utrecht, The Netherlands.* URL: http://aisel.aisnet.org/ecis2013/137.

Li, L., Su, F., Zhang, W. and Mao, J. Y. (2018). 'Digital transformation by SME entrepreneurs: A capability perspective'. In: *Information Systems Journal* 28.6. URL: https://doi.org/10.1111/isj.12153.

Lukyanenko, R., Storey, V. C. and Pastor, O. (2022). 'System: A core conceptual modeling construct for capturing complexity'. In: *Data & Knowledge Engineering* 141, pp. 1–29. URL: https://doi.org/10.1016/j.datak.2022.102062.

Maier, J. R. A. and Fadel, G. M. (2009). 'Affordance based design: a relational theory for design'. In: *Research in Engineering Design* 20.1, pp. 13–27. URL: https://doi.org/10.1007/s00163-008-0060-3.

Malinova, M. and Mendling, J. (2021). 'Cognitive Diagram Understanding and Task Performance in Systems Analysis and Design'. In: *MIS Quarterly* 45.4, pp. 2101–2158. URL: https://doi.org/10.25300/misq/2021/15262.

Markus, M. L. (2004). 'Technochange management: using IT to drive organizational change'. In: *Journal of Information Technology* 19.1, pp. 4–20. URL: https://doi.org/10.1057/palgrave.jit.2000002.

Matt, C., Hess, T. and Benlian, A. (2015). 'Digital Transformation Strategies'. In: *Business & Information Systems Engineering* 57.5, pp. 339–343. URL: https://doi.org/10.1007/s12599-015-0401-5.

Morton, J., Wilson, A. and Cooke, L. (2016). 'Open Strategy Initiatives: Open, IT-Enabled Episodes of Strategic Practice'. In: *Pacific Asia Conference on Information Systems, Chiayi, Taiwan.* URL: https://aisel.aisnet.org/pacis2016/212.

Mylopoulos, J. (1998). 'Information modeling in the time of the revolution'. In: *Information Systems* 23.3, pp. 127–155. URL: https://doi.org/10.1016/S0306-4379(98)00005-2.

Nicolaescu, P., Rosenstengel, M., Derntl, M., Klamma, R. and Jarke, M. (2016). 'View-Based Near Real- Time Collaborative Modeling for Information Systems Engineering'. In: *Advanced Information Systems Engineering. CAiSE 2016.* Ed. by S. Nurcan, P. Soffer, M. .Bajec and J. Eder. Springer, pp. 3–17. URL: https://doi.org/10.1007/978-3-319-39696-5_1.

Ostergaard, K. J. and Summers, J. D. (2009). 'Development of a systematic classification and taxonomy of collaborative design activities'. In: *Journal of Engineering Design* 20.1, pp. 57–81. URL: `https://doi.org/10.1080/09544820701499654`.

Pastor, O., Noel, R., Panach, I. and Ruiz, M. (2022). 'The LiteStrat Modelling Method: Towards the Alignment of Strategy and Code'. In: *Domain-Specific Conceptual Modeling*. Ed. by D. Karagiannis, M. Lee, K. Hinkelmann and W. Utz. Springer, pp. 141–159. URL: `https://doi.org/10.1007/978-3-%20030-93547-4_7`.

Peffers, K., Tuunanen, T., Rothenberger, M. A. and Chatterjee, S. (2007). 'A design science research methodology for Information Systems Research'. In: *Journal of Management Information Systems* 24.3, pp. 45–77. URL: `https://doi.org/10.2753/Mis0742-1222240302`.

Pellegrinelli, S. (2011). 'What's in a name: Project or programme?' In: *International Journal of Project Management* 29.2, pp. 232–240. URL: `https://doi.org/10.1016/j.ijproman.2010.02.009`.

Peppard, J., Galliers, R. D. and Thorogood, A. (2014). 'Information systems strategy as practice: Micro strategy and strategizing for IS'. In: *Journal of Strategic Information Systems* 23.1, pp. 1–10. URL: `https://doi.org/10.1016/j.jsis.2014.01.002`.

Persson, A. and Stirna, J. (2001). 'Why Enterprise Modelling? An Explorative Study into Current Practice'. In: *Advanced Information Systems Engineering. CAiSE 2001.* Ed. by K. Dittrich, A. Geppert and M. Norrie, pp. 465–468. URL: `https://doi.org/10.1007/3-540-%2045341-5_31`.

Project Management Institute (2008). *The standard for program management.* Tech. rep. Newtown, Pennsylvania, USA.

Proper, H. A. and Guizzardi, G. (2021). 'On Domain Conceptualization'. In: *Advances in Enterprise Engineering XIV.* Ed. by D. Aveiro, G. Guizzardi, R. Pergl and H. A. Proper, pp. 49–69. URL: `https://doi.org/10.1007/978-3-030-74196-9_4`.

Recker, J. (2017). 'Continued use of process modeling grammars: the impact of individual difference factors'. In: *European Journal of Information Systems* 19.1, pp. 76–92. URL: `https://doi.org/10.1057/ejis.2010.5`.

Renger, M., Kolfschoten, G. L. and de Vreede, G.-J. (2008). 'Challenges in Collaborative Modeling: A Literature Review'. In: *International Workshop on Cooperation and Interoperability, Architecture and Ontology; Workshop on Enterprise and Organizational Modeling and Simulation.* Ed. by J. L. G. Dietz, A. Albani and J. Barjis, pp. 61–77. URL: `https://doi.org/10.1007/978-3-540-%2068644-6_5`.

Roelens, B. (2022). 'PGA 2.0: A Modeling Technique for the Alignment of the Organizational Strategy and Processes'. In: *Domain-Specific Conceptual Modeling*. Ed. by D. Karagiannis, M. Lee, K. Hinkelmann and W. Utz. Springer, pp. 121–139. URL: `https://doi.org/10.1007/978-3-030-93547-4_6`.

Rosemann, M. and Vessey, I. (2008). 'Toward improving the relevance of information systems research to practice: The role of applicability checks'. In: *MIS Quarterly* 32.1, pp. 1–22. URL: `https://doi.org/10.2307/25148826`.

Roth, M., Kasperek, D. and Lindemann, U. (2014). 'Verifying the Abstraction Level of Structural Models'. In: *Conference on Systems Engineering Research, Redondo Beach, California, USA.* URL: `https://doi.org/10.1016/j.procs.2014.03.061`.

Rouse, W. B. (2005). 'Enterprises as systems: Essential challenges and approaches to transformation'. In: *Systems Engineering* 8.2, pp. 138–150.

Rummel, F., Husig, S. and Steinhauser, S. (2022). ‘Two archetypes of business model innovation processes for manufacturing firms in the context of digital transformation’. In: *R & D Management* 52.4, pp. 685–703. URL: https://doi.org/10.1111/radm.12514.

Sandkuhl, K., Fill, H.-G., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A., Schwabe, G., Uludag, Ö. and Winter, R. (2018). ‘From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling’. In: *Business & Information Systems Engineering* 60.1, pp. 69–80. URL: https://doi.org/10.1007/s12599-017-0516-y.

Sandkuhl, K. and Seigerroth, U. (2020). ‘Participative or Conventional Enterprise Modelling? Multiple- Case Analysis on Decision Criteria’. In: *European Conference on Information Systems, Marrakech, Morocco*. URL: https://aisel.aisnet.org/ecis2020_rp/161.

Sandkuhl, K., Stirna, J., Persson, A. and Wißotzki, M. (2014). *Enterprise Modeling: Tackling Business Challenges within the 4EM Method*. Berlin, Heidelberg: Springer. URL: https://doi.org/10.1007/978-%203-662-43725-4.

Sneath, P. H. and Sokal, R. R. (1973). *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. San Francisco, California, USA: Freeman.

Ssebuggwawo, D., Hoppenbrouwers, S. and Proper, H. A. (2010). ‘Assessing Collaborative Modeling Quality Based on Modeling Artifacts’. In: *IFIP Working Conference on The Practice of Enterprise Modeling*. Ed. by P. van Bommel, S. Hoppenbrouwers, S. Overbeek, E. Proper and J. Barjis, pp. 76–90. URL: https://doi.org/10.1007/978-3-642-16782-9_6.

Steen, M. W. A., Akehurst, D. H., ter Doest, H. W. L. and Lankhorst, M. M. (2004). ‘Supporting viewpoint-oriented enterprise architecture’. In: *International Conference on Enterprise Distributed Object Computing, Monterey, California, USA*. URL: https://doi.org/10.1109/EDOC.2004.1342516.

Stirna, J. (2001). ‘The Influence of Intentional and Situational factors on Enterprise Modelling Tool Acquisition in Organisations’. PhD thesis. Kista, Sweden: Royal Institute of Technology and Stockholm University. URL: http://urn.kb.se/resolve?urn=urn%3Anbn%3Ase%3Akth%3Adiva-%203266.

Stirna, J. and Persson, A. (2018). *Enterprise Modeling*. Cham: Springer. URL: https://doi.org/10.1007/978-3-%20319-94857-7.

Stirna, J., Persson, A. and Sandkuhl, K. (2007). ‘Participative Enterprise Modeling: Experiences and Recommendations’. In: *International Conference on Advanced Information Systems Engineering*. Ed. by J. Krogstie, A. Opdahl and G. Sindre, pp. 546–560. URL: https://doi.org/10.1007/978-3-540-%2072988-4_38.

Stirna, J. and Sandkuhl, K. (2018). ‘Enterprise Modelling: Establishing the Fundament for Capability Management’. In: *Capability Management in Digital Enterprises*. Ed. by K. Sandkuhl and J. Stirna, pp. 85–100. URL: https://doi.org/10.1007/978-3-319-90424-5_5.

Ubiparipovic, B., Matkovic, P. and Pavlicevic, V. (2022). ‘Key activities of digital business transformation process’. In: *Strategic Management* (1-8). URL: https://doi.org/10.5937/StraMan2200016U.

van Gils, B., Hoppenbrouwers, S. and Proper, H. A. (2022). ‘Conceptual Modeling in Digital Transformations – Enabling enterprise design dialogues’. In: *The Practice of Enterprise Modeling 2022 Forum, London, UK*. URL: https://ceur-ws.org/Vol-3327/paper05.pdf.

Venable, J. R. (2006). ‘The Role of Theory and Theorising in Design Science Research’. In: *International Conference on Design Science Research in Information Systems and Techno-*

logy, Claremont, California, USA. URL: https://www.researchgate.net/publication/228670522_The_role_of_theory_and_theorising_in_%20design_science_research.

Vessey, I. and Galletta, D. (1991). 'Cognitive Fit: An Empirical Study of Information Acquisition'. In: *Information Systems Research* 2.1, pp. 63–84. URL: https://doi.org/10.1287/isre.2.1.63.

vom Brocke, J., Winter, R., Hevner, A. and Maedche, A. (2020). 'Special Issue Editorial –Accumulation and Evolution of Design Knowledge in Design Science Research: A Journey Through Time and Space'. In: *Journal of the Association for Information Systems* 21.3, pp. 520–544. URL: https://doi.org/10.17705/1jais.00611.

Wessel, L., Baiyere, A., Ologeanu-Taddei, R., Cha, J. and Jensen, T. B. (2021). 'Unpacking the Difference Between Digital Transformation and IT-Enabled Organizational Transformation'. In: *Journal of the Association for Information Systems* 22.1, pp. 102–129. URL: https://doi.org/10.17705/1jais.00655.

Whittington, R. (2014). 'Information Systems Strategy and Strategy-as-Practice: A joint agenda'. In: *Journal of Strategic Information Systems* 23.1, pp. 87–91. URL: https://doi.org/10.1016/j.jsis.2014.01.003.

Winter, R. (2011). 'Design of Situational Artefacts—Conceptual Foundations and Their Application to IT/Business Alignment'. In: *Information Systems Development*. Ed. by J. Pokorny, V. Repa, K. Richta, W. Wojtkowski, H. Linger, C. Barry and M. Lang, pp. 35–49. URL: https://doi.org/10.1007/978-%201-4419-9790-6_3.

Wißotzki, M. and Sandkuhl, K. (2017). 'The Digital Business Architect – Towards Method Support for Digital Innovation and Transformation'. In: *The Practice of Enterprise Modeling*. Ed. by G. Poels, F. Gailly, E. Serral Asensio and M. Snoeck, pp. 352–362. URL: https://doi.org/10.1007/978-3-319-%2070241-4_24.

Yoshioka, T., Herman, G., Yates, J. and Orlikowski, W. (2001). 'Genre taxonomy'. In: *ACM Transactions On Information Systems* 19.4, pp. 431–456. URL: https://doi.org/10.1145/502795.502798.

Chapter 13

Omnichannel Processes in Retailing SMEs: Digitalisation in a Furniture Company

Jörg Becker, Paul Kruse, and Andreas Hermann

13.1 The Starting Point of Digital Transformation

Digital transformation is a complex endeavour companies face to stay competitive in an ever-changing business environment (Fletcher and Griffiths 2020). Various options are available to digitalise a company through the continuous development of new digital technologies. However, to implement the new technology, the processes within the organisation need to be transformed as well. Hence, many scholars develop procedure models structuring the general transformation process (Barann et al. 2019). To apply the digital transformation successfully, general knowledge about the existing process landscape is required. With this process knowledge, the company's digital transformation will achieve the desired result. Unfortunately, the general process knowledge in small and medium-sized enterprises (SMEs) often needs to be improved (Mendling et al. 2010). As a consequence, SMEs often do not know the interrelation of their processes and rather conduct an unstructured transformation process.

To conduct a structured transformation, SMEs need good Business Process Management (BPM) practices. An overview of the existing processes is required to set up BPM successfully. A rigorous process model can provide this overview. As models are an abstraction of reality, the respective process model depicts a real-life process in an abstract form. Without the process model, employees know their daily routines and often do not have the bigger picture in mind. Collecting and synthesising the individual processes into an exhaustive process model can provide a general overview of joining the dots. Yet, achieving this overview requires a process-oriented mindset and appropriate modelling skills. However, many different modelling techniques consist of different modelling layers and loose or fixed levels of detail. Hence, a general understanding of the respective modelling technique is required. The required knowledge to model processes correctly is often lacking in SME as they often do not have the resources or personnel to apply proper process modelling within the organisation (Mendling et al. 2010). Next, external experts, such as consulting companies, are often too expensive for SMEs. Even though a consulting company conducted the modelling, the employees of the SME need additional training to read and understand the respective models. Summarising, a rigorous BPM is based on well-modelled business processes which require expert knowledge and is often unnecessarily complex.

Hence, setting up rigour BPM requires a modelling technique applicable to everyone and, above all, understandable. The technique should provide a model that employees can easily understand without having participated in external training courses beforehand. Modelling languages generally have a high degree of modelling freedom (Becker et al. 1995). The modelling freedom makes it challenging to create comparable, analysable, uniform models. In addition, the models are sometimes very detailed, which requires a great deal of time to create and are sometimes only understandable with expert knowledge. Hence, the modelling technique should make the created models comparable, analysable and reusable to provide the best benefit possible to the organisation. Such characteristics are provided by 'icebricks' which is a modelling technique based on three different layers to present the process landscape of an organisation in different degrees of detail. Besides, 'icebricks' can also be used to support other aspects of BPM, such as system selection or the conceptualization of specific ideas. In the following, some aspects of BPM are covered and 'icebricks' is introduced. Afterwards, 'icebricks' simplicity and usability are shown in a furniture company. We end with a short discussion and conclusion.

13.2 Background

13.2.1 Some Aspects of Business Process Management

Due to increasing globalization, the management of (business) processes plays an increasingly important role in practice and academia (Ko et al. 2009). Consequently, BPM research emerged in different principles, methods, and tools that combine knowledge from different disciplines, such as information technology, management science, and industrial engineering (Van Der Aalst et al. 2016). A definition of BPM is delivered by van der Aalst (2004, p. 3) as they define BPM as *'supporting business processes using methods, techniques and software to design, enact, control and analyse operational processes involving humans, organisations, applications, documents and other sources of information'*. Thereby, BPM is not a one-time activity but a continuous task. As Hammer (2015, p. 12) state: *'Every good process eventually becomes a bad process'*. Hence, Dumas et al. (2018, p. 22) considered BPM as a *'continuous cycle comprising the following phases: Process identification, Process discovery, Process analysis, Process redesign, Process implementation, and Process Monitoring'* (see Figure 13.1).

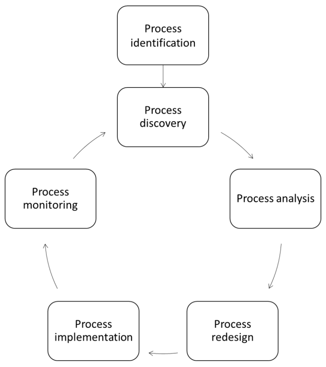


Figure 13.1: BPM lifecycle (Dumas et al. 2018, p. 23)

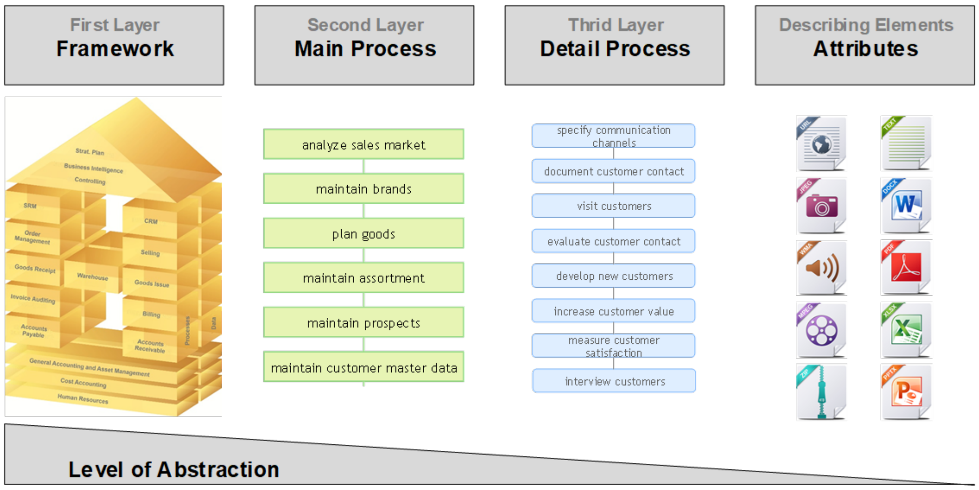


Figure 13.2: Layers of the ‘icebricks’ method (Becker et al. 2016)

Within the Process identification, a problem within the business is identified and the related process or a set of processes is selected. Afterwards, the Process discovery or sometimes called as-is modelling, is conducted. That means the current state of a process is documented before the Process analysis follows. The analysis output is a collection of problems ordered by their potential impact and estimated effort to solve them (Dumas et al. 2018, p. 23). The Process redesigns' goal is the identification of process improvements or changes that optimise the process. The result of this phase is a to-be model, which is then implemented in the following step. The last step monitors the redesigned process, which may create new issues.

Consequently, problem identification and thorough as-is modelling are necessary to perform proper BPM. Without thorough preparation, all subsequent steps are based on incomplete or incorrect information. Since general process knowledge is not explicitly available in SMEs, i. e., no suitable personnel are available and no process documentation exists, this should be considered when selecting methods, technology and software tools.

13.2.2 icebricks

An approach to transparently mapping a company's processes is the ‘icebricks’ method (Becker et al. 2016). Thereby, different aspects are covered by ‘icebricks’, such as organisational design to the preparation and support of application system design. However, the initial scope of ‘icebricks’ is in organisational documentation and continuous process management.

‘icebricks’ is designed in such a way that a systematic design of a process landscape is guaranteed by considering the ‘Guidelines of Modelling’ (GoM) (Becker et al. 1995). Applying the GoM, a comparable level of modelling detail in modelling projects is ensured (Becker et al. 1995; Mendling et al. 2010). The design of ‘icebricks’ is depicted in Figure 13.2 and contains the following layers: Framework, Main Process, and Detail Process. The structure of the layers is ordered hierarchically. The highest abstraction is the Framework (first layer) which represents the modelled process landscape, i. e., the essential functional areas of a company. Each element of the Framework is (by definition) a Main Process (second layer). Each Element of a Main Process is (also by definition) a Detail Process (third layer). By only using the two modelling elements (process steps and control flow), 'icebricks' *'contains the complexity of the resulting models as far as possible and enables modelling even for non-experts'* (Becker et al. 2016, p. 420). In most (non-'icebricks') modelling projects, only a tiny part of the available elements is used (zur Muehlen and Recker 2008; Siau et al. 2005). 'icebricks' offers only very few modelling elements (mainly process steps). If additional information is required, the modeller can use typed attributes to augment the process models. In 'icebricks', typed attributes can vary from *'simple textual attributes for description to evaluable, numeric attributes to attributes that enable the annotation of responsibilities of processes or of supporting IT systems'* (Becker et al. 2016, p. 420).

In addition to the higher comparability of the models due to the layered structure, the designation of the process steps plays an equally important role with regard to the model quality. Delfmann et al. (2009) figured out that a high degree of freedom in modelling supports the usage of a multitude of words within the process model. Hence, the comparability of models is limited and even hampers the understanding of models. This problem is addressed in 'icebricks' through a glossary and a standardised verb-object-phrase structure that is based on the findings from (Mendling et al. 2010). The standardised phrase structure in 'icebricks' leads to high model quality and thus inevitably promotes model understanding (Delfmann et al. 2009). The glossary in 'icebricks' contains Business Objects and Activities. The glossary allows only desired, and meaningful process steps to be used for designation. That means the procedure applied to a Business object is predefined. The order may vary between English and German according to the respective grammar. In an English model, the predefined order is '<Activity><Business Object>', like 'print invoice' or 'pick article'.

Next to the process view, an organisational view is provided by 'icebricks' picturing the organisational structures within companies. Besides, a view for resources is integrated to depict the used artefacts, e. g., IT systems, in a process. Consequently, *'the 'icebricks' method can be considered a comprehensive method of managing entrepreneurial activities'* (Becker et al. 2016, p. 421). Furthermore, reference models are offered for quite a few domains. Most known is the reference model for the retail industry (the 'retail-H model'), there are also models for the manufacturing industry, and procedure models for different projects (process management, software selection, software implementation, DIN-EN-ISO 9001quality certification, etc.).

13.3 Transformation of a System Landscape

13.3.1 Case Motivation

The retail landscape is experiencing tremendous changes due to the continuous progress in information and communication technologies. To improve the customer experience, more and more retailers have started integrating advanced omni-channel management concepts. Omni-channel management is *'the synergetic management of the numerous available channels and customer touchpoints'* across various information systems (Verhoef et al. 2015, p. 3). However, managing channels and touchpoints of systems of different vendors with different data structures via system interfaces is a complex and challenging task. Additionally, online players such as Amazon and Zalando, who are already well-positioned to implement

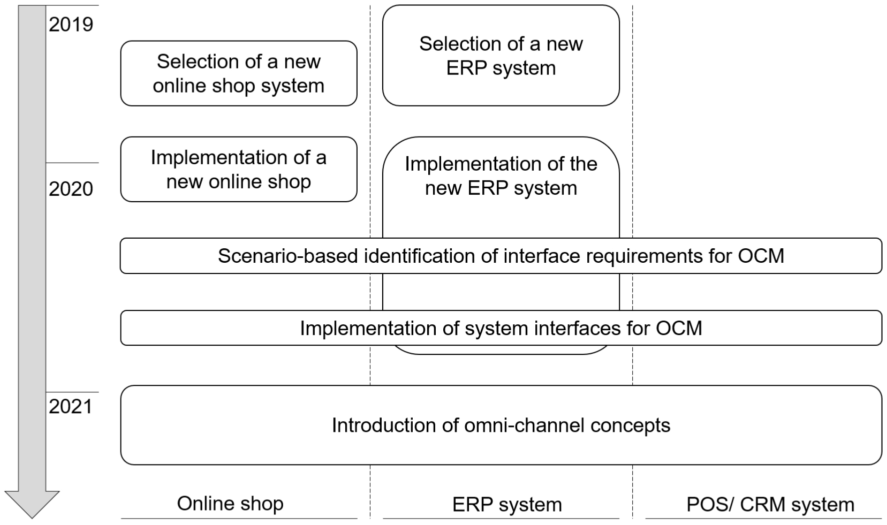

2019
2020
2021
Selection of a new online shop system
Selection of a new ERP system
Implementation of a new online shop
Implementation of the new ERP system
Scenario-based identification of interface requirements for OCM
Implementation of system interfaces for OCM
Introduction of omni-channel concepts
Online shop
ERP system
POS/ CRM system

Figure 13.3: Roadmap for the modernization of the system landscape

omni-channel scenarios because of their sophisticated IT infrastructure, are pushing into the offline market. This creates high expectations of channel integration among customers, creating additional pressure on medium-sized companies. In order to meet the customers' expectations, medium-sized companies must also face the challenge of implementing omni-channel management solutions as well. Due to the complexity of establishing omni-channel management, a process-driven approach should be followed. In a step-wise fashion, it is reasonable to follow the general idea of the BPM lifecycle (see Section 13.2) and model, analyse, and inform the implementation of a retailer's processes. For this endeavour, selecting a suitable process modelling language is crucial. To demonstrate that 'icebricks' is a state-of-the-art process modelling language for such seemingly complex, process-driven IT projects, the case of a retailing company is presented in what follows.

13.3.2 The Case of Møblar

Møblar GmbH is a German retail company selling furniture, decoration, and accessories, mainly imported from Asia. Within the last 25 years, the company had grown from being a single store to a group of 18 legally independent stores and an online shop with more than 250 employees. The increased customer expectations and their desire for omni-channel scenarios such as click & collect or omni-channel vouchers caused the decision to implement a corresponding solution. Additionally, several shortcomings within the system landscape constituted obstacles for daily operations, e. g., missing warehouse management functionalities of the ERP system or problems with the performance and scalability of the online shop system. Therefore, Møblar's decides to renew its system landscape and implement omni-channel management. More specifically, it was agreed to select and implement a new ERP and online shop system, implement system interfaces that support omni-channel management, and introduce selected omni-channel concepts.

13.3.3 Actions Taken

For selecting a new ERP system, a structured and sound procedure was followed (see Vering 2002). A comprehensive analysis of the company's as-is situation and the definition of a to-be state were focused as a first step.

As a key result of the as-is analysis, an exhaustive overview of the business processes at Møblar's was created. To arrive at that process overview, six days of semi-structured interviews were spent to document Møblar's core business processes. The complete process documentation was created with the 'icebricks' modelling notation and web tool (Clever 2016). In addition, the 'retail-H' reference model (Becker and Schütte 2004) served as a blueprint for structuring the interviews and the process modelling. Besides, the system landscape and over 180 documents, receipts, and forms were analysed. The previously created process models from this analysis were enriched with process-relevant information and persisted as attributes (see Section 13.2.2).

Moreover, the employees' wishes for improvement were gathered. A to-be model has been derived based on the analysis of the as-is state and these wishes. Just as the as-is process documentation, this to-be model has been created using the 'icebricks' modelling notation. The to-be model served as the basis for deriving over 400 requirements. These have been structured along the main processes and documented within the 'icebricks' web tool.

The second phase, the pre-selection of suitable systems, required a comprehensive overview of the ERP systems market. Therefore, a dedicated platform has been used, which lists over 500 vendors. The as-is, as well as the to-be process models, supported the evaluation of suitable system vendors. Finally, three systems have been deemed promising to be introduced at Møblar's. The vendors of these three systems were invited for two days of system presentation and related discussions. Vering (2002) describes a structured test roadmap of over 90 concrete process scenarios and functionality questionnaires to be demonstrated on a test instance of the systems, which has been elaborated. Put differently, the vendors were asked to demonstrate how their system implements selected processes that were previously modelled in 'icebricks'.

Moreover, the vendors were requested to show the system's omni-channel management functionality. This step was meant to ensure that the future system at Møblar's supports selected omni-channel management scenarios (e. g., a click & collect process) and is equipped with the corresponding interfaces. As the future IT system landscape is supposed to provide dedicated interfaces that enable selected omni-channel scenarios, corresponding interface requirements should be documented. Therefore, a collection of system interface requirements describing the technical perspective of the desired Omni-channel-scenarios was created. For this task, the functionality of 'icebricks' to document typed attributes was utilised (see Section 13.2.2). Omni-channel-scenario requirements were broken down into a corresponding core subject, a scenario description, associated customer benefits, relevant data, and organisational implications. Based on the evaluation of the system's functionality and suitability to implement the daily processes at Møblar's, a final decision was made.

13.3.4 Case Results

As of today, the five steps of the digitalisation roadmap (see Figure 13.3) have been completed: the selection of a new ERP system, the selection of a new online shop, the implementation

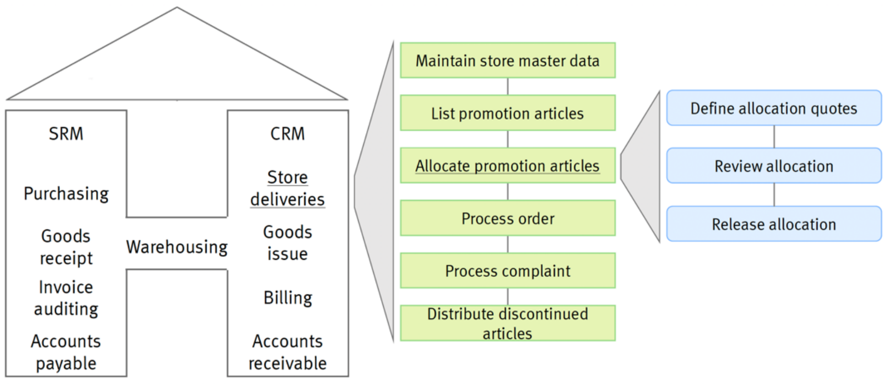


Figure 13.4: Example of the ‘icebricks’ to-be model

of a new online shop, the scenario-based identification of interface requirements for omni-channel management, the implementation of the new ERP system, and the implementation of system interfaces for omni-channel management. In particular, the selection of a new ERP system and the scenario-based identification of interface requirements for omni-channel management benefited from the application of ‘icebricks’ for activities such as modelling or process analysis.

As described previously, the structured selection of a new ERP system has resulted in identifying and selecting one vendor from a set of over 500 whose system suits Møblar’s requirements best. This vendor was able to prevail over the other top candidates since he provided the most suitable synthesis of requirements fulfilment, economic viability, performance in the system presentations, the relevance of reference customers, and future readiness of the technical basis of the system. In addition, selecting of a new ERP system led to further valuable results. First, a comprehensive as-is model over three levels (framework, main processes, and detailed processes) has been created using the ‘icebricks’ modelling notation and web tool. On the framework level, the eleven main processes, i. e., the core processes of Møblar’s on the highest level of abstraction, have been modelled. Each main process consists of several detailed processes (e. g., store deliveries consists of six detailed processes) and each detailed process consists of several process steps (e. g., allocate promotion articles consists of three process bricks). The process steps have been further detailed with attributes like description, responsible organisational unit, IT system, and attached documents and forms. The analysis of the as-is situation has revealed numerous improvement potentials. As a result, 85 of the over 215 process steps of Møblar’s as-is model were modified when the to-be model was derived. The to-be model was used to formulate the functional requirements for the selection of a new ERP system and to provide the vendor of the new ERP system with process descriptions for the system implementation. Exemplary improvement potentials, structured along Møblar’s main processes, are:

Main Process: *Supplier Relationship Management*

As-is situation: Suppliers’ declarations and certificates are not attached to their corresponding master data but maintained manually in the file system.

Improvement potential: Hierarchical material requirements planning, integrated display of the container capacity level, and automated splitting into container-suitable order sizes within the ERP system.

Main Process: *Purchasing*

As-is situation: Manual planning of order quantities and splitting into container-suitable order sizes

Improvement potential: Integrated archive solution for classified storage of all declarations and certificates within the master data of the ERP system.

Main Process: *Goods receipt*

As-is situation: Excel-based management of shipments without reference to the order in the ERP system.

Improvement potential: Integrated shipment management in the ERP system, allowing an n-to-n-relationship between orders and containers at item level.

Main Process: *Invoice auditing*

As-is situation: Manual archiving of invoices without reference to the order in the ERP system.

Improvement potential: Automatic archiving of invoices directly within the corresponding transaction in the ERP system.

Main Process: *Accounts payable*

As-is situation: Excel-based advance payment management with manual look-up of exchange rates.

Improvement potential: Integrated advance payment management with automatic update of the daily exchange rates of the European Central Bank within the ERP system.

Main Process: *Warehousing*

As-is situation: Excel-based warehouse management including the tracking of articles' stock locations and empty storage locations.

Improvement potential: Integrated warehouse management including automatic storage location selection and management within the ERP system.

Main Process: *CRM*

As-is situation: Paper-based customer loyalty card on which sales must be registered by hand.

Improvement potential: Digital customer loyalty card and automatic sales tracking within the ERP system.

Main Process: *Store deliveries (selling)*

As-is situation: Excel-based article allocation to the individual sales locations.

Improvement potential: Integrated and automated article allocation by defined quotes within the ERP system.

Main Process: *Goods issue*

As-is situation: Manual generation of pallet labels via Excel.

Improvement potential: Generation and printing of pallet labels directly within the ERP system.

Main Process: *Billing*

As-is situation: Excel-based tracking of invoiced deliveries and credit notes.

Improvement potential: Standard report for collective invoices and credit notes within the ERP system.

Main Process: *Accounts receivable*

As-is situation: Excel-based documentation of incoming and outgoing payments.

Improvement potential: Integrated payment management within the financial accounting module of the ERP system.

As a result of the actions taken, the project team identified and documented a set of omni-channel scenarios relevant to Møblar's. In total, eight scenarios were identified and analysed. In the following, an excerpt from essential omni-channel scenarios is presented, focusing on the affected systems, anticipated customer benefits, the exchanged data, and potential organisational implications.

Omni-channel vouchers: *A customer can buy (or receive) a voucher from any store or the online shop and redeem it in any store or the online shop.*

Affected systems: CRM, online Shop, POS (point of sale)

Customer benefit: The customer benefits from an alternative payment method that is not bound to a single location.

Data exchanged: The POS and the online shop system must be able to check the validity and value of vouchers in real-time. Therefore, one system must be responsible for managing all vouchers, and the others must be linked with a system interface that allows real-time requests and responses. In the case of Møblar's, the CRM system includes voucher management functionality and should be responsible for managing the vouchers. Moreover, vouchers must have a unique identifier (e. g., an EAN13, scannable as a barcode). Once the voucher has been redeemed, the CRM system needs to update its value.

Organisational implications: If the issuing store and the store where a voucher is redeemed differ, an internal settlement process must be in place since every store is legally independent.

Digital customer loyalty program: *A customer can collect reward points with purchases in any store or the online shop with a digital customer loyalty card and redeem points as a means of payment.*

Affected systems: CRM, online shop, POS

Customer benefit: The customer is being rewarded for his loyalty in the form of a cashback.

Data exchanged: Each customer participating in the digital customer loyalty program must have customer master data including a unique customer identity (CID) number. Each transaction at the POS and the online shop must register the CID and report the turnover to the system which is responsible for managing the digital customer loyalty program. Moreover, the POS and online shop systems must be able the check a customer's reward points. At Møblar's, the digital customer loyalty program has been integrated into the CRM system, because the system interface to the POS is sufficiently fast and most date exchange is expected between these systems.

Organisational implications: If a customer returns articles, the reward points need to be corrected. Moreover, an internal settlement process must be in place, since a customer may collect reward points in different stores but can redeem the points in only one store at a time.

Click & collect: *A customer orders an article online, selects the desired store where the article is being picked up later.*

Affected systems: CRM, POS, online shop, ERP

Customer benefit: The customer benefits from increased flexibility in a way that she/he chooses the preferred time and location for the pickup of the articles.

Data exchanged: The classical click & collect scenario requires the online shop to transfer data relevant to the order and customer. The order information, including the article data, customer data, and the pick-up location, needs to be readily available in the POS system. At the same time, this information is supposed to trigger a reservation of the selected articles in the ERP system. In the case of Møblar's, the POS system requests this data from the CRM system. Therefore, the CRM system has to exchange the order data with the ERP system.

Organisational implications: The basic scenario is dependent on an internal settlement process to make sure the store at which the customer picks up his/her articles gets the credit for the order. Furthermore, depending on the store, internal transport to a specified pick-up area may be necessary. The online shop should receive a provision. In cases, where articles are not in stock at the desired pick-up location, an intercompany order needs to be placed and settled.

Ship to home: *A customer requests an article in a store that is out-of-stock; another store which has the article in stock ships it to his/her home.*

Affected systems: POS, CRM, ERP

Customer benefit: The customer saves time she/he might have invested in searching for alternatives or purchasing the desired article from another store.

Data exchanged: The scenario requires the exchange of information regarding the customer (name and address) and his order. The POS system needs to provide this data to the ERP, which triggers the shipment process (e. g., reservation of the articles and notification of the logistics service provider) in the issuing store. Since the POS system stores this data in the CRM system in the case of Møblar's, the ERP system receives this data from the CRM system. Afterwards, the customer needs to be informed about the initiated delivery through automatic messaging from the ERP system.

Organisational implications: This scenario triggers an internal settlement process because the issuing store is not the same as the debiting store. Furthermore, system integration with the logistics service provider may be required to support this process. During the order process at the point of sale, the customer has to provide his name and address. Therefore, the customer needs to consent to the transmission of his personal data for shipment purposes. This requires the setup of a support process to conform to the General Data Protection Regulation GDPR.

Return of articles: *A customer returns one or more articles in any store, which he has bought online.*

Affected systems: POS, CRM, online shop, ERP

Customer benefit: The customer benefits from a more convenient process, i. e., she/he does not need to initiate a shipment. She/he might save time and costs.

Data exchanged: When the customer returns her/his article(s) at a store, the POS system needs to take track of the articles and the order. The stock of the returned articles must be updated in the ERP system and the order balance needs to be recalculated so that the customer receives a credit note. In the case of Møblar's, the CRM system receives the data from the POS system and forwards them to the ERP system.

Organisational implications: Regardless of the store where the customer returns her/his article(s) to, an internal settlement process will follow as the online shop is legally separated from the stores.

This collection of omni-channel scenarios serves as a blueprint for the following implementation of the system interfaces and marks a milestone towards omni-channel retailing at Møblar's. The pre-defined goals could be achieved by following a tailored procedure that is focusing on quick wins. Overall, the project resulted in a well-conceived concept for the future system landscape at Møblar's with a focus on enabling omni-channel retailing.

13.4 Lessons Learned

To perform efficient and cost-effective BPM, the use and adaptation of reference models are highly recommended as a starting point. An example of a reference model embedded in the 'icebricks' method is the 'retail-H model' (Becker and Schütte 2004). Especially in the retail context, the value of the retail-H model in combination with 'icebricks' has been demonstrated in many cases. Multiple practical projects have shown that the reduction in complexity of the resulting models has promoted understanding of the models by project participants and that model comparability and quality have been consistently high. Within these projects, different foci were followed, such as *'pure process documentation, the support of software selection projects and certification projects'* (Becker et al. 2016).

In the case of Møblar, 'icebricks' is used to first model the business processes, change the processes to fit the new challenges of omnichannel business, and, finally, as the foundation for an ERP software selection. One learning from the case study concerns the applicability of 'icebricks' in the context of the digital transformation of SMEs. One reason for this success is the build-in consideration of the GOMs (guidelines of accuracy, relevance, efficiency, clarity, comparability, and systematic structure) in the 'icebricks' method. First, the applied glossary ensures the usage of the same activities and business objects in all processes. The glossary addresses the guideline of clarity and comparability. Second, the applied layer architecture of 'icebricks' provides a well-structured process model (see Figure 13.4). The layered architecture addresses the guidelines of clarity, comparability, and systematic structure. Third, the layered architecture ensures that the mapped processes are broken down into simple process sections. The resulting processes map the complex process landscape into simple and understandable process models. The resulting process can be quickly checked and validated by domain experts without extensive modelling knowledge. The simple and understandable processes address the guidelines of accuracy, relevance, clarity, and systematic structure. Fourth, by using attributes, issues such as weak spots or suggestions for improvement can already be deposited to the respective models during the as-is modelling. This has several advantages. On the one hand, domain experts are actively involved in the to-be modelling, which promotes later identification with the renewed processes. On the other hand, the first approaches are directly visible and can be weighted according to feasibility. For example, 'quick wins' can be implemented directly, or improvement suggestions can be prioritised. The simple identification and correction of flaws address the guideline of efficiency.

Besides, business processes need to be re-worked to transform a business digitally. This includes rethinking existing processes, introducing new processes, and questioning decisions made in the past (Schallmo and Rusnjak 2017). At Møblar's, most of the process steps identified as potentials for improvement during the selection of the new ERP system are part of mandatory changes. In the future, manual process steps bypassing the systems must be avoided as far as possible to exploit the capabilities of the new ERP system. But the scope of digital transformation goes beyond business process re-engineering. Business network redesign and business scope redefinition were also encountered, which are considered as the highest levels of IT-enabled business transformation (Venkatraman 1994). For example, in the past, the legally independent stores of Møblar's acted relatively isolated from each other. The company's digital transformation led the company to consider redesigning its business network of shops. In the future, the shops will be closely linked with each other such that, for example, vouchers can be bought and redeemed anywhere (online and offline), or customers can look up the availability of articles in any of the shops. Besides, the business scope has been redefined as new services of omni-channel management like click & collect and ship to home will be introduced once the implementation of the new ERP system has been completed.

13.5 Conclusion

The challenge introduced in this chapter is to set up rigorous BPM with the help of the 'icebricks' method. The objective is to show the applicability of 'icebricks' in the context of BPM to drive the digital transformation of an SME. In the context of the introduced case study, the objective was to select and implement a new ERP and online shop system, implement system interfaces that support omni-channel management, and introduce selected omni-channel concepts. To achieve this objective, 'icebricks' is actively used in the first three steps of the BPM lifecycle, Process discovery, Process analysis, and Process redesign. By applying 'icebricks' in the first three phases of the BPM lifecycle, the case study demonstrates the applicability of 'icebricks' to both the as-is and to-be process models. The models created in this way are persuasive due to their ease of use, easy-to-understand processes, simple weak spot identification, and applicability for everyone. With the help of the processes modelled in 'icebricks', a software selection was subsequently carried out. The implementation of the ERP solution is outside the scope of this work. However, it builds on the knowledge gained through 'icebricks'.

References

Barann, B., Hermann, A., Cordes, A.-K., Chasin, F. and Becker, J. (2019). 'Supporting Digital Transformation in Small and Medium-sized Enterprises: A Procedure Model Involving Publicly Funded Support Units'. In: pp. 4977–4986.

Becker, J., Riehle, D. M. and Clever, N. (2016). 'Ansätze zur Unternehmensmodellierung: Eine Einordnung'. In: *Geschäftsprozessorientierte Systementwicklung*. Springer, pp. 415–425.

Becker, J., Rosemann, M. and Schütte, R. (1995). 'Grundsätze ordnungsmäßiger Modellierung'. In: *Wirtschaftsinformatik* 37 (5), pp. 435–445.

Becker, J. and Schütte, R. (2004). *Handelsinformationssysteme.* MI Wirtschaftsbuch.

Clever, N. (2016). 'icebricks. Konstruktion und Anwendung eines Prozessmodellierungswerkzeugs'. Westfälische Wilhelms-Universität Münster.

Delfmann, P., Herwig, S. and Lis, L. (2009). 'Unified Enterprise Knowledge Representation With Conceptual Models-Capturing Corporate Language in Naming Conventions'. In: pp. 1–16.

Dumas, M., Rosa, M. L., Mendling, J. and Reijers, H. A. (2018). *Fundamental Business Process Management.* 2nd ed. Springer.

Fletcher, G. and Griffiths, M. (Dec. 2020). 'Digital transformation during a lockdown'. In: *International Journal of Information Management* 55, p. 102185.

Hammer, M. (2015). *What is Business Process Management?* Ed. by M. vom Brocke Jan and Rosemann.

Ko, R. K. L., Lee, S. S. G. and Lee, E. W. (Jan. 2009). 'Business Process Management (BPM) Standards: A Survey'. In: *Business Process Management Journal* 15 (5), pp. 744–791.

Mendling, J., Reijers, H. A. and Aalst, W. M. van der (2010). 'Seven Process Modeling Guidelines (7PMG)'. In: *Information and Software Technology* 52 (2), pp. 127–136.

Schallmo, D. and Rusnjak, A. (2017). 'Roadmap zur Digitalen Transformation von Geschäftsmodellen'. In: *Digitale Transformation von Geschäftsmodellen.* Ed. by D. Schallmo, A. Rusnjak, J. Anzengruber, T. Werani and M. Jünger. Springer Fachmedien Wiesbaden, pp. 1–31.

Siau, K., Erickson, J. and Lee, L. (2005). 'Theoretical vs. Practical Complexity: The Case of UML'. In: *Journal of Database Management (JDM)* 16 (3), pp. 40–57.

van der Aalst, W. (Jan. 2004). 'Business Process Management: A Personal View'. In: *Business Process Management Journal* 10 (2), pp. 1–6.

Van Der Aalst, W., La Rosa, M. and Santoro, F. (2016). 'Business Process Management'. In: *Business & Information Systems Engineering* 58 (1), pp. 1–6.

Venkatraman, N. (1994). 'IT-Enabled Business Transformation: From Automation to Business Scope Redefinition'. In: *Sloan Management Review* (35), pp. 73–87.

Verhoef, P. C., Kannan, P. K. and Inman, J. J. (2015). 'From Multi-Channel Retailing to Omni-Channel Retailing: Introduction to the Special Issue on Multi-Channel Retailing'. In: *Journal of Retailing* 91 (2), pp. 174–181.

Vering, O. (2002). *Methodische Softwareauswahl im Handel: ein Referenz-Vorgehensmodell zur Auswahl standardisierter Warenwirtschaftssysteme*. Logos-Verlag.

zur Muehlen, M. and Recker, J. (2008). 'How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation'. In: Springer, pp. 465–479.

Chapter 14

Towards Reference Models with Conformance Relations for Structure

Marco Konersmann, Judith Michael and Bernhard Rumpe

The term ‘reference model’ is broadly used in publications and discussions. To our understanding, the role of reference models and their use in modeling is not standardized and understood mostly informally in many communities. We posit that a more concise and formal understanding of the concept of a reference model is needed, to help make tool-assisted use of reference models and their relation to concrete models. In this contribution, we present our understanding of reference models and conformance relations, which specify which models are valid concretizations of a reference model. We argue why no general conformance relation can exist for all modeling languages. On the example of data models using UML class diagrams, we discuss potential recurring requirements of conformance relations, representations for mappings between concrete models and reference models, and the challenge of given code implementations to be attached to reference models and how to transfer this code to a concretization.

14.1 Introduction

In existing publications, there seems to be no agreement on what the term ‘reference model’ explicitly means. Authors define ‘technical reference models’ (Joshi and Michel 2008), ‘business reference models’ (Eom and Fountain 2013; Frank and Strecker 2007; Frank 2007), ‘business process reference models’ (Scheer 1994; Fettke et al. 2006; Reinhartz-Berger et al. 2010), the ‘ISO/OSI 7 layer communication reference model’ (American National Standards Institute 1981), ‘enterprise architecture reference models’ (Zimmermann et al. 2015), ‘performance reference models’ (Forme et al. 2007), ‘reference models for deep learning frameworks’ (Atouani et al. 2021), ‘reference models for digital shadows’ (Michael et al. 2023), and various others. Occasionally meta-models (Bucaioni et al. 2022; Michael et al. 2023; Mayr et al. 2018), meta-models and OCL (van der Aalst and Kumar 2001), or component-connector architecture models (Dalibor et al. 2022) are used and treated as reference models.

Gray and Rumpe raise the question of whether it is possible to leverage the notion of a reference model. In particular: ‘How can the notion of a reference model, explicitly denoted in a given modeling language, be formalized together with a precisely defined and tool-assisted notion of a conformance relation that would enable concrete realizations of the reference model?’ (Gray and Rumpe 2021). In this contribution, we present a formalization of reference models for structure data and their conformance relations along with operations upon them for tool-assisted conformance checking and for detecting mappings between concrete models and their reference models.

We consider a reference model to be a model, according to our understanding of the general definition of 'model' (which is based on Stachowiak 1973): (1) there should be an original, real system/entity that is being modeled, (2) the model should be an abstraction of the original, and (3) the model has something to do with the original system. Based on this definition, we draw the following consequences for reference models, which will be formally defined in Definition 1:

- A model can be used as a substitution for the real system: The *Principle of Substitution* then applies to reference models as well. It states that a useful subset of questions about the original can be answered by querying the model. This seems to fit the definition of a reference model because such a model should be a reference for many (similar) systems, even though there is typically a concretization development step between the definition of the reference model and the finalization of the system.
- A model is usually defined *explicitly in an appropriate modeling language.* For example, if we design a reference model for some data standard, we can use class diagrams as a language for its expression.
- It is often feasible to *use a similar or the same language for both reference and concrete models*, even though sometimes specific language constructs may only be used in the reference model, respectively, the concrete model. This may be a restriction for the general notion of the reference model because, in principle, it might be possible to actually use different modeling languages for reference models and their concrete realizations, but we ignore this possibility for now.
- It is often *not easy to distinguish a reference model from the concrete realizations* that are possible. It may be that the same model may serve as a concrete model or as a reference model in different contexts.
- Given a concrete model that is used for realization within a concrete system, it must be clearly decidable whether and how far the *concrete model actually conforms to the reference model.* A somewhat unclear conformance relation is not necessarily a problem when the reference model serves education, communication, or an otherwise informal purpose. However, when there is tool assistance available or needed, the question of whether a concrete model conforms to its reference becomes more critical. When the reference model is defined from a legal or standard context, the conformance question also becomes relevant for certification.

While the word 'reference model' is used quite frequently, to our understanding, many domains and their communities have different and mostly informal understandings of the significance of a reference model and, consequently, different approaches to using reference models in a particular domain (see also Arora et al. 2022). We posit that a more concise and formal understanding of the concept of a reference model is needed that improves the tool-assisted relationship between a general 'reference model' and the set of concrete realizations that conform to that reference model.

The remainder of this contribution is structured as follows: The next section introduces a running example of structure reference models. We present and discuss our understanding of structure reference models and their conformance relations in Section 14.3 and their consequences on operations upon models and reference models, including conformance checking in Section 14.4. Alternatives on representing mappings between reference models and concrete models are shown in Section 14.5. In Section 14.6, we discuss the challenge of given code implementations to be attached to reference models and how to transfer this code to a concretization. We show related work in Section 14.7 before we conclude in Section 14.8.

14.2 Example Reference Model for Structure Data Models

We illustrate a possible conformance relation for class diagrams as known from the UML (Rumpe 2017) using an example. A class diagram includes concepts for classes, interfaces, enumerations, attributes, methods, constants, and binary associations and compositions of various forms. In the following listing, we show a class diagram in textual form, i. e., a model of MontiCore's CD4A language[1] (Rumpe 2017), which we regard as a reference model:

```
classdiagram AccessRights {
        class User;
        class Role;
        class Right;
        class Create extends Right;
        class Update extends Right;
        association userroles User -> Role [*];
        association Role -> Right [*];
}
```

The next listing shows a concrete model example where we want to make statements about conformance.

```
classdiagram Flensburg {
        class Person;
        class Admin;
        class CarOwner;
        association Person -> Admin [0..1];
        association myCars   Person -> CarOwner [*];
        class Permission;
        class CreateCar extends Permission;
        class UpdateCar extends Permission;
        association allowed [1] CarOwner <-> Permission [1..3];
}
```

The concrete model can be considered conforming with the reference model if we consider a mapping of concrete model elements to reference model elements as informally shown in Listing 14.1. We call this mapping an *incarnation mapping*, because it describes which concrete model elements (on the left side) *incarnate* which reference model element on the right side. Here, we use the term 'incarnation' in contrast to instantiation, to highlight a specific difference in its characteristics: incarnation relates models elements of the same kind, here classes with classes and associations with associations, while instantiation normally relates different kinds, e. g., objects with classes.

1 https://github.com/MontiCore/cd4analysis

Listing 14.1: A mapping that renders the concrete model conformant to the reference model (informal)

```
// Mapping of classes
Person                        ===>  User
Admin, CarOwner               ===>  Role
Permission                    ===>  Right
CreateCar                     ===>  Create
UpdateCar                     ===>  Update
// Mapping of associations
(Person -> Admin), myCars     ===>  userroles
allowed                       ===>  (Role -> Right)
```

To decide whether the concrete model is a valid concretization of the reference model, we need rules that describe a valid *conformance relation.* In our example, these conformance rules state that each reference model element must be incarnated at least once, and the cardinality of reference associations may be restricted in the concrete model, but not relaxed (we will discuss further rules in Section 14.3.4). The information provided by the reference model, the concrete model, the incarnation mapping, and the conformance relation can now be used to validate whether the concrete model is a valid concretization of the reference model. In our example, the mapping can be considered valid with respect to the conformance rules, denoting it a valid concretization.

14.3 Reference Model, Conformance, and Incarnation

We introduce a formal definition for structure reference models, their conformance relations, and their constructive realization, namely the incarnation mapping. For that purpose, we also introduce a set of conformance rules.

14.3.1 Reference Model

Definition 1 (Reference Model). *A reference model in systems, enterprise, and software engineering is*

- *a model in a given language,*
- *used to describe a set of domain concepts and their domain-specific relations, and*
- *there exists a conformance relation, that precisely identifies the set of conforming models within the language the reference model belongs to.*

As a consequence of Definition 1, each reference model is also regarded as normal (concrete) model. The main difference between a reference model and a concrete model is, therefore, not a syntactical issue but based on the purpose of the model—or to be more precise, on the multiple purposes in the various activities of a development process. We identify these main purposes for reference models, where not all purposes must be addressed with each model:

P1 The introduction of *domain concepts* (which itself could be regarded as a fuzzy term). We assume that a domain concept has a name and potentially a number of properties that, together, can somehow be represented in our modeling language. These domain concepts are (and this is again somewhat informal) of general interest and, thus, typically can also be regarded as elements of a glossary or an ontology of a certain domain (Mayr and Thalheim 2020).

P2 *Domain-specific relations* connect these domain concepts. These relations are also described in the reference model.

P3 The *reference model itself* usually also has a name and, thus, allows us to talk about the overall system structure by the means of the reference model.

P4 There may be *additional information* or *methodical development advice* available that fits the domain concepts and, thus, can be manually reused in a development process.

P5 There may be a predefined *reference implementation* available that fits the domain concepts and, thus, can be automatically reused directly or mapped to the concrete model and its implementation via smart tooling.

P1 and P2 are usually the case in data models describing the application data of a domain, such as `Person` or `Account`, and their relation, like `owns`. P3 is, e. g., the case in the ISO/OSI 7 layer communication model or in various design patterns. Another example is a traffic light model with two states, `green` and `red`, and the corresponding transitions between those states as a reference model. The latter serves all three purposes (P1–3) if refinable, e. g., to traffic lights with the third state `yellow`.

Using reference models, however, becomes really interesting if P4 or even P5 can be leveraged. In adequate tooling, for example, the concept `User` of our example in the first listing may be equipped with possibilities for a user to administrate her own setting, acquire roles, or receive rights to read/write data. It depends on the conformance relation and the tooling that (a) checks conformance and (b) has smart concepts on how to transfer predefined functionality from the reference model to the concrete model, respectively, its implementation.

The purposes may arise independently from the model itself. Therefore, any model can in principle be regarded as a reference model when deemed necessary or helpful. With this respect, we define the set of reference models as a subset of the set of models (see Definition 1). The sets of reference models and concrete models of a modeling language are not disjoint but may have a large overlap. However, they don't necessarily have to be identical. There may be models that only can be used as reference models, e. g., because certain abstract concepts in the models such as abstract classes do not have concretizations yet.

14.3.2 Conformance Relation

Definition 2 (Conformance Relation). *Let $\mathbb{M}$ be the set of all concrete models, and $\mathbb{R} \subseteq \mathbb{M}$ the set of all reference models. The conformance relation*

$$C \subseteq \mathbb{M} \times \mathbb{R}$$

is a binary, reflexive, and transitive relation that describes whether the concrete model $M \in \mathbb{M}$ is a concretization of the reference model $R \in \mathbb{R}$.

The conformance relation (see Definition 2) is defined as a binary relation between models. It includes the reference model – describing relevant concepts and relations – and its implementation, respectively, the concrete model. This definition has some consequences for conformance relations: First of all, the conformance relation generally relates more abstract models with more concrete models. Because the reference model and concrete model belong to the same language (see the consequences in Section 14.1), we may accept that a reference model can also be a concrete implementation, which means conformance is *reflexive*. The conformance relation can also be *transitively* applied, which can become interesting when concrete models become reference models themselves or when reference models for other reference models are created. It is, however, *not symmetric*, but it relates the more abstract to the more concrete models. It may be that it is not fully asymmetric either, but we currently see no application scenario for this. In general, different conformance relations can exist for different reference models. And finally: multiple conformance relations can exist for a reference model.

In a development project that uses a reference model, the conformance relation is potentially a development artifact on its own. Please note that we do not enforce that a reference model describes a structure, such as a data structure or a communication infrastructure. We can perfectly understand that people use Statecharts, BPMN models, Petri Nets, or activity diagrams as reference models. Furthermore, it may well be that a domain-specific language created for a specific purpose also comes along with the construct of reference models within the language.

However, we made the following observations with respect to the languages that we analyzed: The construction of reference models, and, in particular, the conformance relation needed, is highly driven by the semantics of the particular language. Thus, a general, purely syntax-based conformance relation will not be sufficient. As a consequence, conformance relations need to be defined for modeling languages individually.

Because the conformance relation is an abstract relation between models, it is necessary to refine this relation into a relation between individual elements of the concrete and the reference model. We call such a detailed relation a *incarnation mapping*.

14.3.3 Incarnation Mapping

Incarnation mappings describe which concrete model elements map to which reference model elements, as shown in the example in Listing 14.1. Therefore, an incarnation mapping looks into the concrete syntax of the models to be related and by definition is highly dependent on the modeling language. Without precisely clarifying the concept of ‘model element’ in this contribution, we assume that for a given model M, we can denote the set of relevant, syntactic model elements by M_e. We define incarnation mappings as follows:

Definition 3 (Incarnation Mapping). *For a given concrete model M and reference model R an incarnation mapping is a binary relation between their relevant, model elements*

$$I(M,R) \subseteq M_e \times R_e$$

such that the existence of an incarnation mapping ensures the conformance of the models, i. e., $C(M,R) \iff \exists i \in I \,:\, i(M,R)$.

The definition uses the term ”relevant model elements”, without precisely defining what is relevant. This is deferred to concrete applications. In our class diagram example from Section 14.2, we can identify classes and associations as relevant (even though other options would be possible).

A well-formed incarnation mapping identifies a conformance relation. To develop a concrete model that conforms to a reference model, the task is thus to establish a well-formed incarnation mapping.

The relation between concrete model elements and reference model elements is closely related to instantiation. We use the term ‘incarnation’ as in Definition 3 to highlight specific differences in its characteristics: (1) Incarnation relates models elements of the same kind, i. e., in our class diagram examples it relates classes with classes and associations with associations, while instantiation normally relates different kinds, e. g.,, objects with classes. (2) Incarnation is transitive and reflexive on the model elements (similar to the conformance relation on models itself), while instantiation is not. In general, multiple incarnation mappings can be possible for any non-trivial concrete and reference model. This means that the same reference structure is being applied several times in a concrete model in either completely disjoint or potentially overlapping substructures.

As an example, the next listing shows a reference model for user ids. The corresponding concrete model shown below describes a data structure where a `Person` has exactly one personal id and one tax id, which allows for two different incarnation mappings using two different ids from the concrete model.

```
classdiagram UserID {
        class User;
        class ID;
        association id User -> ID [1];
}
classdiagram TaxablePersonell {
        class Person;
        class PersonalID;
        class TaxID;
        association pId Person -> PersonalID [1];
        association tId Person -> TaxID [*];
}
```

In this example, the `Person` has multiple IDs. There are two mapping relations as shown in the first two mappings given in Listing 14.2. The first and second sets of mappings denote that the `Person` is mapped to the class `User` once in the personal ID mapping and once in the tax ID mapping. This allows for checking these sets individually. Following the conformance rules stated in the example of Section 14.2, the personal ID mapping is valid. The tax ID mapping is, however, to be regarded as invalid, because the association `tId` is relaxed in the concrete model, which contradicts a (below discussed and rather useful) rule that associations must retain or restrict their cardinalities, but not relax them.

Listing 14.2: Multiple incarnation mappings (informal)

```
// First Incarnation Mapping (Personal ID Mapping) - valid
Person      ===> User
PersonalID  ===> ID
pId         ===> id

// Second Incarnation Mapping (Tax ID Mapping) - invalid
Person      ===> User
TaxID       ===> ID
tId         ===> id // invalid
```

Incarnation mappings may be of complex forms, and it could well be that there is no generalized form for all purposes. However, we expect at least the following property to hold: the correctness of an incarnation mapping and thus of a conformance relation between concrete and reference model can be effectively identified and checked by an algorithm. However, there are two fundamentally different relevant cases, (a) the mapping is already and explicitly given as some kind of parameter of the conformance relation, vs. (b) the mapping must be constructed. Case (b) is definitely more complex and potentially ambiguous, but would be of strong interest for developers.

14.3.4 Conformance Rules on the Class Diagram Case

We can argue that the model `Flensburg` in our running example is a conforming implementation of the reference model `AccessRights` with respect to the incarnation mapping and the two rules stated in Section 14.2. The rules stated in the example are simplified for illustrative purposes. In this section, we discuss rules that specify conformance relations for structure reference models. For clarification, we talk about *concrete classes C* and *concrete associations A* when referring to the *concrete model* and *reference classes R*, and *reference association S* when referring to the *reference model.* We found the following set of rules helpful in the case of UML class diagrams that we typically use:

R1 A concrete class C incarnates (at most) one reference class R (e. g., `Person` incarnates `User`). Thus, it is possible that additional classes exist in the concrete model.

R2 The same reference class R may be incarnated by many concrete classes (e. g., `Role` by `Admin` and `CarOwner`), but at least one must be given.

R3 A concrete association A incarnates (at most) one reference association S (e. g., `allowed` incarnates `(Role -> Rights)`). Again it is possible that additional associations exist in the concrete model.

R4 In analogy to classes, the same reference association S may be incarnated by many concrete associations (e. g., `userroles` has two independent incarnations, `(Person -> Admin) and myCars`), but at least one must be given.

R5 Subclassing relations are preserved, i. e., the concrete class $C2$ has to extend $C1$ if their reference classes $R2$ extends $R1$. Class $C2$ does not necessarily need to be a direct subclass but can be found in the subclassing hierarchy of $C1$.

R6 Reference associations are preserved. This means that given concrete classes $C1$ and $C2$, their reference classes $R1$ and $R2$, and a reference association $S : R1 \rightarrow R2$, then there must also be a concrete association $A : C1 \rightarrow C2$, such that A incarnates S.

These rules are definitely incomplete, because all practical and complete conformance relations also need to take typing and cardinalities of associations into account. Furthermore, real class diagrams also contain attributes, methods, abstract classes and interfaces, as well as a sophisticated type system, including enumerations.

In our example, all concrete classes have respective concrete associations attached except the `Admin`, which violates condition R6. We could argue that the admin doesn't need explicit permissions because, in our application, she has all permissions anyway, but it does violate the conformance rule.

As said earlier, in different mappings, it is allowed that one concrete class incarnates multiple reference classes, therefore, playing different 'roles' with respect to the reference model. As an example, consider the observer pattern as reference model (Listing 14.3) and a class diagram where two classes are mutual observers (Listing 14.4), with the incarnation mapping shown in Listing 14.5.

Listing 14.3: Simplified reference model for the observer pattern (as CD4A Model)

```
classdiagram ObserverPattern {
        class Observer;
        class Observed;
        association observes Observer -> Observes [*];
}
```

Listing 14.4: Concrete model of mutually observing classes (as CD4A Model)

```
classdiagram MutualObservers {
        class Attacker;
        class Defender;
        association watches Attacker <-> Defender [*];
}
```

Listing 14.5: Mapping mutual observers to the observer pattern (informal)

```
// Incarnation Mapping 1
Attacker ===> Observer
Defender ===> Observed
watches  ===> observes

// Incarnation Mapping 2
Defender ===> Observer
Attacker ===> Observed
watches  ===> observes
```

In specific projects or situations, the rules for conformance relations and their incarnation mappings might be more complex than stated above. Several properties can potentially be relaxed or restricted. Let us give several examples:

- The reference model uses an attribute `long ident` as an internal identification mechanism. It may be allowed to incarnate that attribute by `String ident`, because strings also have a linear order or by `int id` because we know that the reduced range is sufficient. In case a framework implementation, however, already relies on the data type `long`, such a change would be illegal.
- The reference model uses a certain cardinality for association $S : C1 \rightarrow C2$ `[card]`. Cardinality can, in principle, be restricted because we know from the domain that certain restrictions are okay, or extensions, e. g., from `[1]` to `[*]`, because that is needed in the domain. It may be that both changes are okay, or neither of them.
- The reference model has a direct subclass relationship between classes R_1 and subclass R_2. In the concrete model, there may be many incarnations of R_2 arranged in a deep hierarchy.
- Furthermore, it may be that adding additional attributes or associations is restricted because there are predefined implementations (or predefined generative techniques) of certain reference classes.

Potentially, there are many more details to be handled, such as freely chooseable independent names or attribute types, how to deal with enumerations, what are valid or necessary incarnations of interfaces and abstract classes, etc. A specifically interesting question is, for example, whether a reference attribute or a reference association can be incarnated in the concrete model by a derived attribute or a composed association. In both cases, the realization is hidden in the combination of data structure and algorithmic functionality, such as getters with calculation bodies for derived attributes or an OCL-like logic formula describing how to derive a composed association from basic ones.

To summarize, we argue that there is no such thing as one single conformance relation even for class diagrams, but there may be a core, as shown in the requirements R1 to R6, plus a number of additional or optional relaxations. These rules define the conformance relation and therefore need to be processable by tools.

When looking at other kinds of languages, for example, state charts, the very same observations apply. There we have to map states, transitions, nesting of states, preconditions, entry and exit actions, etc. Preconditions may, for example, be relaxed or strengthened, or alternative transitions in nondeterministic state charts may be removed. A simulation relation for state charts or, more precisely, a refinement calculus like in Rumpe (1996) or Paech and Rumpe (1994) could be applied here. Especially challenging is the question of whether a reference transition may be realized by a sequence of concrete transitions and what happens if the sequence can be interrupted in the middle.

14.4 Operations upon Models and Reference Models

Having a concise and formal definition of the concept of a reference model as shown in Section 14.3 allows us to define functions for automating the handling of the relation between models and reference models. We can:

1. Check the conformance of models with respect to a reference model and
2. Constructively find valid mappings of a model to a reference model.

14.4.1 Conformance Checking

Definition 4 (Conformance Checking). *A conformance check is defined as a function*

$$checkConformance(M, I, R, C)$$

which defines whether the concrete model M conforms to the reference model R with respect to the given incarnation mapping I and the conformance rules C.

Conformance checking (see Definition 4) means to validate whether a model conforms to a reference model. Here we have to take the incarnation mapping into account and check it against the conformance rules of the conformance relation. The result of the checking in general is the statement whether the given model is a valid concretization of the reference model, taking as input the concrete model, the reference model, and the incarnation mapping. With the definition of incarnation mappings and conformance relations we can identify which mapping pair of model elements in M_e and R_e violates which rule of the conformance relation and regard this as additional, explaining output.
The implementation of this function depends on the rules defined for the conformance relation. Only few rules can be checked by taking only the incarnation mapping into account, but many rules also have to take the model and reference model into account. For example, rules R5 and R6 require knowledge about the inheritance relationships and associations

in both models. We consider graph matching algorithms (Livi and Rizzi 2012) a suitable implementation strategy for implementing conformance checking. This allows for a formal specification of rules in terms of graph pattern matching. When the conformance check requires information about the semantics, this must be encoded in the graph matching rules. Efficient implementations of graph matching exist for various modeling frameworks, e. g., QVT (Kurtev 2007), ATL (Jouault et al. 2006), or Henshin (Strüber et al. 2017) for the Eclipse Modeling Framework or MontiTrans (Hölldobler 2018) for MontiCore (Hölldobler et al. 2021).

Semantic differencing (Nachmann et al. 2022; Drave et al. 2019; Maoz et al. 2011) is an advanced analysis technique to understand the differences between two class diagrams on the basis of the actual object structures that are allowed in one but not the other model. This kind of semantic difference uses identical class, attribute, and association names to identify comparable classes. Semantic differencing, however, can be extended to reference models and their incarnations to understand the differences between the concrete realization and the original reference model when incorporating the conformance relation into the semantic difference calculation.

14.4.2 Incarnation Mapping Detection

Definition 5 (Incarnation Mapping Detection). *Detecting one or many incarnation mappings of a model M and a reference model R is defined as a function*

$$\mathit{findMappings}(M, R, C) = \{I_1, I_2, ..., I_n\}$$

that detects a set of valid incarnation mappings for a model M to a reference model R with respect to a conformance relation C.

Incarnation mapping detection (see Definition 5) means to automatically find valid incarnation mappings between a model and a reference model, given a conformance relation. There is not always exactly one valid incarnation mapping as a result, but the resulting set can contain zero, one, or many mappings. However, the search space as well as the result is finite, because the number of model elements M_e and R_e is finite.

Finding valid incarnation mappings can be computationally expensive, e. g., if all potential mappings between all model and reference model elements are checked against the conformance rules. We see some approaches for decreasing this computational complexity. (1) We can use additional knowledge to reduce the search space. If a model uses a known naming convention related to the reference model, this can be exploited. For example, in Listing 14.1, we can see that the incarnations of `Create` and `Update` all start with the respective name from the reference model elements. (2) We can produce a model resembling the target model and the corresponding mapping by applying the conformance rules constructively, as it is known from the generation of trace graphs in triple graph grammars (Schürr 1995). For example, consider the reference model in Section 14.2 given by the reference model `AccessRights`, the model `Flensburg`, and the conformance rules from Section 14.3.4.
We can produce a mapping by first applying rule R1, and let the concrete class `Person` incarnate the reference class `User` (correctly), and the concrete class `CarOwner` incarnate the reference class `Role` (correctly). Now we can apply the rules R6 together with R1 and R3, for the association `myCars` from `Person` to `CarOwner` to correctly incarnate `userroles` from `User` to `Role`. If we instead tried to produce a mapping where we let the concrete class `Person`

incarnate `Role` (instead of `User`), and the concrete class `CarOwner` incarnate the reference class `Role` (correctly), we also have to find an association that incarnates `Role -> Right [*]` according to rule R6. But there is none in the concrete model. The only remaining association to build from the `Person` is defined as `Person -> CarOwner`, and `CarOwner` does not incarnate `Right`. This incarnation mapping is not possible with respect to the rule R6. Further approaches to efficiently implement an incarnation mapping identification may exist and are subject to research.

It should be noted, that also a partial incarnation mapping can be helpful. A result of an incarnation mapping identification might also be a partial mapping, stating that the model does not conform to the reference model, but there are some incarnation mappings that come close. As an example, consider a scenario where a novel software architecture reference model for a specific domain has been developed based on existing software architecture models in practice. The developers of the reference model want to know to which degree existing architecture models conform to the novel reference architecture model. Partial mappings can now show for each individual architecture model how close it is to a valid concretization of the novel reference model, and consequently whether and where the reference model or the concrete model should be adapted.

Such partial mappings can also be used as a basis for repair techniques, like defined in Kautz and Rumpe (2018), which is the identification of necessary modifications to the concrete model to establish a complete realization of the reference model.

14.5 Representations of the Incarnation Mapping

An interesting question is what an explicit representation of an incarnation mapping looks like, if needed. Above, we have given an informal list to describe this mapping. The list contains explicitly given names for classes and associations but also maps unnamed associations by using the `(C1 -> C2)` notation. We actually see several alternatives for representing such mappings:

A1 An explicit mapping language similar to the artifact shown in Listing 14.1 can be used.

A2 The mapping can be extracted from the two models by using naming conventions.

A3 The concrete model explicitly refers to the reference model by using the domain concepts introduced in the reference model as stereotypes, labels, or flags.

A4 The concrete model is actually an extension of the reference model, which means that all model elements (here, classes and associations) defined in the reference model are explicitly included and therefore used in the concrete model.

Alternative A3 can be seen quite commonly when generators are used. For example, consider the example in Listing 14.6. Here a lot of stereotypes pollute the otherwise clean concrete model. Using a number of conventions, it may be that certain labels have not been defined explicitly but can be reconstructed. This is regularly the case for a concrete association when both its concrete classes have been stereotyped.

Another possibility is to combine the explicit labeling with the implicit use of naming conventions, as described in Alternative A2 such as (1) same name or (2) a shared prefix (see `UpdateCar`) or analogously a shared postfix (like `StopState`). This, of course, shares the problem of unwanted, coincidental matches but largely reduces the amount of labeling necessary. This is a common approach when applying design patterns, which we also regard as a special form of reference models denoted as class diagrams, e. g., in Gamma et al. (1995), where often the roles of the individually participating classes are used as parts of the actual class names.

Listing 14.6: Concrete model using stereotypes (as CD4A Model)

```
classdiagram <<labels:"AccessRights">> Flensburg {
        class <<User>> Person;
        class <<Role>> Admin;
        class <<Role>> CarOwner;
        association <<userroles>> Person -> Admin [0..1];
        association <<userroles>> myCars   Person -> CarOwner [*];
        class <<Right>> Permission;
        class <<Create>> CreateCar extends Permission;
        class <<Update>> UpdateCar extends Permission;
        association allowed [1] CarOwner <-> Permission [1..3];
}
```

To prevent potential pollution with stereotypes from various technological spaces, we have explored the possibility of using an extra tagging language that allows labeling model elements without invasively changing the original model stored in extra artifacts in Greifenberg et al. (2015). Tagging models can be applied in various domains (Dalibor et al. 2019; Maoz et al. 2016).

Alternative A4 is quite common, especially if the reference model has a framework implementation, where the classes of the reference model are already and explicitly implemented under the very same names. Using different names is then not a possibility. The concrete model would then have to be specified as shown in Listing 14.7.

Listing 14.7: Concrete model as extension (as CD4A model)

```
classdiagram Flensburg extends AccessRights {
        class Person extends User;
        class Admin extends Role;
        class CarOwner extends Role;
        class Permission extends Right;
        class CreateCar extends Permission, Create;
        class UpdateCar extends Permission, Update;
}
```

This model is relatively compact because associations need not be mentioned explicitly again but are inherited from AccessRights. However, this approach does neither allow for adapting cardinalities of associations nor building multiple incarnations of associations. It also has limitations with respect to multiple inheritance: In the example in Listing 14.7, the classes CreateCar and UpdateCar are modeled to extend the class Permission. In addition they now also extend their respective reference model classes. While this is possible in UML class diagrams, multiple inheritance is not permitted in many programming languages. It is, thus, ambiguous how to implement the classes. A similar approach could be to develop the concrete model independently at first and then apply a generalized merge algorithm, similar to the one provided by MontiCore's CD4A language, and merge the framework classes, associations, and, thus, infrastructure into the concrete model.

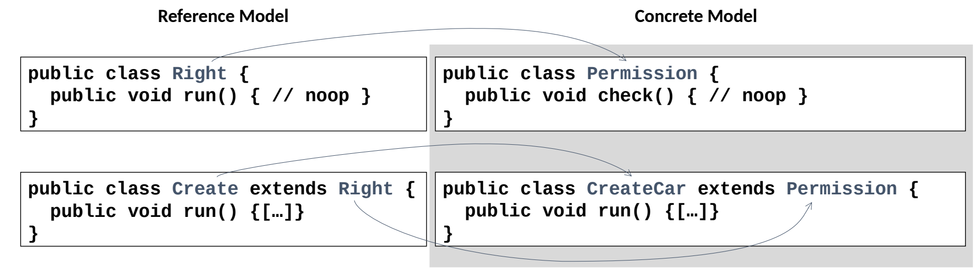
14.6 Mapping a Reference Implementation to a Concrete Implementation

The reuse of models and the use of reference models becomes particularly interesting, when not only the reference model and its particular design can be reused, but also a reference implementation exists, that already embodies a lot of technical details and which can also be reused. We therefore assume, that as described in Section 14.3, reference models may be accompanied by a reference implementation.

The really interesting part is, therefore, the transfer of an existing reference implementation, e. g., for `AccessRights` into the concrete implementation. Ideally, this is done as automatically as possible. If the reference implementation exists in the form of a methodological approach this must be done manually, but still is of great help. If it is predefined in a framework there are chances that the concrete implementation reuses that framework either through subclassing or through delegation. Even better would be a setting, where a generator, which would map the model to code anyway, directly adds all the technical details from the reference implementation, such as rights management, database storage, etc., into such a concrete implementation.

Smart tooling is required to transfer predefined code for the reference model to the concrete model, respectively, its implementation. This transfer technologically depends on the representation of the mapping (see Section 14.5). If an explicit mapping language is used (A1), code can be generated based on a reference implementation and an incarnation mapping model. Here it will be necessary to exchange the names of code elements during the generation, as the mapping can effectively be seen as a refactoring. The same applies when the mapping can be extracted by using naming conventions (A2) or stereotypes, labels, or flags (A3). If a concrete model extends the reference model (A4), there is also the opportunity to define a reference implementation in the extended classes (e. g., `AccessRights`, `User`, `Role`, and `Right` in Section 14.2). Figure 14.1 shows the renaming approach on the example of `AccessRights` to Listing 14.1. In this alternative, the code given by the reference model element is copied and refactored. The reference implementation itself is not part of the code for the concrete model.

The left side of the example shows excerpts of code for reference model elements, while the right side shows the refactored code for the concrete model elements. The class `Right` is



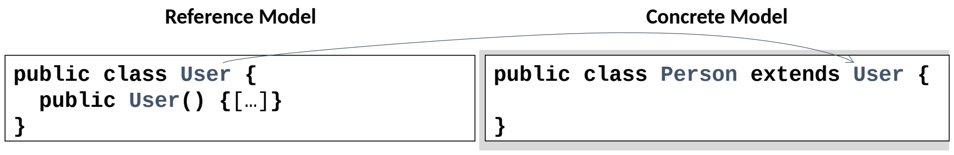
```
public class Right {
  public void run() { // noop }
}
```

```
public class Permission {
  public void check() { // noop }
}
```

```
public class Create extends Right {
  public void run() {[…]}
}
```

```
public class CreateCar extends Permission {
  public void run() {[…]}
}
```

Figure 14.1: Example of the concretization of given Java code for reference models via renaming



```
public class User {
  public User() {[…]}
}
```

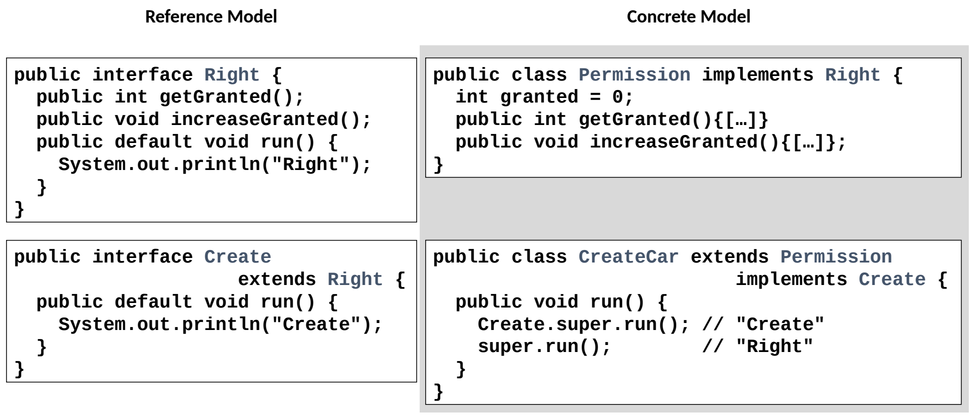
```
public class Person extends User {

}
```

Figure 14.2: Example of the concretization of given Java code for reference models via extension

code that implements the default functionality of the respective reference model element (upper left part of Figure 14.1). The default functionality is that it provides a public method `run`, that does nothing but can be overridden by subclasses. The code for the concrete model element is created by applying a renaming refactoring on a copy of that code, that replaces the reference model element name with the concrete model element name. The reference model element `Create` on the lower left extends the class `Right`. It overrides the method `run` to implement a default functionality for the right to create something. The lower right side shows the refactored code for the concrete model element `CreateCar`. Here we renamed the class name and the name of the extended class according to the example's incarnation mapping.
Figure 14.2 shows the extension approach for the reference model element `User` and its concretization `Person`. In contrast to the former alternative, here the code for the reference model element is part of the code for the concrete model. The reference implementation on the left side of Figure 14.2 provides a constructor. The code for the concrete model element extends that reference model class. Due to the inheritance the constructor of the class `Person` is resolved to the constructor of the reference model element class `User`, thus effectively providing a default implementation to the concrete model element class `Person`.

The extension mechanism has drawbacks due to the limitation to multiple inheritance in many programming languages. For example, the extension mechanism could not be used unchanged in this example for the specific rights `Create` and `Update`. One could expect that code for these reference model elements is implemented as Java classes that extend the class `Right`. With the extension mechanism for implementation, we thus define the two classes `CreateCar` and `UpdateCar`, which extend both classes `Create` or `Update` respectively due to the extension mechanism and `Permission` due to the inheritance relation in the concrete model. This multiple inheritance cannot be implemented in languages such as Java due to them not allowing multiple inheritance.

A workaround to this issue in Java could be to use Java interfaces with their default implementation functionality, because Java classes can implement an arbitrary number of interfaces. Figure 14.3 shows this workaround for the example above. Instead of classes, in this example we define Java interfaces for the reference model elements `Right` and `Create`. They provide their reference implementation via default methods in Java interfaces. To showcase the consequence, these default implementations print their class names. The concrete model elements `Permission` and `CreateCar` are defined as classes. Following the extension mechanism, `Permission` now implements the interface `Right`. `CreateCar` implements the interface `Create`, following the extension mechanism, and extends the class `Permission` as modeled in the concrete model. It is now not unambiguously resolvable, which method will be executed upon calling `CreateCar.run`. The Java compiler marks this



```
public interface Right {
  public int getGranted();
  public void increaseGranted();
  public default void run() {
    System.out.println("Right");
  }
}
```

```
public class Permission implements Right {
  int granted = 0;
  public int getGranted(){[…]}
  public void increaseGranted(){[…]};
}
```

```
public interface Create
                    extends Right {
  public default void run() {
    System.out.println("Create");
  }
}
```

```
public class CreateCar extends Permission
                           implements Create {
  public void run() {
    Create.super.run(); // "Create"
    super.run();        // "Right"
  }
}
```

Figure 14.3: Example of the concretization of given Java code for reference models via interfaces with default implementations

as error. Therefore, we override the method `run` and explicitly call the respective method. Conceptually, the class represents the functionality of the model class `CreateCar`, which incarnates the reference model element `Create`. Therefore, we expect that the functionality of `Create` should be called. In the example, we show how to execute either method for reference.

Default implementations for interfaces do not allow saving states between method calls. That is, interfaces cannot have attributes. Therefore, using interfaces with default implementations limits the code for reference model elements to operations without side effect. To showcase how to save state as a functionality with this method, we added a getter and setter method to the interface `Right` as functionality for the respective reference model element. This requires the concrete implementation `Permission` to handle these methods and provide means to save the state. Alternatively, a runtime could inject a context element as method parameter. Another issue with this alternative arises when there are multiple interfaces that implement the functionality of mapped reference classes, which declare the same methods. For example, consider that the class `CreateCar` incarnates multiple reference model classes, which all implement a method `run`. In that case the implementing class has to decide which method to execute, as shown in the example in the class `CreateCar`. If multiple of them are executed, it also has to decide on an order. Our example shows the drawback of the extension mechanism. Nevertheless, such libraries and frameworks are widespread in practice. This alternative has the advantage that the respective libraries can be efficiently distributed by the usual means (e. g., providing download links or providing them using repositories for dependency resolvers like Maven[2] for Java or pip[3] for Python) and dynamically linked. With the renaming alternative, there is no library to distribute, but the complete code needs to be generated and included in the resulting program code.

2 https://maven.apache.org/
3 https://pypi.org/project/pip/

14.7 Related Work

In a literature survey, Arora et al. (2022) identified that there is no universally accepted understanding or definition of reference models. They extracted nine qualities to characterize reference models: *reusable, flexible, reliable, designed systematically, generally valid, required, user-centered, comprehensive*, and *educative*. They define a reference model as 'a holistic collection of methodologies systematically structured in an architecture in which every element (e. g., guidelines, methods, procedures, and entities, ...) is made transparent and outlined as a generic solution based on both best practices and innovative approaches.' We take a similar, but more formal perspective on reference models. It is more focused on the formalization of the concepts of reference models and the relation between concrete models and reference models.

Reference models are subject to research in the domain of software architecture frequently. The ArCh approach by Herold et al. (2013) describes reference architectures and their constraints with an ontology and first-order logic formulas. Concrete models are extracted from models or source code and translated to first-order logic formulas. The correspondence check is then implemented by checking the reference architecture and constraint formulas against the concrete model formulas. Our approach uses the same language for reference models as it is used for concrete models, which is closer to the day-to-day development of software engineers.

Weinreich and Buchgeher (2014) describe reference architectures for service-oriented architectures (SOA) and automated conformance checking. In this approach, reference models are described using roles, relationships between roles, and logic-based constraints for those elements. Roles and their relations are defined using rules, which include rules for mapping concrete elements to reference model elements by properties. For example, a role can be mapped to all links between concrete services with the property 'local'. Conformance is checked by validating that the rules hold for the given SOA. In contrast, we explicitly distinguish the mapping from the reference model elements in the conceptualization and describe multiple alternatives to represent incarnation mappings.

Bucaioni et al. (2022) describe MORE, an MDE approach for modeling reference architecture models, checking the conformance of concrete architectures to reference architectures, and to ensure the compliance of concrete models by providing guidelines and rules. They describe a reference architecture meta model that can be used to describe reference architecture models. These models are then lifted to domain-specific meta models of which concrete architecture models can be constructed. A meta model can be used to define explicit mappings from concrete architectures to reference architectures. A conformance check can then validate whether the concrete model conforms to the reference model based on the relationship between meta models and their models. In this approach architecture models instantiate the domain specific meta model. Our definition allows for arbitrary model-based languages.

14.8 Conclusion

The term 'reference model' is broadly used publicly with a variety of meanings, as many different domains and their communities have different and mostly informal understandings of reference models. A more concise and formal understanding of the concept of reference models is needed to improve the tool-assisted relationship between a general reference model and the set of concrete realizations that conform to that reference model. In this contribution, we discussed our understanding of reference models and conformance relations, which specify which concrete models are valid with respect to the reference model. We demonstrated our understanding of conformance relations on structure reference models for data, represented as class diagrams from the UML. We argue that conformance relations must be defined for modeling languages individually because they are driven by the semantics of the particular language. Purely syntax-based conformance relations will not suffice.

To be able to validate whether concrete models conform to reference models it is required to specify a mapping between concrete model elements and reference model elements. We describe different ways to represent these mappings with different characteristics and argue that there is no generalized form that is fit for all purposes. We discuss the challenges of implementing conformance checking and the detection of mappings between reference model elements and concrete model elements. We also identify the attachment of an implementation or other additional information to reference models as a challenge and provide alternative solutions for automating the translation of such a reference implementation to a concrete implementation for concrete models. Further research is necessary towards the automation of conformance checking and for automatically detecting (partial) mappings, as well as smart tooling, e. g., to automatically develop valid concrete models from reference models as development support, and to transfer reference implementations to concrete implementations.

Acknowledgement

We would like to thank our colleagues Max Stachon and Sebastian Stüber for the valuable discussions on this topic.

In Honour of Ulrich Frank

We contribute to this anthology in honour of Ulrich Frank as we have hold him in high esteem for his professional expertise, his willingness to engage in discussion and the lively exchange of ideas for many years. Some of his research highlights which we appreciate very much are his work towards open reference models (Frank and Strecker 2007), the evaluation of reference models (Frank 2007), and his contribution to globalizing domain-specific languages (Cheng et al. 2015) together with the lively discussions in the related Dagstuhl seminar. His research on a method for designing domain-specific modeling languages (Frank 2010), inspired us to restructure our initial approach for a domain-specific modeling method to create languages for the ambient assistance domain (Michael and Mayr 2015). More recently, he worked on characteristics of low-code platforms (Bock and Frank 2021b; Bock and Frank 2021a), providing open research ideas and opportunities for future developments. This work inspired us to discuss adaptation mechanisms for data and process models within our work on low-code development platforms for digital twins (Dalibor et al. 2022).

As an active member of the conceptual modeling (ER conference) and model-driven software engineering (MODELS conference) community, he is a link between advocates of data modeling and those who have a stronger focus on software engineering and model-driven methods. Due to his interest in industry collaborations and keen sense of business challenges, we were not surprised by his research on a conceptual framework and modeling languages for multi-perspective enterprise modeling (MEMO) (Frank 2002) integrating different perspectives. For his current focus on multi-level modeling (Frank 2022; Frank 2014b, which, to cite him, 'has not yet made it to the research mainstream' (Frank 2022), the research community is showing an increased awareness of the problems it promises to solve.

Ulrich Frank is an active member of the German Informatics Society and especially the QFAM[4], a forum for modeling enthusiasts from different areas in Computer Science. He is an active participant in our regular modeling discussion rounds with Bernhard Thalheim and Heinrich C. Mayr introducing new perspectives. His broad expertise and interests make him an excellent organizer of PhD symposium on conferences, as he provides helpful feedback and inspires PhDs to rethink their research ideas. Back in 2012 in a lecture room full of new PhD candidates (Sinz and Schürr 2012), it was his statement and questions which influenced one of the co-authors of this article to rethink the evaluation (Mayr and Michael 2012; Michael and Mayr 2017) of the modeling language created in her PhD (Michael and Mayr 2013). His suggestions on increasing the value of research by promoting value to the researcher are timelessly current: He suggests getting outraged, relaxing and not taking the factual manifestations of academia or ourselves as researchers too seriously, enjoying research, being free and ambitious, to surprise your peers, and help to create a better world (Frank 2014a).

We would like to thank Ulrich Frank for all his very valuable contributions and his stimulating and discussion-promoting remarks and questions. Our experience has shown that it is worthwhile to listen actively to his remarks and to reflect on them in one's own research context. Thus, we are looking forward to all next conferences, workshops, and discussion sessions with a versatile modeling enthusiast with practical relevance.

References

Atouani, A., Kirchhof, J. C., Kusmenko, E. and Rumpe, B. (Oct. 2021). 'Artifact and Reference Models for Generative Machine Learning Frameworks and Build Systems'. In: *20th ACM SIGPLAN Int. Conf. on Generative Programming: Concepts and Experiences (GPCE 21).* Ed. by E. Tilevich and C. De Roover. ACM SIGPLAN, pp. 55–68.

American National Standards Institute (Apr. 1981). 'Data Processing-Open Systems Interconnection - Basic Reference Model'. In: *SIGCOMM Comput. Commun. Rev.* Vol. 11. 2. Association for Computing Machinery (ACM), pp. 15–65. DOI: `10.1145/1015586.1015589`.

Arora, S., Ceccolini, C. and Rabe, M. (2022). 'Approach to Reference Models for Building Performance Simulation'. In: *10th Int. Conf. on Model-Driven Engineering and Software Development, MODELSWARD 2022.* Ed. by L. F. Pires, S. Hammoudi and E. Seidewitz. SCITEPRESS, pp. 271–278. DOI: `10.5220/0010888800003119`.

Bock, A. C. and Frank, U. (2021a). 'In Search of the Essence of Low-Code: An Exploratory Study of Seven Development Platforms'. In: *2021 ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 57–66. DOI: `10.1109/MODELS-C53483.2021.00016`.

4 Querschnittsfachaussuss Modellierung der Gesellschaft für Informatik e.V. `https://qfam.gi.de/`

Bock, A. C. and Frank, U. (2021b). 'Low-Code Platform'. In: *Business & Information Systems Engineering* 63.6, pp. 733–740. DOI: 10.1007/s12599-021-00726-8.

Bucaioni, A., Di Salle, A., Iovino, L., Malavolta, I. and Pelliccione, P. (Aug. 2022). 'Reference architectures modelling and compliance checking'. In: *Journal Software and Systems Modeling (SoSyM)*. DOI: 10.1007/s10270-022-01022-z.

Cheng, B. H. C., Degueule, T., Atkinson, C., Clarke, S., Frank, U., Mosterman, P. J. and Sztipanovits, J. (2015). 'Motivating Use Cases for the Globalization of DSLs'. In: *Globalizing Domain-Specific Languages: International Dagstuhl Seminar, Dagstuhl Castle, Germany, October 5-10, 2014, Revised Papers*. Springer International Publishing, pp. 21–42. DOI: 10.1007/978-3-319-26172-0_3.

Dalibor, M., Heithoff, M., Michael, J., Netz, L., Pfeiffer, J., Rumpe, B., Varga, S. and Wortmann, A. (2022). 'Generating Customized Low-Code Development Platforms for Digital Twins'. In: *Journal of Computer Languages (COLA)* 70. DOI: 10.1016/j.cola.2022.101117.

Dalibor, M., Jansen, N., Kirchhof, J. C., Rumpe, B., Schmalzing, D. and Wortmann, A. (2019). 'Tagging Model Properties for Flexible Communication'. In: *Proc. of MODELS 2019. Workshop MDE4IoT*. Ed. by N. Ferry, A. Cicchetti, F. Ciccozzi, A. Solberg, M. Wimmer and A. Wortmann. CEUR Workshop Proceedings, pp. 39–46.

Drave, I., Kautz, O., Michael, J. and Rumpe, B. (2019). 'Semantic Evolution Analysis of Feature Models'. In: *Int. Systems and Software Product Line Conference (SPLC'19)*. Ed. by T. Berger, P. Collet, L. Duchien, T. Fogdal, P. Heymans, T. Kehrer, J. Martinez, R. Mazo, L. Montalvillo, C. Salinesi, X. Tërnava, T. Thüm and T. Ziadi. ACM, pp. 245–255.

Eom, S.-J. and Fountain, J. E. (2013). *Enhancing information services through public-private partnerships: Information technology knowledge transfer underlying structures to develop shared services in the U.S. and Korea*, pp. 15–40. DOI: 10.4018/978-1-4666-4173-0.ch002.

Fettke, P., Loos, P. and Zwicker, J. (2006). 'Business Process Reference Models: Survey and Classification'. In: *Business Process Management Workshops*. Ed. by C. J. Bussler and A. Haller. Springer Berlin Heidelberg, pp. 469–483.

Forme, F. G. L., Botta-Genoulaz, V. and Campagne, J. (2007). 'A framework to analyse collaborative performance'. In: *Comput. Ind.* 58.7, pp. 687–697. DOI: 10.1016/j.compind.2007.05.007.

Frank, U. (2002). 'Multi-perspective enterprise modeling (MEMO) conceptual framework and modeling languages'. In: *35th Annual Hawaii International Conference on System Sciences*, pp. 1258–1267. DOI: 10.1109/HICSS.2002.993989.

Frank, U. (2007). 'Evaluation of Reference Models'. In: *Reference Modeling for Business Systems Analysis*. Ed. by P. Fettke and P. Loos. IGI Global, pp. 118–140. ISBN: 9781599040547. DOI: 10.4018/978-1-59904-054-7.ch006.

Frank, U. (2010). *Outline of a method for designing domain-specific modelling languages*. ICB-Research Report 42.

Frank, U. (2014a). 'Higher Value of Research by Promoting Value for Researchers'. In: *Communications of the Association for Information Systems* 34. DOI: 10.17705/1CAIS.03443.

Frank, U. (2014b). 'Multilevel Modeling: Toward a New Paradigm of Conceptual Modeling and Information Systems Design'. In: *Business & Information Systems Engineering* 6.6, pp. 319–337. DOI: 10.1007/s12599-014-0350-4.

Frank, U. (2022). 'Multi-level modeling: cornerstones of a rationale'. In: *Software and Systems Modeling* 21.2, pp. 451–480. DOI: 10.1007/s10270-021-00955-1.

Frank, U. and Strecker, S. (2007). 'Open Reference Models - Community-driven Collaboration to Promote Development and Dissemination of Reference Models'. In: *Enterprise*

Modelling and Information Systems Architectures – International Journal of Conceptual Modeling (EMISAJ) 2.2, pp. 32–41. DOI: 10.18417/emisa.2.2.4.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional.

Greifenberg, T., Look, M., Roidl, S. and Rumpe, B. (2015). 'Engineering Tagging Languages for DSLs'. In: *Conference on Model Driven Engineering Languages and Systems (MODELS'15)*. ACM/IEEE, pp. 34–43.

Gray, J. and Rumpe, B. (2021). 'Reference models: how can we leverage them?' In: *Journal Software and Systems Modeling (SoSyM)* 20.6, pp. 1775–1776.

Herold, S., Mair, M., Rausch, A. and Schindler, I. (2013). 'Checking Conformance with Reference Architectures: A Case Study'. In: *17th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2013*. Ed. by D. Gasevic, M. Hatala, H. R. M. Nezhad and M. Reichert. IEEE Computer Society, pp. 71–80. DOI: 10.1109/EDOC.2013.17.

Hölldobler, K., Kautz, O. and Rumpe, B. (2021). *MontiCore Language Workbench and Library Handbook: Edition 2021*. Aachener Informatik-Berichte, Software Engineering, Band 48. Shaker Verlag.

Hölldobler, K. (Dec. 2018). *MontiTrans: Agile, modellgetriebene Entwicklung von und mit domänenspezifischen, kompositionalen Transformationssprachen*. Aachener Informatik-Berichte, Software Engineering, Band 36. Shaker Verlag.

Joshi, H. and Michel, H. (2008). 'Integrated Technical Reference Model and Sensor Network Architecture'. In: *Int. Conf. on Wireless Networks*. Ed. by H. R. Arabnia and V. A. Clincy. CSREA Press, pp. 570–576.

Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I. and Valduriez, P. (2006). 'ATL: a QVT-like transformation language'. In: *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, pp. 719–720.

Kautz, O. and Rumpe, B. (2018). 'On Computing Instructions to Repair Failed Model Refinements'. In: *Conference on Model Driven Engineering Languages and Systems (MODELS'18)*. ACM, pp. 289–299.

Kurtev, I. (2007). 'State of the art of QVT: A model transformation language standard'. In: *International Symposium on Applications of Graph Transformations with Industrial Relevance*. Springer, pp. 377–393.

Livi, L. and Rizzi, A. (Aug. 2012). 'The graph matching problem'. In: *Pattern Analysis and Applications* 16.3, pp. 253–283. DOI: 10.1007/s10044-012-0284-8.

Mayr, H. C. and Michael, J. (2012). 'Control pattern based analysis of HCM-L, a language for cognitive modeling'. In: *Int. Conference on Advances in ICT for Emerging Regions (ICTer2012)*. IEEE, pp. 169–175. DOI: 10.1109/ICTer.2012.6421414.

Mayr, H. C. and Thalheim, B. (2020). 'The triptych of conceptual modeling'. In: *Software and Systems Modeling*. DOI: 10.1007/s10270-020-00836-z.

Michael, J. and Mayr, H. C. (2017). 'Intuitive understanding of a modeling language'. In: *Asia-Pacific Conference on Conceptual Modeling at the Australasian Computer Science Week Multiconference (ACSW'17)*. ACM.

Michael, J., Koren, I., Dimitriadis, I., Fulterer, J., Gannouni, A., Heithoff, M., Hermann, A., Hornberg, K., Kröger, M., Sapel, P., Schäfer, N., Theissen-Lipp, J., Decker, S., Hopmann, C., Jarke, M., Rumpe, B., Schmitt, R. H. and Schuh, G. (2023). 'A Digital Shadow Reference Model for Worldwide Production Labs'. In: *Internet of Production: Fundamentals, Applications and Proceedings*. Ed. by C. Brecher, G. Schuh, W. van der Aalst, M. Jarke, F. Piller and M. Padberg. Springer, pp. 1–28. DOI: 10.1007/978-3-030-98062-7_3-2.

Michael, J. and Mayr, H. C. (2013). 'Conceptual Modeling for Ambient Assistance'. In: *Conceptual Modeling - ER 2013*. Vol. 8217. LNCS. Springer, pp. 403–413.

Michael, J. and Mayr, H. C. (2015). 'Creating a Domain Specific Modelling Method for Ambient Assistance'. In: *International Conference on Advances in ICT for Emerging Regions (ICTer2015)*. IEEE, pp. 119–124.

Mayr, H. C., Michael, J., Shekhovtsov, V. A., Ranasinghe, S. and Steinberger, C. (2018). 'A Model Centered Perspective on Software-Intensive Systems'. In: *Enterprise Modeling and Information Systems Architectures (EMISA'18)*. Ed. by M. Fellmann and K. Sandkuhl. Vol. 2097. CEUR Workshop Proceedings. CEUR-WS.org, pp. 58–64.

Maoz, S., Ringert, J. O. and Rumpe, B. (2011). 'CDDiff: Semantic Differencing for Class Diagrams'. In: *ECOOP 2011 - Object-Oriented Programming*. Ed. by M. Mezini. Springer Berlin Heidelberg, pp. 230–254.

Maoz, S., Ringert, J. O., Rumpe, B. and Wenckstern, M. v. (2016). 'Consistent Extra-Functional Properties Tagging for Component and Connector Models'. In: *WS on Model-Driven Engineering for Component-Based Software Systems (ModComp'16)*. Vol. 1723. CEUR Workshop Proceedings, pp. 19–24.

Nachmann, I., Rumpe, B., Stachon, M. and Sebastian, S. (June 2022). 'Open-World Loose Semantics of Class Diagrams as Basis for Semantic Differences'. In: *Modellierung 2022*. Gesellschaft für Informatik, pp. 111–127.

Paech, B. and Rumpe, B. (1994). 'A new Concept of Refinement used for Behaviour Modelling with Automata'. In: *Proceedings of the Industrial Benefit of Formal Methods (FME'94)*. LNCS 873. Springer, pp. 154–174.

Reinhartz-Berger, I., Soffer, P. and Sturm, A. (2010). 'Extending the Adaptability of Reference Models'. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 40.5, pp. 1045–1056. DOI: 10.1109/TSMCA.2010.2044408.

Rumpe, B. (2017). *Agile Modeling with UML: Code Generation, Testing, Refactoring*. Springer International.

Rumpe, B. (1996). *Formale Methodik des Entwurfs verteilter objektorientierter Systeme*. Dissertation, Fakultät für Informatik, Technische Universität München. München: Herbert Utz Verlag Wissenschaft.

Scheer, A.-W. (1994). *Business Process Engineering: Reference Models for Industrial Enterprises*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-79142-0.

Schürr, A. (1995). 'Specification of graph translators with triple graph grammars'. In: *Graph-Theoretic Concepts in Computer Science*. Springer Berlin Heidelberg, pp. 151–163. DOI: 10.1007/3-540-59071-4_45.

Sinz, E. J. and Schürr, A., eds. (2012). *Modellierung 2012*. Lecture Notes in Informatics P-201. Springer.

Stachowiak, H. (1973). *Allgemeine Modelltheorie*. Wien, New York: Springer Verlag.

Strüber, D., Born, K., Gill, K. D., Groner, R., Kehrer, T., Ohrndorf, M. and Tichy, M. (2017). 'Henshin: A usability-focused framework for emf model transformation development'. In: *International Conference on Graph Transformation*. Springer, pp. 196–208.

van der Aalst, W. and Kumar, A. (2001). 'A reference model for team-enabled workflow management systems'. In: *Data & Knowledge Engineering* 38.3, pp. 335–363. ISSN: 0169-023X. DOI: https://doi.org/10.1016/S0169-023X(01)00034-9.

Weinreich, R. and Buchgeher, G. (2014). 'Automatic Reference Architecture Conformance Checking for SOA-Based Software Systems'. In: *IEEE/IFIP Conference on Software Architecture, WICSA 2014*. IEEE, pp. 95–104. DOI: 10.1109/WICSA.2014.22.

Zimmermann, A., Schmidt, R., Sandkuhl, K., Wißotzki, M., Jugel, D. and Möhring, M. (2015). ‘Digital Enterprise Architecture - Transformation for the Internet of Things’. In: *19th IEEE International Enterprise Distributed Object Computing Workshop, EDOC Workshops 2015, Adelaide, Australia, September 21-25, 2015*. Ed. by J. Kolb, B. Weber, S. Hallé, W. Mayer, A. K. Ghose and G. Grossmann. IEEE Computer Society, pp. 130–138. DOI: 10.1109/EDOCW.2015.16.

Chapter 15

Enterprise Use Cases Combining Knowledge Graphs and Natural Language Processing

Phillip Schneider, Tim Schopf, Juraj Vladika and Florian Matthes

Knowledge management is a critical challenge for enterprises in today's digital world, as the volume and complexity of data being generated and collected continue to grow incessantly. Knowledge graphs (KG) emerged as a promising solution to this problem by providing a flexible, scalable, and semantically rich way to organize and make sense of data. This paper builds upon a recent survey of the research literature on combining KGs and Natural Language Processing (NLP). Based on selected application scenarios from enterprise context, we discuss synergies that result from such a combination. We cover various approaches from the three core areas of KG construction, reasoning as well as KG-based NLP tasks. In addition to explaining innovative enterprise use cases, we assess their maturity in terms of practical applicability and conclude with an outlook on emergent application areas for the future.

15.1 Introduction

As modern organizations continuously adapt to the evolving requirements of the digital age, the importance of successfully managing enterprise data has never been greater. In this article, we define the term 'enterprise' as a large-scale business, which operates on a national or international level and typically involves significant risks and resources. The competitive advantage of data-driven decisions affects all industry sectors and nearly every part of the value chain (Schopf et al. 2022b). For example, market trends are analyzed for business development, production is optimized through process metrics, and customer reviews are monitored for predictive maintenance. However, raw data alone is insufficient for decision-making. In order to become actionable information, data has to be endowed with meaning and purpose (Rowley 2007). This can be achieved by data enrichment through a relevant context. A compelling approach to achieve this is by modeling knowledge in the form of graph connections between data items (Martin et al. 2021).

In view of the above, knowledge graphs (KGs) have emerged as a powerful representation for integrating knowledge from multiple information sources. They model semantic relationships among key entities of interest, such as customers, markets, or services. In this sense, a KG serves as an abstraction layer to combine both business data and explicit business knowledge. Even though Google coined the term knowledge graph back in 2012 when they announced their new web search strategy (Singhal 2012), it is not an entirely novel concept. The notion of knowledge graphs stems from scientific advancements in areas like semantic web, database systems, and machine learning. Fueled by the demand for intelligent data management and strong enterprise use cases, the integration of ideas from the aforementioned fields has attracted a surge of interest from both academia and industry in the last years (Hitzler 2021).

In particular, the adoption of KGs has been noticeable in the field of natural language processing (NLP). The underlying paradigm is that graphs can serve as an intermediate formalism to bridge structured and unstructured, textual knowledge. This entails potential benefits for all kinds of NLP tasks. Because a major part of business knowledge resides in unstructured text sources, leveraging KGs and NLP for data management appears to be an auspicious endeavor. In our comprehensive survey on the research landscape of KGs in NLP, we provide an overview of the rapidly expanding body of literature (Schneider et al. 2022a). Parallel to this trend, the variety of explored use cases and application domains grew as well. We identified 20 individual domains, many of which belong to enterprise-related subdomains. Yet, despite numerous scientific contributions, there are very few publications discussing the promised usefulness in practice. Therefore, the following research question arises:

What is the added value of combining KGs and NLP for enterprise use cases? Taking a step towards answering this research question, we build upon our survey results and investigate the synergies between KGs and NLP with selected enterprise use cases, ranging from knowledge acquisition to knowledge application. Concrete examples are discussed that showcase the added value that this form of graph-based knowledge processing brings. Besides explaining innovative methods for the three core areas of KG construction, KG reasoning, and KG-based NLP applications, we also discuss their maturity in terms of practical applicability. We conclude with an outlook on particularly promising KG and NLP application scenarios for future research.

15.2 Background and Terminology

15.2.1 Knowledge Graph Concept

In recent years, KGs have emerged as an approach for semantically representing knowledge about real-world entities in a machine-readable format. In contrast to semantic text representations, which may be used primarily for similarity comparisons in a variety of different NLP downstream tasks (Braun et al. 2021; Schopf et al. 2021a; Schopf et al. 2021b; Schopf et al. 2022d; Schopf et al. 2022c; Schopf et al. 2022a; Schneider et al. 2022b), KGs can additionally capture all kinds of semantic relationships between different entities. Despite the rising popularity of KGs, there is still no common understanding of what exactly a KG is. Although prior work has already attempted to define KGs (Pujara et al. 2013; Ehrlinger and Wöß 2016; Paulheim 2017; Färber et al. 2018), the term is not yet used uniformly by researchers. Most works implicitly adopt a broad definition of KGs, where they are understood as *'a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities'* (Hogan et al. 2022). KGs originated from research on semantic networks, domain-specific ontologies, as well as linked data, and are thus not an entirely new concept (Hitzler 2021). Since the concepts of ontologies, semantic networks, and KGs are strongly related, these terms are often mistakenly used as synonyms, leading to additional confusion (Ehrlinger and Wöß

2016). Although an ontology can also hold instances (i. e., the population of the ontology), an ontology mainly consists of classes and properties as a formal, explicit specification of a shared conceptualization that is characterized by high semantic expressiveness required for increased complexity (Feilmayr and Wöß 2016). Semantic networks can be seen as predecessors of KGs (Hogan et al. 2022) and represent semantic relations between conceptual units as a net-like graph ('Semantic networks' 1992). One of the most well-known semantic networks in NLP is WordNet, which consists of English nouns, verbs, adjectives, and adverbs that are organized into sets of synonyms, each representing a lexicalized concept (Miller 1995).

Relational Database Management Systems (RDBMS) are collections of data that are stored in manually modeled tables that are connected to each other (Medhi and Baruah 2017). In contrast to KGs, RDBMS do not store relationships between individual instances (nodes) but use predefined tabular schemas. Therefore, RDBMS are more powerful when processing large amounts of data that do not change in structure. KGs, on the other hand, are more powerful when data is highly connected, the data model changes frequently, or data retrieval and analysis are more important than simply storing the data. Since companies often have multiple databases that are semantically related and also require analysis or further processing of the data for downstream NLP tasks, the current tendency is to create enterprise knowledge graphs.

Enterprise knowledge graphs consist of connected concepts, properties, instances and relationships representing and referencing foundational and domain knowledge within or across different enterprises (Fraunhofer IAIS 2021). They are realizations of linked enterprise data, creating valuable business insights by bringing together semantic technologies and enterprise data infrastructures (Fraunhofer IAIS 2021). Enterprise knowledge graphs are typically internal to a company and applied for commercial use-cases (Noy et al. 2019). Prominent industries using enterprise knowledge graphs include web search, e-commerce, social networks, and finance, among others, where applications include search, recommendations, information extraction, personal agents, advertising, business analytics, risk assessment, automation, and more (Hogan et al. 2022).

15.2.2 Knowledge Graphs in Natural Language Processing

NLP is a field at the intersection of linguistics, computer science, and artificial intelligence. Its goal is to build computer systems that can understand and interact with human language and solve various tasks involving language data. The way humans communicate and exchange knowledge is often in an unstructured format – large corpora of written text documents, presentation slides, e-mails, or chat messages. There is a trend of trying to utilize the structured representation of knowledge found in KGs in combination with the knowledge in unstructured text. KGs are a useful tool for NLP because they provide a structured, machine-readable representation of real-world entities and the relationships between them. This allows NLP systems to understand the meaning of words and phrases in context, and to use this understanding to answer questions, generate text summaries, and perform other knowledge-intensive tasks. Additionally, KGs can be used to improve the accuracy and performance of NLP systems by providing them with a wealth of background information about the world that they can use to disambiguate words and phrases and make more informed decisions.

15.3 Analysis of Enterprise Use Cases

Based on the tasks identified in the literature on KGs in NLP, we developed the empirical taxonomy shown in Figure 15.1. If one or multiple of these tasks are applied to a specific domain with the goal to create business value, we consider it as an enterprise use case. In this regard, one can distinguish between two top-level categories, namely knowledge acquisition and knowledge application. Knowledge acquisition contains NLP tasks to construct KGs from unstructured text (knowledge graph construction) or to conduct reasoning over already constructed KGs (knowledge graph reasoning). KG construction tasks are further split into the subcategories of knowledge extraction and knowledge integration. KG reasoning is used to maintain and update facts inside KGs. Lastly, knowledge application, being the second top-level category, encompasses common NLP tasks, which can be enhanced through structured knowledge from KGs.

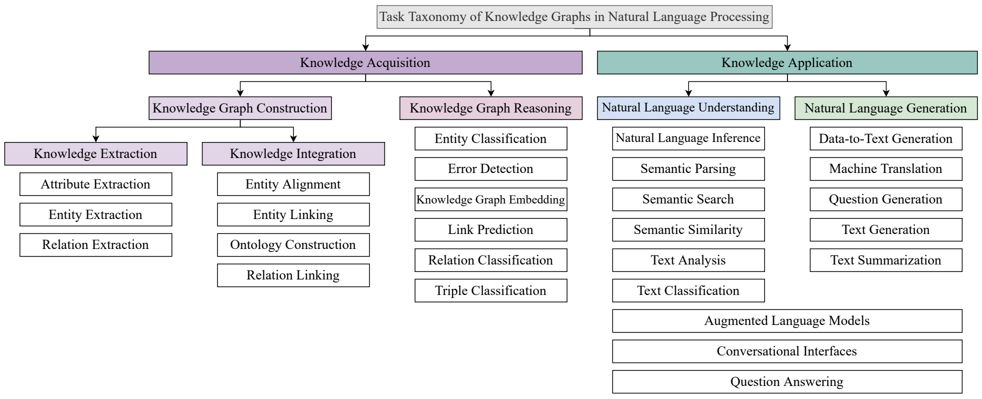


Figure 15.1: Taxonomy of tasks in the literature on KGs in NLP (Schneider et al. 2022a).

In the following subchapters, the aforementioned tasks combining KGs and NLP are explained in more detail. To demonstrate the potential value of such a combination, we introduce selected studies that exemplify enterprise use cases across multiple industry sectors. Table 15.1 provides an overview of additional studies for the described use cases.

15.3.1 Knowledge Graph Construction

Knowledge graph construction in NLP considers tasks which involve the explicit modeling of knowledge in KGs from unstructured text. As shown in Figure 15.1, KG construction can be further divided into the two subcategories of knowledge extraction and knowledge integration. Knowledge extraction tasks are used to populate KGs with entities, relations, or attributes, whereas knowledge integration tasks are used to update KGs (Schneider et al. 2022a).

Usually, *entity extraction* is the first step in the KGs construction process and is used to derive real-world entities from unstructured text (Al-Moslmi et al. 2020). After the entities are extracted, their relationships can be extracted with the task of *relation extraction* (Zhang et al. 2019). Entity extraction in combination with relation extraction is often used to construct new KGs, e. g., for news events (Rospocher et al. 2016), scholarly research (Luan

Enterprise Use Cases	Identified Domains	Selected Studies
Analysis & monitoring	Operations, tech. scouting, reporting	Gua and Liu 2019; Qin and Chow 2019; Braaksma et al. 2021; Hu et al. 2021
Digital assistant	Finance, human resources	Singh et al. 2018; Avila et al. 2020; McCrae et al. 2021
Document generation	Scholarly publishing	Fan et al. 2019; Koncel-Kedziorski et al. 2019; Wang et al. 2021
Knowledge repository	Engineering, scholarly publishing	Li et al. 2020; Sarica et al. 2020; Wang et al. 2020
Search engine	Energy, engineering, law, software	Birnie et al. 2019; Sarica et al. 2019; Zhou et al. 2020; Gao et al. 2021
Trade recommendation	Finance, law	Ding et al. 2016; Du et al. 2022

Table 15.1: Selected studies of enterprise use cases in the literature on KGs in NLP.

et al. 2018; Shen et al. 2018), patent mining (Wu et al. 2021), engineering technologies (Sarica et al. 2020), or the petroleum industry (Zhou et al. 2020). *Entity linking* tasks link entities recognized in a text to already existing entities in KGs (Moon et al. 2018; Wu et al. 2020). Oftentimes, synonymous or similar entities exist in different KGs. To reduce redundancy and repetition in future tasks, *entity alignment* can be applied to align these entities (Gangemi et al. 2016; Chen et al. 2018). Coming up with the rules and schemes of KGs, i. e., their structure and format of knowledge presented in it, is done with the task of *ontology construction* (Haussmann et al. 2019; Schneider et al. 2022a). In the following, we present two selected KGs constructed from texts which support knowledge-intensive processes in enterprises of different industries.

Microsoft Academic Graph. The average growth rate ranged from 5 to 6.5 per cent for scientific articles and from 2 to 3 per cent for journals between 2015 and 2020, causing the number of publications to grow exponentially over the past centuries (International Association of Scientific, Technical and Medical Publishers 2021). Due to this unprecedented growth of scientific knowledge, efficiently identifying relevant research has become an ever-increasing challenge (Shen et al. 2018). Enterprises often seek to increase their capabilities and innovation by incorporating new scientific discoveries into their business processes. In order to assist researchers and businesses to navigate the entirety of scientific information, the Microsoft Academic Graph (MAG) organizes scientific knowledge in a hierarchical manner (Shen et al. 2018). MAG is a KG in which the nodes represent scientific entities and the edges represent the relationships between them (Wang et al. 2020). In the MAG, publications are the main entity type to which all other entities (e. g., authors, venues, field of study concepts, etc.) are connected to in one way or another (Wang et al. 2020).

TechNet. Various NLP techniques were used to extract terms and their relationships from the complete 1976–2017 U.S. patent database and to create TechNet (Sarica et al. 2020). It is a semantic network that covers elemental concepts in all domains of technology and their semantic associations. TechNet may serve as an infrastructure to support a wide range of applications, e. g., technical text summaries, search query predictions, relational knowledge discovery, and design ideation support, in the context of engineering and technology, and complement or enrich existing semantic databases. Therefore, TechNet can support corporations with an up-to-date resource for technical knowledge and help them stay competitive in a rapidly changing technological landscape.

15.3.2 Knowledge Graph Reasoning

Subsequent to the construction of a KG, it contains world knowledge in a structured format. This knowledge can be general encyclopedic knowledge or domain-specific knowledge that is useful for a certain target group. The tasks in the group of KG reasoning aim to infer new knowledge and conclusions by using the already present knowledge in the KG.

Written text often contains named entities and concepts that have to be categorized in order for NLP techniques to better understand them. The task of classifying entities found in the text based on the knowledge from a KG is called *entity classification*, while the task of retrieving a relation from a KG between two entities found in text is called *relation classification.* If two entities and their relation is considered jointly in a triple format, this task is called *triple classification.* The task of *link prediction* tries to infer missing links between entities in existing KGs and is often performed via ranking entities as possible answers to queries. It is also possible that there are erroneous entities or relations in a KG – the task of *error detection* addresses this problem and aims to refine the knowledge of a KG. *Knowledge graph embedding* techniques are used to create dense vector representations of knowledge from the graph.

Misinformation detection is a NLP task where KG reasoning and external knowledge are important for enterprises. This task has gained wide attention because the spread of fake news and biased news can have significant socio-economic consequences, increase mistrust within society, and lead to stock market perturbations or other economic disruptions (Scheufele and Krause 2019; Palić et al. 2019; Clarke et al. 2021). Some approaches for fake news detection rely solely on linguistic cues such as how sensationalist or persuasive the text of a news article is, or on social cues like where the news first originated and how it was propagated. Nevertheless, for optimal performance, NLP models for this task need to incorporate background knowledge for a deeper understanding of what is being described in the text. For example, Hu et al. (2021) match entities such as public persons, places, or events mentioned in news articles to their respective entities in a knowledge graph and then train graph neural networks (GNNs) to learn about the context of each mentioned entity. This new knowledge is then used to enhance the performance of fake news detection.

Most KG reasoning applications rely on representing knowledge graph concepts as embeddings which are then in a suitable format to be used as input to machine learning models like deep neural networks. This concept is also used in NLP to represent words as vector embeddings. Sometimes text and graph embeddings are learned jointly, as to make their representation closer in the embedding space. In Ding et al. (2016), the authors propose a model that jointly learns event embeddings by using both the textual and world knowledge from a KG. The model utilizes world knowledge from the YAGO knowledge graph, which was constructed using knowledge from Wikipedia (Suchanek et al. 2007). Examples of triples in that KG include *(Bill Gates, created, Microsoft)* and *(Apple, isA, electronics company).* Afterward, the authors use these event embeddings for the task of stock market prediction and show that such joint embeddings achieved better performance on the task of stock market volatility prediction.

15.3.3 Knowledge-Based Natural Language Understanding and Generation

After the construction and reasoning-enabled refinement of a KG, it can be used as an information source for enhancing tasks in the areas of natural language understanding and natural language generation. While the former is concerned with parsing syntax, semantics, or pragmatics and thus understanding the meaning of natural language, the latter focuses on producing human-like text based on linguistic or non-linguistic input.

Automated approaches for extracting the content and meaning of textual data are of great importance for enterprises. Having pertinent information at hand is the foundation of data-driven decision-making. In this regard, the main challenge of managing knowledge lies not in storing data in digital repositories, but in retrieving the relevant information pieces amongst an overwhelming mass of other data items. One of the key tasks of using KGs in natural language understanding is *semantic search*, which refers to 'search with meaning'. Semantic search goes beyond literal keyword matches and aims at understanding the search intent and context (Bast et al. 2016). This increases the information retrieval performance despite ambiguous or poorly phrased queries. Semantic search is often a part of business information systems like search engines, recommender systems, or analytical tools. Aside from understanding language, KGs can be used for the task of *text generation*. This concerns automatically generating business reports or other textual outputs that provide a concise representation of the enterprise data contained in a KG.

Bringing together language understanding and generation capabilities, *question answering (QA)* is another common NLP task. Its goal is to find an answer given a user query formulated in natural language. QA systems can be distinguished into three categories: (1) textual QA for extracting answers from a document corpus, (2) question answering over knowledge bases (KBQA) for getting answers from structured knowledge bases such as KGs (Fu et al. 2020), and (3) hybrid approaches that use a combination of text corpora as well as knowledge bases (Yu et al. 2022). *Conversational interfaces* surpass the restricted question-answer scheme and emulate a realistic dialogue in human language. Users can interact with conversational interfaces through multiple conversation turns, wherefore they are especially suitable to handle more complex information-seeking scenarios (McTear et al. 2016). To name just a few examples, conversational interfaces are commonly used to inform employees about company policies, provide data-based insights for business operations, or improve customer service by quickly resolving issues.

As a semantic search use case related to the previously introduced *TechNet* graph, Sarica et al. (2019) propose a patent retrieval system (*SerKeys*) based on keywords from a constructed engineering knowledge graph. Engineers often search for patents regarding a design topic to learn about the prerequisites of existing technologies. However, discovering patents relevant to a specific design interest, such as additive manufacturing or autonomous logistic robots, is a non-trivial task because keywords are often intuitively chosen and limited by the vocabulary of the searcher. This leads to overlooked keywords, suboptimal queries, and poor search results. To address this concern, the KG-based patent retrieval system lets users select recommended keywords to expand and refine their queries. In a small case study, the authors show that keyword discovery improves the completeness of queries and reduces the search effort of engineers. An additional example of semantic search is *PetroKG*, a KG that integrates heterogeneous data distributed in various sources in the upstream sector of the enterprise PetroChina (Zhou et al. 2020). To search for fossil fuel deposits and extract these resources from the earth, PetroChina has to accumulate a huge amount of production data, online encyclopedia data, and unstructured documents, which are all integrated and semantically interlinked in the PetroKG. The latter is used for a KG-powered semantic search system that detects entities from user input, executes graph queries, and returns concise answers. This knowledge retrieval service supports researchers and engineers employed at PetroChina who previously had to search through long lists of documents and industrial data logs in their daily work.

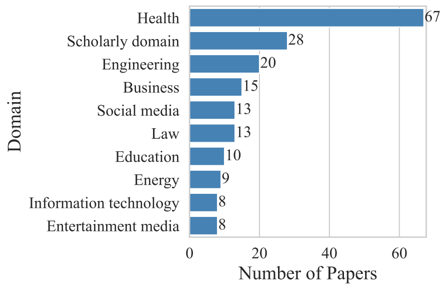
One example of conversational interfaces is *KNADIA*, an intra-enterprise knowledge-assisted dialogue system that has been deployed in production in Tata Consultancy Services (Singh et al. 2018). Its system architecture supports two use cases: virtual assistance and knowledge synthesis. While the former is about handling questions from employees about human resource policies, the latter aims at storing information about reusable assets, completed projects, or research documents in a KG and making it accessible through a conversational agent. This agent is also able to approach employees with questions to autonomously update the KG. The KNADIA architecture consists of a high-level intent identification layer and a bot manager which matches all user requests to a number of task-specific conversational bots. After its deployment, the creators of KNADIA observed a significant growth of user queries per day, especially subsequent to opening an auto-suggest feature, where the system proactively showed possible questions users could ask based on recent conversations.

Moreover, there are many potential enterprise use cases for generating texts from KGs. Although prior work has demonstrated that language models can produce short texts using structured information from a KG, the existing models are still prone to missing important relations or hallucinating wrong facts (Koncel-Kedziorski et al. 2019; Wang et al. 2021). Strategies for mitigating these problems are a matter of ongoing NLP research. For these reasons, the proposed solutions are almost never deployed and tested in practice.

15.4 Discussion

The described studies reveal how enterprises can use KGs as a means of of structuring information and modeling complex relationships. If there is no existing KG to build upon, an enterprise has to go through the mentioned three phases, whereas each phase heavily depends on NLP techniques. The first phase is KG construction, which involves collecting and organizing data from heterogeneous sources such as databases, documents, or external data providers. Once the KG is constructed, it can be refined through machine learning and manual curation. This process relies on reasoning algorithms that can not only identify and correct inconsistencies in the data, but also add new information to the graph. Finally, the KG can be used for various NLP tasks to improve the efficiency and effectiveness of an enterprise's data management and decision-making processes.

Another observation from the presented use cases is the capability of KGs to capture a great variety of domain knowledge in an expressive and flexible fashion. The graph representation of nodes and edges is especially useful for areas with highly interconnected data items, which is more difficult to manage in traditional, relational databases. In our survey, we identified 20 different application domains. A subset of the most prominent ones can be seen in the left bar chart in Figure 15.2. Apart from the dominating health domain, there are many enterprise use cases emerging from other domains such as business, engineering, law, or energy. In particular, our investigation of use cases included topics



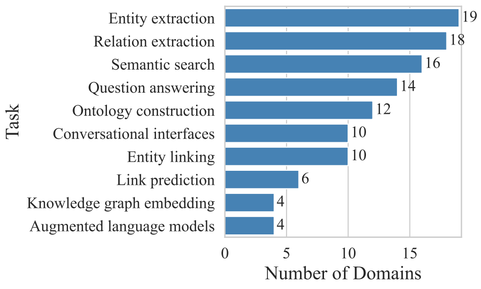


Figure 15.2: Number of papers by most popular application domains (left) and overview of most popular tasks by number of application domains (right) (Schneider et al. 2022a).

about patents, fossil fuel exploration and production, finance, software engineering, human resources, product design, and many more. This diversity shows that KGs and NLP can be valuable for a very wide range of domains and applications.

In view of the large number of tasks studied in the literature (see Figure 15.1), it is important to note that some are more mature in terms of practical applicability than others. According to our definition, maturity reflects the degree to which a given task has been examined with different research methods, resulting in a diverse set of available research contributions. Further, a given task can be considered more mature if it has been studied across a larger number of application domains, as depicted in the right bar chart in Figure 15.2. More mature tasks have a higher reliability of creating benefits in the discussed enterprise use cases. In line with the findings of our survey, tasks focused on KG construction as well as KG-based search systems are already mature (Schneider et al. 2022a). Accordingly, most papers from our analysis represent use cases for building up KGs from unstructured text or retrieving information from KGs, often in form of semantic search engines. Hence, the technology readiness of NLP-supported storage and access of data in KGs is reasonably high. While QA is an established task, there is a significant increase in research interest in the more advanced conversational interfaces. They can leverage the contextualized data in graphs to handle complicated information-seeking dialogues. However, more research work on deploying conversational agents and evaluating their usefulness in industrial settings is required. Other emerging research streams around KG embedding or KG-augmented language models have a high potential for enterprise use cases, too, although there are extremely few studies that apply these tasks in practical scenarios. Consequently, use cases that rely on graph embeddings or augmented language models are still actively researched and thus less mature. This concerns, for example, automatic document generation, embedding-based trading recommendation, or systems that use reasoning to infer new knowledge.

15.5 Conclusion

We analyzed enterprise use cases regarding the combination of KGs and NLP. By modeling semantic relationships among key entities and leveraging unstructured text sources, KG-based NLP can provide valuable insights and support decision-making in various application scenarios. Building upon the findings of our previous survey, we introduced the three core areas of KG construction, reasoning, and KG-based NLP tasks. For each corea area, we presented selected studies that apply these tasks in an enterprise context, including domains such as engineering, law, publishing, or finance. In addition, we assessed the maturity of use cases in terms of practical applicability.

Overall, we expect that the rapidly growing body of literature on combining KG and NLP will continue to bring forth innovative technologies and scholars will continue to explore even more application domains. Other enterprises will start to adopt KGs as a strategic tool to drive digital transformation, following the example of big technology companies like Google, which have harnessed the advantages of KGs for their internal processes and products for almost a decade now. Nonetheless, despite the strong business use cases, we observed a lack of applied research in practice. Compared to the whole body of literature about KGs in NLP, papers that focus on concrete enterprise use cases are still scarce. This might seem counterintuitive because the research interest in academia about KGs has been driven by the industry to a large extent. Hence, we call for more evaluation research in real-world scenarios that shed light on the added value of implemented solutions. This is crucial to make significant strides in advancing the field and transferring findings from academia to practical applications.

References

Al-Moslmi, T., Ocaña, M. G., Opdahl, A. L. and Veres, C. (2020). 'Named Entity Extraction for Knowledge Graphs: A Literature Overview'. In: *IEEE Access* 8, pp. 32862–32881. DOI: 10.1109/ACCESS.2020.2973928.

Avila, C. V. S., Franco, W., Maia, J. G. R. and Vidal, V. M. P. (2020). 'CONQUEST: A Framework for Building Template-Based IQA Chatbots for Enterprise Knowledge Graphs'. In: *Natural Language Processing and Information Systems - 25th International Conference on Applications of Natural Language to Information Systems, NLDB 2020, Saarbrücken, Germany, June 24–26, 2020, Proceedings*. Ed. by E. Métais, F. Meziane, H. Horacek and P. Cimiano. Vol. 12089. Lecture Notes in Computer Science. Springer, pp. 60–72. DOI: 10.1007/978-3-030-51310-8_6.

Bast, H., Buchhold, B., Haussmann, E. et al. (2016). 'Semantic search on text and knowledge bases'. In: *Foundations and Trends in Information Retrieval* 10.2–3, pp. 119–271. DOI: 10.1561/1500000032.

Birnie, C. E., Sampson, J., Sjaastad, E., Johansen, B., Obrestad, L. E., Larsen, R. and Khamassi, A. (2019). 'Improving the Quality and Efficiency of Operational Planning and Risk Management with ML and NLP'. In: SPE Offshore Europe Conference and Exhibition. DOI: 10.2118/195750-MS.

Braaksma, B., Daas, P., Raaijmakers, S., Geurts, A. and Meyer-Vitali, A. (2021). 'AI-Supported Innovation Monitoring'. In: *Trustworthy AI - Integrating Learning, Optimization and Reasoning*. Ed. by F. Heintz, M. Milano and B. O'Sullivan. Springer, pp. 220–226. DOI: 10.1007/978-3-030-73959-1_20.

Braun, D., Klymenko, O., Schopf, T., Kaan Akan, Y. and Matthes, F. (2021). 'The Language of Engineering: Training a Domain-Specific Word Embedding Model for Engineering'. In: *2021 3rd International Conference on Management Science and Industrial Engineering, MSIE 2021, Osaka, Japan*, pp. 8–12. DOI: 10.1145/3460824.3460826.

Chen, M., Tian, Y., Chang, K., Skiena, S. and Zaniolo, C. (2018). 'Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment'. In:

Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden. Ed. by J. Lang. ijcai.org, pp. 3998–4004. DOI: 10.24963/ijcai.2018/556.

Clarke, J., Chen, H., Du, D. and Hu, Y. ((2021). 'Fake News, Investor Attention, and Market Reaction'. In: *Information Systems Research* 32.1, pp. 35–52. DOI: 10.1287/isre.2019.0910.

Ding, X., Zhang, Y., Liu, T. and Duan, J. (Dec. 2016). 'Knowledge-Driven Event Embedding for Stock Prediction'. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers.* Osaka, Japan: The COLING 2016 Organizing Committee, pp. 2133–2142. URL: https://aclanthology.org/C16-1201.

Du, W., Yan, Q., Zhang, W. and Ma, J. (2022). 'Leveraging online behaviors for interpretable knowledge-aware patent recommendation'. In: *Internet Research* 32.2, pp. 568–587. DOI: 10.1108/INTR-08-2020-0473.

Ehrlinger, L. and Wöß, W. (2016). 'Towards a Definition of Knowledge Graphs'. In: *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016.* Ed. by M. Martin, M. Cuquet and E. Folmer. Vol. 1695. CEUR Workshop Proceedings. CEUR-WS.org. URL: https://ceur-ws.org/Vol-1695/paper4.pdf.

Fan, A., Gardent, C., Braud, C. and Bordes, A. (2019). 'Using Local Knowledge Graph Construction to Scale Seq2Seq Models to Multi-Document Inputs'. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* Association for Computational Linguistics, pp. 4186–4196. DOI: 10.18653/v1/D19-1428.

Färber, M., Bartscherer, F., Menne, C. and Rettinger, A. (2018). 'Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO'. In: *Semantic Web* 9.1, pp. 77–129. DOI: 10.3233/SW-170275.

Feilmayr, C. and Wöß, W. (2016). 'An analysis of ontologies and their success factors for application to business'. In: *Data Knowl. Eng.* 101, pp. 1–23. DOI: 10.1016/j.datak.2015.11.003. URL: https://doi.org/10.1016/j.datak.2015.11.003.

Fraunhofer IAIS (2021). *Enterprise knowledge graphs - Fraunhofer IAIS.* Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS). URL: https://www.iais.fraunhofer.de/en/business-areas/enterprise-information-integration/enterprise-knowledge-graphs.html.

Fu, B., Qiu, Y., Tang, C., Li, Y., Yu, H. and Sun, J. (2020). 'A Survey on Complex Question Answering over Knowledge Base: Recent Advances and Challenges'. In: *CoRR* abs/2007.13069. URL: https://arxiv.org/abs/2007.13069.

Gangemi, A., Alam, M., Asprino, L., Presutti, V. and Recupero, D. R. (2016). 'Framester: A Wide Coverage Linguistic Linked Data Hub'. In: *Knowledge Engineering and Knowledge Management - 20th International Conference, EKAW 2016, Bologna, Italy, November 19–23, 2016, Proceedings.* Ed. by E. Blomqvist, P. Ciancarini, F. Poggi and F. Vitali. Vol. 10024. Lecture Notes in Computer Science, pp. 239–254. DOI: 10.1007/978-3-319-49004-5_16.

Gao, P., Liu, X., Choi, E., Soman, B., Mishra, C., Farris, K. and Song, D. (2021). 'A System for Automated Open-Source Threat Intelligence Gathering and Management'. In: *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20–25,* *2021*. Ed. by G. Li, Z. Li, S. Idreos and D. Srivastava. ACM, pp. 2716–2720. DOI: `10.1145/3448016.3452745`.

Gua, H. and Liu, K. (2019). 'Hierarchical ontology graph for solving semantic issues in decision support systems'. In: *Proceedings of the International Conference on Enterprise Information Systems, 3–5 May 2019, Crete, Greece*, pp. 483–487. URL: `https://centaur.reading.ac.uk/85004/1/ICEIS_2019_186.pdf`.

Haussmann, S., Seneviratne, O., Chen, Y., Ne'eman, Y., Codella, J., Chen, C.-H., McGuinness, D. L. and Zaki, M. J. (2019). 'FoodKG: A Semantics-Driven Knowledge Graph for Food Recommendation'. In: *The Semantic Web – ISWC 2019*. Ed. by C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. Cruz, A. Hogan, J. Song, M. Lefrançois and F. Gandon. Springer, pp. 146–162.

Hitzler, P. (2021). 'A Review of the Semantic Web Field'. In: *Communication of the ACM* 64.2, pp. 76–83. DOI: `10.1145/3397512`.

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A.-C. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S. and Zimmermann, A. (2022). 'Knowledge Graphs'. In: *ACM Computing Surveys* 54.4, pp. 1–37. DOI: `10.1145/3447772`.

Hu, L., Yang, T., Zhang, L., Zhong, W., Tang, D., Shi, C., Duan, N. and Zhou, M. (2021). 'Compare to The Knowledge: Graph Neural Fake News Detection with External Knowledge'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, pp. 754–763. DOI: `10.18653/v1/2021.acl-long.62`.

International Association of Scientific, Technical and Medical Publishers, ed. (2021). *STM Global Brief 2021 – Economics & Market Size: An STM Report Supplement*. URL: `https://www.stm-assoc.org/2022_08_24_STM_White_Report_a4_v15.pdf`.

Koncel-Kedziorski, R., Bekal, D., Luan, Y., Lapata, M. and Hajishirzi, H. (2019). 'Text Generation from Knowledge Graphs with Graph Transformers'. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pp. 2284–2293. DOI: `10.18653/v1/N19-1238`.

Li, X., Chen, C.-H., Zheng, P., Wang, Z., Jiang, Z. and Jiang, Z. (2020). 'A Knowledge Graph-Aided Concept–Knowledge Approach for Evolutionary Smart Product–Service System Development'. In: *Journal of Mechanical Design* 142.10, p. 101403. DOI: `10.1115/1.4046807`.

Luan, Y., He, L., Ostendorf, M. and Hajishirzi, H. (2018). 'Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction'. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 3219–3232. DOI: `10.18653/v1/D18-1360`.

Martin, S., Szekely, B. and Allemang, D. (2021). *The Rise of the Knowledge Graph*. O'Reilly.

McCrae, J. P., Mohanty, P., Narayanan, S., Pereira, B., Buitelaar, P., Karmakar, S. and Sarkar, R. (2021). 'Conversation Concepts: Understanding Topics and Building Taxonomies for Financial Services'. In: *Inf.* 12.4, p. 160. DOI: `10.3390/info12040160`. URL: `https://doi.org/10.3390/info12040160`.

McTear, M. F., Callejas, Z. and Griol, D. (2016). *The Conversational Interface: Talking to Smart Devices*. Springer. URL: `https://doi.org/10.1007/978-3-319-32967-3`.

Medhi, S. and Baruah, H. K. (2017). ‘Relational database and graph database: A comparative analysis’. In: *Journal of Process Management and New Technologies* 5.2, pp. 1–9. DOI: 10.5937/jouproman5-13553.

Miller, G. A. (1995). ‘WordNet: A Lexical Database for English’. In: *Communications of the ACM* 38.11, pp. 39–41. DOI: 10.1145/219717.219748.

Moon, S., Neves, L. and Carvalho, V. (2018). ‘Multimodal Named Entity Disambiguation for Noisy Social Media Posts’. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Association for Computational Linguistics, pp. 2000–2008. DOI: 10.18653/v1/P18-1186.

Noy, N., Gao, Y., Jain, A., Narayanan, A., Patterson, A. and Taylor, J. (Apr. 2019). ‘Industry-Scale Knowledge Graphs: Lessons and Challenges: Five Diverse Technology Companies Show How It’s Done’. In: *Queue* 17.2, pp. 48–75. DOI: 10.1145/3329781.3332266.

Palić, N., Vladika, J., Čubelić, D., Lovrenčić, I., Buljan, M. and Šnajder, J. (2019). ‘TakeLab at SemEval-2019 Task 4: Hyperpartisan News Detection’. In: *Proceedings of the 13th International Workshop on Semantic Evaluation.* Association for Computational Linguistics, pp. 995–998. DOI: 10.18653/v1/S19-2172.

Paulheim, H. (2017). ‘Knowledge graph refinement: A survey of approaches and evaluation methods’. In: *Semantic Web* 8.3, pp. 489–508. DOI: 10.3233/SW-160218.

Pujara, J., Miao, H., Getoor, L. and Cohen, W. W. (2013). ‘Knowledge Graph Identification’. In: *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21–25, 2013, Proceedings, Part I.* Ed. by H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. F. Noy, C. Welty and K. Janowicz. Vol. 8218. Lecture Notes in Computer Science. Springer, pp. 542–557. DOI: 10.1007/978-3-642-41335-3_34.

Qin, S. and Chow, K. P. (2019). ‘Automatic Analysis and Reasoning Based on Vulnerability Knowledge Graph’. In: *Cyberspace Data and Intelligence, and Cyber-Living, Syndrome, and Health.* Ed. by H. Ning. Springer, pp. 3–19.

Rospocher, M., van Erp, M., Vossen, P., Fokkens, A., Aldabe, I., Rigau, G., Soroa, A., Ploeger, T. and Bogaard, T. (2016). ‘Building Event-Centric Knowledge Graphs from News’. In: *Journal of Web Semantics* 37.C, pp. 132–151. DOI: 10.1016/j.websem.2015.12.004.

Rowley, J. (2007). ‘The wisdom hierarchy: representations of the DIKW hierarchy’. In: *Journal of Information Science* 33.2, pp. 163–180. DOI: 10.1177/0165551506070706.

Sarica, S., Luo, J. and Wood, K. L. (2020). ‘TechNet: Technology semantic network based on patent data’. In: *Expert Systems with Applications* 142, p. 112995. DOI: 10.1016/j.eswa.2019.112995.

Sarica, S., Song, B., Low, E. and Luo, J. (2019). ‘Engineering Knowledge Graph for Keyword Discovery in Patent Search’. In: *Proceedings of the Design Society: International Conference on Engineering Design.* Vol. 1. 1. Cambridge University Press, pp. 2249–2258. DOI: 10.1017/dsi.2019.231.

Scheufele, D. A. and Krause, N. M. (2019). ‘Science audiences, misinformation, and fake news’. In: *Proceedings of the National Academy of Sciences* 116.16, pp. 7662–7669. DOI: 10.1073/pnas.1805871115.

Schneider, P., Schopf, T., Vladika, J., Galkin, M., Simperl, E. and Matthes, F. (2022a). ‘A Decade of Knowledge Graphs in Natural Language Processing: A Survey’. In: *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing.* Association for Computational Linguistics, pp. 601–614. URL: https://aclanthology.org/2022.aacl-main.46.

Schneider, P., Voggenreiter, M., Gulraiz, A. and Matthes, F. (2022b). 'Semantic Similarity-Based Clustering of Findings From Security Testing Tools'. In: *5th International Conference on Natural Language and Speech Processing, ICNLSP 2022, Trento, Italy, December 16–17, 2022.* Ed. by M. Abbas and A. A. Freihat. Association for Computational Linguistics, pp. 134–143. URL: https://aclanthology.org/2022.icnlsp-1.16.

Schopf, T., Braun, D. and Matthes, F. (2021a). 'Lbl2Vec: An Embedding-based Approach for Unsupervised Document Retrieval on Predefined Topics'. In: *Proceedings of the 17th International Conference on Web Information Systems and Technologies, WEBIST 2021, October 26–28, 2021.* Ed. by F. J. D. Mayo, M. Marchiori and J. Filipe. SciTePress, pp. 124–132. DOI: 10.5220/0010710300003058.

Schopf, T., Braun, D. and Matthes, F. (2021b). 'Semantic Label Representations with Lbl2Vec: A Similarity-Based Approach for Unsupervised Text Classification'. In: *Web Information Systems and Technologies - 16th International Conference, WEBIST 2020, November 3-5, 2020, and 17th International Conference, WEBIST 2021, October 26-28, 2021, Virtual Events, Revised Selected Papers.* Ed. by M. Marchiori, F. J. D. Mayo and J. Filipe. Vol. 469. Lecture Notes in Business Information Processing. Springer, pp. 59–73. DOI: 10.1007/978-3-031-24197-0_4. URL: https://doi.org/10.1007/978-3-031-24197-0%5C_4.

Schopf, T., Braun, D. and Matthes, F. (2022a). 'Evaluating Unsupervised Text Classification: Zero-shot and Similarity-based Approaches'. In: *Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval, NLPIR 2022, Bangkok, Thailand, December 16–18, 2022.* ACM, pp. 6–15. DOI: 10.1145/3582768.3582795.

Schopf, T., Dresse, K. and Matthes, F. (2022b). 'Towards AI Platforms for Stationary Retail'. In: *2022 5th International Conference on Artificial Intelligence for Industries (AI4I).* DOI: 10.1109/AI4I54798.2022.00012.

Schopf, T., Klimek, S. and Matthes, F. (2022c). 'PatternRank: Leveraging Pretrained Language Models and Part of Speech for Unsupervised Keyphrase Extraction'. In: *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2022, Volume 1: KDIR, Valletta, Malta, October 24–26, 2022.* Ed. by F. Coenen, A. L. N. Fred and J. Filipe. SCITEPRESS, pp. 243–248. DOI: 10.5220/0011546600003335.

Schopf, T., Weinberger, P., Kinkeldei, T. and Matthes, F. (2022d). 'Towards Bilingual Word Embedding Models for Engineering: Evaluating Semantic Linking Capabilities of Engineering-Specific Word Embeddings Across Languages'. In: *MSIE 2022: 4th International Conference on Management Science and Industrial Engineering, Chiang Mai Thailand, April 28–30, 2022.* ACM, pp. 407–413. DOI: 10.1145/3535782.3535835.

'Semantic networks' (1992). In: *Computers & Mathematics with Applications* 23.2, pp. 1–50. DOI: 10.1016/0898-1221(92)90135-5.

Shen, Z., Ma, H. and Wang, K. (2018). 'A Web-scale system for scientific knowledge exploration'. In: *Proceedings of ACL 2018, System Demonstrations.* Association for Computational Linguistics, pp. 87–92. DOI: 10.18653/v1/P18-4015.

Singh, M. P., Agarwal, P., Chaudhary, A., Shroff, G., Khurana, P., Patidar, M., Bisht, V., Bansal, R., Sachan, P. and Kumar, R. (2018). 'KNADIA: Enterprise KNowledge Assisted DIAlogue Systems Using Deep Learning'. In: *34th IEEE International Conference on Data Engineering,* *ICDE 2018, Paris, France, April 16–19, 2018*. IEEE Computer Society, pp. 1423–1434. DOI: 10.1109/ICDE.2018.00161.

Singhal, A. (2012). *Introducing the knowledge graph: Things, not strings*. Google Blog. URL: https://blog.google/products/search/introducing-knowledge-graph-things-not/.

Suchanek, F. M., Kasneci, G. and Weikum, G. (2007). 'Yago: A Core of Semantic Knowledge'. In: *Proceedings of the 16th International Conference on World Wide Web, WWW'07, Banff, Alberta, Canada*. Association for Computing Machinery, pp. 697–706. DOI: 10.1145/1242572.1242667.

Wang, K., Shen, Z., Huang, C., Wu, C., Dong, Y. and Kanakia, A. (2020). 'Microsoft Academic Graph: When experts are not enough'. In: *Quantitative Science Studies* 1.1, pp. 396–413. DOI: 10.1162/qss_a_00021.

Wang, Q., Yavuz, S., Lin, X. V., Ji, H. and Rajani, N. (2021). 'Stage-wise Fine-tuning for Graph-to-Text Generation'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*. Association for Computational Linguistics, pp. 16–22. DOI: 10.18653/v1/2021.acl-srw.2.

Wu, G., Xu, B., Qin, Y., Kong, F., Liu, B., Zhao, H. and Chang, D. (2021). 'PatentMiner: Patent Vacancy Mining via Context-Enhanced and Knowledge-Guided Graph Attention'. In: *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers New Infrastructure Construction - 6th China Conference, CCKS 2021, Guangzhou, China, November 4-7, 2021, Proceedings*. Ed. by B. Qin, Z. Jin, H. Wang, J. Z. Pan, Y. Liu and B. An. Vol. 1466. Communications in Computer and Information Science. Springer, pp. 227–239. DOI: 10.1007/978-981-16-6471-7_17.

Wu, L., Petroni, F., Josifoski, M., Riedel, S. and Zettlemoyer, L. (2020). 'Scalable Zero-shot Entity Linking with Dense Entity Retrieval'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 6397–6407. DOI: 10.18653/v1/2020.emnlp-main.519.

Yu, D., Zhu, C., Fang, Y., Yu, W., Wang, S., Xu, Y., Ren, X., Yang, Y. and Zeng, M. (2022). 'KG-FiD: Infusing Knowledge Graph in Fusion-in-Decoder for Open-Domain Question Answering'. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pp. 4961–4974. DOI: 10.18653/v1/2022.acl-long.340.

Zhang, N., Deng, S., Sun, Z., Wang, G., Chen, X., Zhang, W. and Chen, H. (2019). 'Long-tail Relation Extraction via Knowledge Graph Embeddings and Graph Convolution Networks'. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pp. 3016–3025. DOI: 10.18653/v1/N19-1306.

Zhou, X., Gong, R., Shi, F. and Wang, Z. (2020). 'PetroKG: Construction and Application of Knowledge Graph in Upstream Area of PetroChina'. In: *Journal of Computer Science and Technology* 35.2, pp. 368–378. DOI: 10.1007/s11390-020-9966-7.

Chapter 16

AdoScript: A DSL for Developing Conceptual Modeling Methods

Dimitris Karagiannis

To support modeling method engineers, this work adopts the paradigm of domain-specific languages and proposes a specialization of it labelled as 'domain-specific language for modeling method realization' (MM-DSL), which is demonstrated by the instance of AdoScript – a language that has been the key enabler for numerous modeling methods and tools developed over the years. An MM-DSL builds on foundational meta-concepts and answers requirements that are specific to the domain of conceptual modeling. This chapter presents the modeling method engineering language AdoScript and how it meets those requirements. It is complemented by the more abstract languages and formalisms MetaMorph and FDMM which serve for the definition and specification of modeling methods according to the Generic Modeling Method Framework (GMMF). Together, this trilogy of languages and their key meta-concepts are operationalized on a concrete metamodeling platform, ADOxx.

16.1 Introduction

To push forward a scientific community like the one of Conceptual Modeling, it is not only necessary but existential to have also understandable, usable, and applicable standards and software platforms. Conceptual Modeling is closely related to disciplines like Software Engineering, Web Design, and Cloud Computing, which have specific standards like MOF, HTML, and ArchiMate. While Conceptual Modeling has previously adopted standards like ER and BPMN for modeling perspectives, it currently lacks standards for language engineering, similar to HTML, and development platforms like Eclipse.

For this reason, we propose in this chapter a starting point targeting a standard for modeling method engineering. This should serve as invariant over time, of course with adaptions according to technological progress regarding the concrete means of realization and operationalization. The chapter introduces AdoScript, a DSL for realizing modeling methods (i. e., a MM-DSL) that encompasses those invariant aspects and meets the requirements pertaining to them. The key meta-concepts fundamental for the proposed language will be examined; they can be interpreted as requirements for a MM-DSL that targets productivity benefits for modeling method implementation, the typical result being a conceptual modeling tool.

In Section 16.2, we give an overview of the relevant notions like domain-specific conceptual modeling and modeling method engineering. The subsequent Section 16.3 then

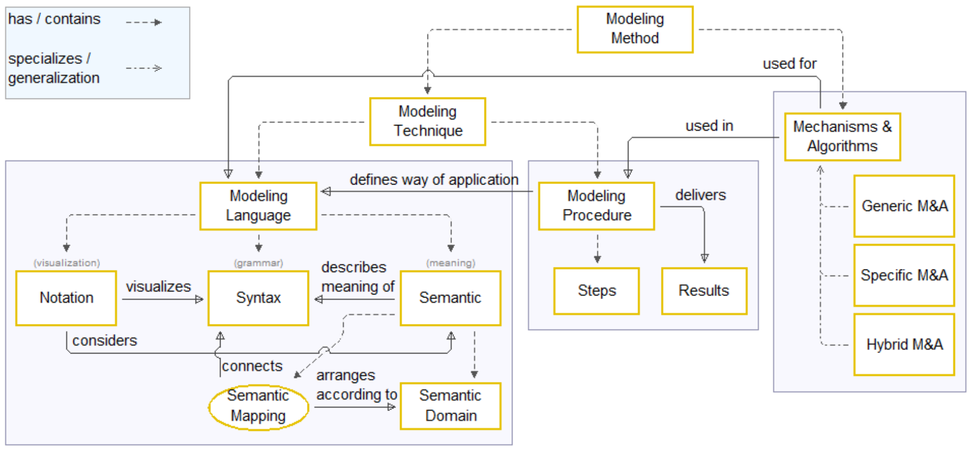


Figure 16.1: The Generic Modeling Method Framework (GMMF)

introduces the meta-concepts of an implementation-oriented MM-DSL and a discussion on how AdoScript supports them towards the realization of modeling methods. In Section 16.4, we demonstrate the operationalization of these meta-concepts in the ADOxx metamodeling platform using the concrete case of the Bee-Up tool. We will conclude in Section 16.5 with a summary and conclusions.

16.2 Background

16.2.1 Modeling Methods and MM-DSLs

Modeling Method Domain-Specific Languages (MM-DSLs) have been designed to simplify the realization of modeling methods. They are completely platform-independent, self-documenting, and easy-to-learn languages for describing modeling method components: a metamodel, graphical representation, and algorithms. Once described, parts of code can be easily reused with or without previous modification. For example, a part of the metamodel can be reused to define another metamodel; an already-defined graphical representation can be slightly modified to be reused. In theory, MM-DSL code can be compiled and executed on any metamodeling platform, if a compiler for that particular platform exists. This has been tested using the ADOxx metamodeling platform. The current MM-DSL specification is described using EBNF and available on the OMiLAB website.[1]

A MM-DSL, the DSL class to which AdoScript belongs, is a domain-specific language for realizing modeling methods and making them available as modeling tools. This requires an understanding of what a modeling method is structurally made of, which will hint at the first-class constructs to be manipulated in such a DSL. Following the definition in Karagiannis and Kühn (2002) where the GMMF (Generic Modeling Method Framework) is introduced, a modeling method consists of a modeling language, a modeling procedure and mechanisms and algorithms as illustrated in Figure 16.1.

1 `https://www.omilab.org/adoscript/` (accessed 06.03.2023)

The modeling language covers the main aspects needed to build a model: the rules for creating models through their syntax, the meaning of concepts through their semantics and their (visual) representation through notation. Additionally, a modeling procedure describes steps on how to use the modeling language to achieve certain results. Mechanisms and algorithms are further used to support the procedure through processing of models, like analysis, generation or application of behavioral semantics. A similar definition is found in Frank (2011), with less emphasis on the mechanisms and algorithms:

> *'A modelling method is a specific kind of method. It is aimed at solving a class of problems through the design and use of models. It consists of at least one modelling language and at least one corresponding process model which guides the construction and analysis of models. In addition to that it includes assumptions about successful patterns of action'* (Frank 2011, p. 40).

A modeling method is not only for the creation of (conceptual) models, but also for their use as a means to solve a problem, to achieve a result determined by goals identified in its community of practice – something that the literature labelled as the 'pragmatics of conceptual models' (Thalheim 2012), complementing the syntax and semantics (covered by the modeling language). An approach for creating and iteratively evolving a modeling method is AMME (Agile Modeling Method Engineering) described in Karagiannis (2018). This chapter focuses on one specific step of AMME: the implementation of the modeling method as a tool ('Develop' step of the lifecycle). The implementation of a modeling method represents its formalization as something that can be executed with tool support.

16.2.2 Metamodels for Method Design

An appealing approach for specifying a modeling method is through a metamodel. Frank states two reasons for that: First, they are based on the same paradigm as the models themselves. Hence, prospective language designers should be more familiar with them than with grammar. Second, they provide a better foundation for designing corresponding modeling tools than grammars, because they can be transformed into object models in a straightforward way (Frank 2011, p. 31). Here, we are aiming to use metamodels to specify not only a modeling language, but as much of a modeling method as possible. To create a metamodel another language must be used: A metamodeling language (cp. Frank 2011, Karagiannis and Kühn 2002). It allows to instantiate the concept of a model from a higher level of abstraction: the meta^2-model. An example for a part of a generalized and simplified meta^2-model is shown in Figure 16.2. It omits many details as it focuses on a few concepts that will be used in later examples. Seven of the elements focus on the modeling language aspect. Two of them are relevant for the modeling procedure and the mechanisms and algorithms, namely 'Mechanism/Algorithm/Functionality' and 'Trigger'.

The meta^2-model is crucial, since it supplies the higher-level concepts that can be used, for example by AdoScript (cp. 'Purpose of a Meta-Metamodel' from Frank 1998). Thus, it specifies what an MM-DSL like AdoScript can work with. For example, if the meta^2-model does not have the concept of an 'attribute type', then a metamodel will not be able to specify the type of an attribute. In other words, the meta^2-model specifies the concepts available in a metamodeling language. Together with a procedure like the one described in Karagiannis (2018) they satisfy the definition of a metamodeling method: A metamodeling method is a modeling method that consists of a metamodeling language and a corresponding process

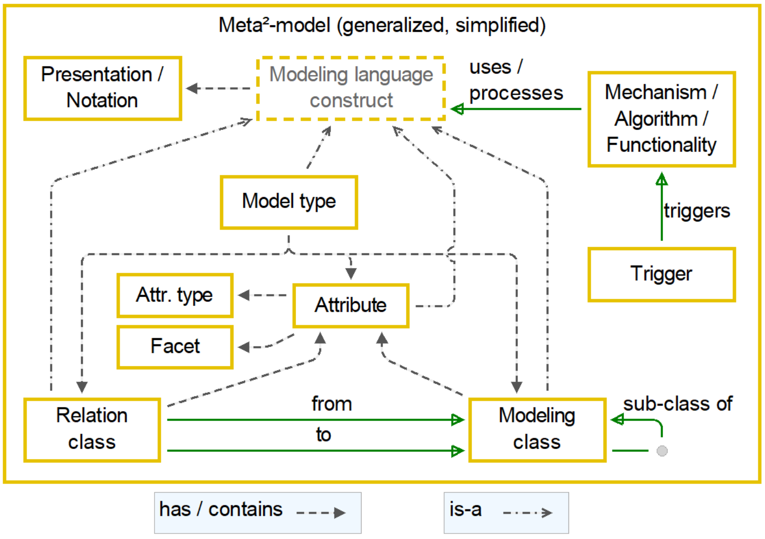


Figure 16.2: A part of a generalized and simplified meta²-model

model that guides the specification of metamodels, which in turn may serve the specification of a modeling language (Frank 2011, p. 40).

16.2.3 Domain-Specific Modeling Languages

Domain-Specific Modeling Languages (DSML) are tailored for the concrete needs of their stakeholders and comprise the concepts established in the domain and relevant for the modeling goals. Frank defines the term 'DSML' as follows:

> *'A DSML is a modelling language that is intended to be used in a certain domain of discourse. It enriches generic modelling concepts with concepts that were reconstructed from technical terms used in the respective domain of discourse. A DSML serves to create conceptual models of the domain it is related to'* (Frank 2011, p. 28).

In the state-of-the-art of conceptual modeling DSMLs gain increasing importance due to a range of benefits over the so-called general-purpose modeling languages (GPML) (Frank 2013). These GPMLs, often standardized like the UML, sacrifice specificity for wide acceptance and reusability across domains. Nevertheless, the difference between DSMLs and GPMLs is ill-defined and the gradations in between are fluid. The degree of specificity can vary for the same basic concept in different languages – from a plain notational symbol carrying all the semantics in a label (e. g., an activity in UML) to a highly specified concept enriched with diverse semantics relevant in the domain (e. g., an activity enriched with all the information necessary for simulation), see Figure 16.3. Buchmann suggests the use of formal concept analysis to describe this gradation between low and high specificity and even goes one step further by proposing the Purpose-Specificity framework putting purpose and specificity as orthogonal dimensions for conceptual modeling languages (Buchmann 2022).

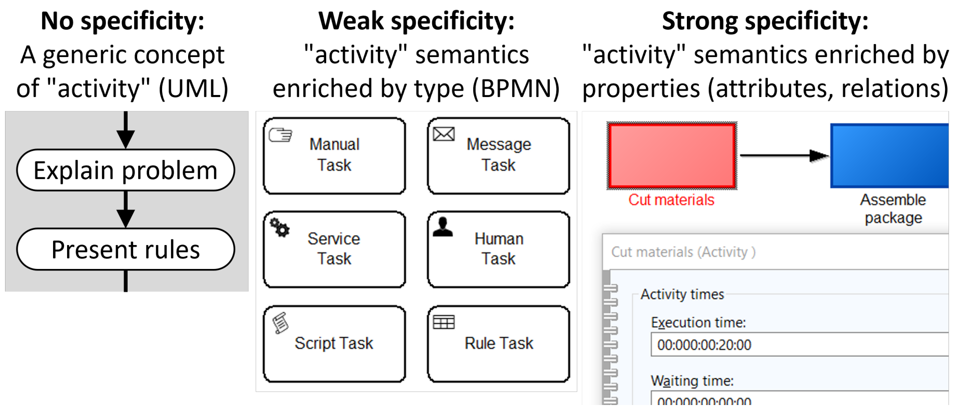


Figure 16.3: Degrees of Specificity

DSMLs overcome drawbacks of general-purpose languages in diverse ways: Firstly, DSMLs offer a vocabulary appropriate for the domain expert without the need to somehow encode meaning in semantic primitive concepts like *class*. Furthermore, they support the correct usage of the language by domain-specific constraints and procedures (Frank 2014). DSMLs offer semantics precisely corresponding to the domain terminology, rather than ad hoc human interpretation of instance names or hacking of semantics by abusing semantically incongruous concepts, e. g., the abuse of notes in BPMN for capturing missing domain concepts such as the ingredients of a cooking recipe exemplified in Buchmann et al. (2019). The more specific the semantics a language offers, the more gain the user has, but on the other hand a precise tailoring for a narrow domain or purpose inevitably confines the area of application – this is not a drawback of DSMLs but a trade-off of benefits (Buchmann 2022). Secondly, the fast adaption to new challenges and changing requirements can be more agile with properly maintained DSMLs than by following the cumbersome progression of standard versioning for regulated GPMLs (Karagiannis 2018). Lastly (not raising the claim to completeness), the graphical notation of domain concepts can be tailored for domain experts and the value of the graphical visualization can be enhanced for improved semantic transparency – a key property recognized by scientific assessments of the visual notation (Moody 2009).

Another benefit of DSMLs is the fact that they usually not only comprise a language but are comprised by modeling methods as previously defined, i. e., a method also offers mechanisms and algorithms to process model contents. This is reflected in the GMMF depicted in Figure 16.1 referencing the language (which can be a DSML) as one of three defining building blocks: the language, the modeling procedure, and the mechanisms and algorithms.

DSMLs in the Domain of Conceptual Modeling

We can also apply the power of modeling to itself by using a DSML for designing DSMLs, i. e., we move a step back and use a DSLM tailored for the domain of conceptual modeling. This procedure is typically also subsumed under the term metamodeling. This brings with it some peculiarities: Similar to language theory we investigate the subject by applying concepts and methods from the subject to itself. The investigator becomes the investigated. The produced output of using such a language is a model of a DSML, e. g., a model of ArchiMate or a model of BPMN (not to be confused with a concrete ArchiMate model or BPMN model).

The domain of discourse in this case covers terms for language specification, language concept definition, modeling process description and many more. For the first part, the language specification, we have to cover terminology for all components of a modeling method according to the GMMF. The syntax for language concept definition is captured in a metamodel constituted by object types, relation types and attributes. To capture the semantics there are a couple of methods at hand, e. g., ontologies or knowledge graphs. For the rest of a modeling language there do not exist established methods for specification. Such a modeling method created for modeling language design is implemented in the tool CoChaCo (Karagiannis et al. 2019). It supports method engineering activities – from the collection of modeling requirements to the definition of modeling functionality, with language design (traceable to requirements and functionality) as a core activity.

Domain-Specificity for Engineering Productivity

A highly successful field working with DSMLs (and textual DSLs) is Software Engineering typically aiming at automated generation of code or configuration. DSMLs are seen as the core enabler for increased productivity (Kelly and Tolvanen 2007). This is due to the fact that the domain-specificity allows for modeling with concepts from the problem domain not the solution domain and the complexity of the mapping between them can be transferred to a code generator. Kelly and Tolvanen (2007) name a narrow focus captured by a high level of abstraction as the core characteristic of DSMLs.

The importance of DSMLs beyond Software Engineering is underpinned by the frequent development of new methods with highly specialised semantics in a diversity of domains. In the OMiLAB community, there were more than 40 new methods presented in the last six years for the domains of education, IoT, or business ecosystems, to name only a few (Karagiannis et al. 2016b; Karagiannis Dimitris et al. 2022). These methods usually carry sophisticated functionality that extends the value of models beyond mere visual representations supporting human comprehension. Obviously, the tailoring of the method for the domain enables algorithms to take richer input from a model. Examples of domain-specific functionality are the digitalization of haptic models Muck and Palkovits-Rauter 2022, IoT simulation Choe and Lee 2016, robot controlling Morita and Yamaguchi 2022, assisted living support Mayr et al. 2016, or the linking of stochastic diagrams for didactic use Döller and Götz 2022. AdoScript, the MM-DSL in the focus of this chapter, is the key enabler for building all these functionalities. It fulfils the productivity goal of DSLs/DSMLs by allowing the method engineer to manipulate constructs with built-in properties that are specific to modeling methods instead of using generic classes and objects.

In line with the Memorandum on design-oriented information systems research (Österle et al. 2011) the development of DSMLs should be guided by the respective research principles proposed therein. While for DSLs unrelated to modeling there are established language development practices (e. g., Fowler 2011 for DSLs in Software Engineering), there are only a few for DSMLs. One contribution explaining the micro-steps in the DSML development process is the work of Frank on requirements and design guidelines (Frank 2013). A similar approach by Michael and Mayr exemplifies the micro-steps on an example (Michael and Mayr 2016). Another process model comprising the macro-steps in method development

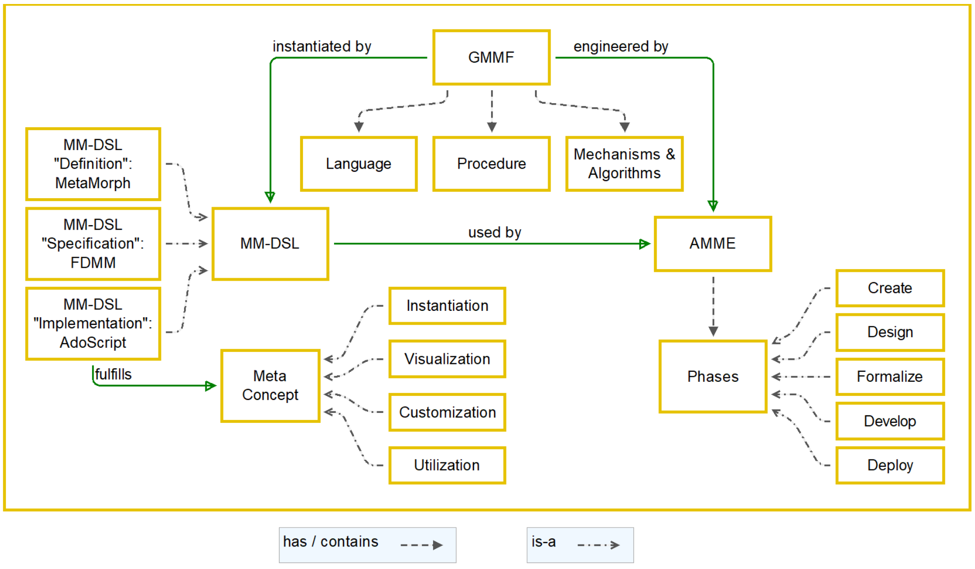


Figure 16.4: Interdependencies between GMMF, MM-DSL, and AMME

is the aforementioned AMME framework with a focus on the agility and dynamicity of modeling languages (Karagiannis 2018). Figure 16.4 provides a graphical representation for better understanding the relation of AdoScript MM-DSL with GMMF and AMME. The remainder of this chapter will discuss the meta-concepts that AdoScript enables for the method engineer, which can be considered requirements for an implementation-oriented MM-DSL.

16.3 AdoScript: The Third Pillar in a Language Trilogy of MM-DSLs

AdoScript is the third pillar in the trilogy of MM-DSLs as described in Döller et al. (2023) and shown in Figure 16.5. Besides METAMORPH, a formalism comprising a definition and theory for modeling methods (Döller 2022; Döller et al. 2023), and FDMM, a specification language for modeling methods (Fill et al. 2012), AdoScript targets the implementation-specific perspective on the realization of a modeling method. The interrelation between the three languages is clarified by the following explanation:

- A formal definition offers means to study modeling methods in general, research their characteristics, and deduce a theory of conceptual modeling. This is comparable to the definition of a graph and the knowledge stack on graph theory.
- A formal specification offers a method for uniquely putting modeling methods on record. This is comparable to the different possibilities of representing a graph, e. g., as visual constructs, as a set of nodes and edges, as an adjacency matrix, or as an incidence matrix.
- A formal implementation provides an implementation-specific runnable code. This is comparable with the implemented data structure of a graph as it is used in a computer, typically exposed through a programming library or API.

MetaMorph

The formalism MetaMorph provides a formal definition of modeling methods. This definition is based on a structural theory canonically reflecting the constructs of conceptual modeling, this is model theory as defined in mathematical logic. By defining modeling languages as a subset of formal languages we gain a structural foundation that offers means to investigate common features of modeling methods and a whole toolbox of methods to approach open research questions Döller 2022. This goes beyond the objective of providing a formal specification language and allows for the development of a well-founded structural theory of the scientific field of conceptual modeling.

FDMM

For the formal specification of modeling methods and models FDMM (short for Formalism for Describing ADOxx Meta Models and Models) is introduced Fill et al. (2012). The goal is to provide a formal syntax for method engineers without the requirement for a strong mathematical background. It is based on set theory to capture the components of modeling languages. It comes with a definition of metamodels and models, although its purpose is restricted to providing a formal specification syntax for domain-specific modeling languages implemented on ADOxx. The resulting specification can serve as input for the implementation of the metamodels and models.

AdoScript

Finally, AdoScript is an implementation-oriented MM-DSL in the sense that its typical outcome is a modeling tool that makes operational a modeling method specification. Such a MM-DSL must fulfil a set of requirements to be synthesised in the following Sections in four foundational meta-concepts: Instantiation, Visualization, Customization, and Utilization (Table 16.1). These meta-concepts and the constructs derived from them make AdoScript a platform-independent language – although it has been traditionally used through its ADOxx implementation, its constructs are sufficiently generic to be translated for any development platform; an example of an alternative syntax detached from ADOxx was defined in EBNF and implemented as an Eclipse compiler in Visic (2016) and Visic et al. (2015).

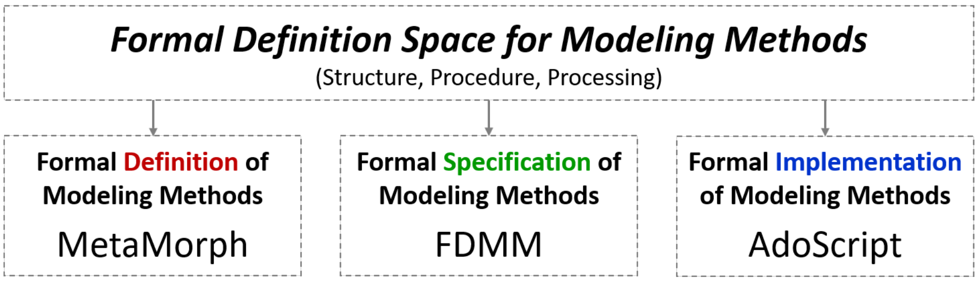

Formal Definition Space for Modeling Methods
(Structure, Procedure, Processing)
Formal Definition of Modeling Methods
MetaMorph
Formal Specification of Modeling Methods
FDMM
Formal Implementation of Modeling Methods
AdoScript

Figure 16.5: Different layers of modeling method formalization, adapted from Döller et al. (2023)

16.3.1 Meta-Concepts of an implementation-oriented MM-DSL

The goal of this section is to introduce the platform-independent underlying meta-concepts potentially acting as requirements for MM-DSLs, specifically for those DSLs supporting the implementation of modeling methods. Later we will show how these are fulfilled by AdoScript and demonstrated on the ADOxx metamodeling platform. We derive these meta-concepts from the Modeling Mantra based on model value co-creation (Strecker et al. 2019) as well as from the experience resulting from the developed modeling method artifacts available in the OMiLAB repository (`https://www.omilab.org/activities/projects/`): *We use abstraction to reduce complexity in a domain for a specific purpose.* The four meta-concepts are based on the corresponding phrase of this mantra as shown in Table 16.1 and will be introduced in detail in the following. In detail, these concepts involve enabling activities and

Modeling Mantra		Meta-concepts
We use abstraction	→	*Instantiation*
to reduce complexity	→	*Visualization*
in a domain	→	*Customization*
for a specific purpose	→	*Utilization*
→ means *primarily influenced by*		

Table 16.1: Mapping of Modeling Mantra to Meta-Concepts

their results in the act of modeling method realization, see Table 16.2. AdoScript provides various constructs and functionalities to support the definition of a conceptual modeling language, here focused on DSMLs, the implementation of mechanisms and algorithms and the tool-level user interaction during the modeling procedure. It specifies a metamodel by instantiating the concepts from a meta^2-model, some of which have been shown in Figure 16.2.

Meta-Concept	Enables	Results
Instantiation	Metamodel design	Metamodel
Visualization	Metamodel enactment/depiction	Notation, user interface, data container
Customization	Metamodel enrichment specification, adapted functionality	Domain semantics & analytics
Utilization	Metamodel-driven programming/-scripting	Operations

Table 16.2: Meta-Concepts of a DSL for modeling method realization

AdoScript sources can be implemented on a metamodeling platform (see subsequent section) and can be roughly categorized into two groups: (1) Passive components which use mostly declarative code to describe structural elements of the modeling method, like the types of objects and relations, the presentation of attributes and their values, the available triggers for functionalities etc., and (2) aActive components which use mostly imperative code to describe behavioral parts of the modeling method, like the scripting to realize a functionality, how to draw the dynamic graphical notation of an object, how to assist a user action during the modeling procedure (e. g. Ternes et al. 2021) etc. While such a categorisation is useful to consider the mode of thinking when writing AdoScript code, it is the domain-specific constructs (for the domain of conceptual modeling) that the language provides that allow achieving the desired results. The following lists several of the relevant AdoScript constructs mapped on the building blocks of the GMMF.

- For modeling language definition, using mostly passive/declarative code:
  - Instantiation of modeling class, relation class, attribute, and model type.[2]
  - Generalization and inheritance between instances.
  - Composition or containment between instances.
  - Visual representation of instances.
- For mechanism and algorithm implementation, using mostly active/imperative code:
  - Processing, like transformation or analytics in a context (mining, learning, optimization, execution etc.)
  - Actions through commands and procedures to execute.
  - Data through values and variables to modify.
  - Coupling to external services and other technologies, like neural networks.
- For modeling procedure interaction, using both passive and active code:
  - Explicit interaction with a user or system realized through mechanisms.
  - Implicit interaction between the user and modeling language constructs provided by the platform applied for interpretation of AdoScript.
  - Interaction through triggers, like menu items or events.

These constructs are then used during the task of realizing the envisioned modeling method. The following is a list of recommended steps for creating a metamodel with AdoScript:

1. Define the main modeling elements by instantiating “modeling class”, “relation class” and “attribute” and specify their relation to each other.
2. Detail the modeling elements, like their graphical notation, applicable restrictions etc.
3. Define the model types and assign modeling elements to them.
4. Realize functionalities and model processing with mechanisms / algorithms and configure their triggers.

These constructs are then used during the task of realising the envisioned modeling method. The following is a list of recommended steps for creating a metamodel with AdoScript:

1. Define the main modeling elements by instantiating ‘modeling class’, ‘relation class’ and ‘attribute’ and specify their relation to each other.
2. Detail the modeling elements, like their graphical notation, applicable restrictions etc.
3. Define the model types and assign modeling elements to them.
4. Realize functionalities and model processing with mechanisms / algorithms and configure their triggers.

Common sense dictates the general order of these tasks, i. e., a modeling class can not be assigned to a model type before it exists. However, these tasks can be iteratively repeated, aligning with the principles of design-oriented information systems research, ultimately leading to the development of a modeling tool.

2 The instances talked about here are located on the metamodel level.

16.3.2 The Operationalization of MM-DSL Meta-Concepts in AdoScript

As mentioned earlier, AdoScript is an implementation-oriented MM-DSL that makes the hereby introduced meta-concepts operational.

Instantiation. *Design of the metamodel of a modeling method by instantiating the corresponding meta^2-model.* The method engineer has to come up with the concepts relevant for the concrete domain and purpose for which the method will be applied. Considering the platform-independent status of AdoScript as a MM-DSL we are free to use concepts in the meta^2-model according to the current needs and are not restricted by the predefined meta^2-model of a concrete platform. This results in the freedom to include, e. g., multiple inheritance for object types and relation types, multiple endpoints of relations, and relations between relations. Implementation-specific restrictions will later apply once a platform is adopted, depending on that platform's capabilities or design decisions. ADOxx is an example of such a platform, with restrictions imposed by its own meta^2-model.

Visualization. *Enactment of the metamodel and provision of a visual interaction facility.* This on the one hand comprises the graphical representation of concepts but also a look-and-feel blueprint for the actual interaction of the modeller with the model, which includes a visual component for manipulating a concept's schema (its machine-readable properties attached to symbolic elements). This makes use of a scripting language for the notation of the concepts in the metamodel, by offering a template for the appearance/visualisation of the modeller's tool and an editing facility for data structures associated with all modeling elements according to their metamodel definition. Again, different platforms may choose to present these in different ways with respect to user interaction modalities.

Customization. *Adapt predefined structures and functionality for a diversity of domains.* Some typical predefined features that use models beyond comprehensible visualization are querying and analysing model content. Of course, the concrete form of operationalization in various modeling methods can differ heavily. Therefore, these functionalities must be open to domain-specific adaption and customization. This also applies to the data structures of the modeling elements as described above. Their design must be open to customization to allow for the required depth of specificity.

Utilization. *Implement functionality that extends the pragmatics of models.* To go beyond the capabilities of customisation, the method engineer must be equipped with a scripting language for processing the model and for offering suitable capabilities to extend what Thalheim (2012) refers to as the 'pragmatics of modeling', i. e., the use of a DSML as required by domain needs. This scripting language therefore must provide facilities for accessing and modifying models, modeling elements, and their structure, for interacting with the modeller, and for connectivity with external sources and services. In this context, it should be taken into consideration that also general-purpose programming languages can be used to extend the method capabilities if the platform of choice supports those languages directly or through interoperability channels. The key specificity imposed by an MM-DSL like AdoScript is that either the input, the output or both for any functionality will be structures emerging from the modeling language prescriptive design, i. e., model elements, model fragments compliant with the metamodel, data structures attached to model elements.

Meta-Concept	**MM-DSL** What I do	**AdoScript** How I do it	**ADOxx** I do it
Instantiation	Design	What has to be designed to enable model content creation?	Design of the metamodel by instantiating the ADOxx meta^2-model.
Visualization	Define	What has to be defined to interact with the method components?	Metamodel enactment via providing a visual notation to visualize the model, a Notebook for structured specification of element attributes, and a template for the overall layout of the modeller's interface.
Customization	Adapt	How to adapt the components? What adaptions does the domain require?	Possibilities to customize the Notebook design, to specify customized queries, and to customize analysis tools.
Utilization	Extend	How to extend the pragmatics of concepts? What operations does the domain require?	A scripting language is provided offering message ports and event handling for the interaction with the model and the modeling environment.

Table 16.3: What activities we do with AdoScript

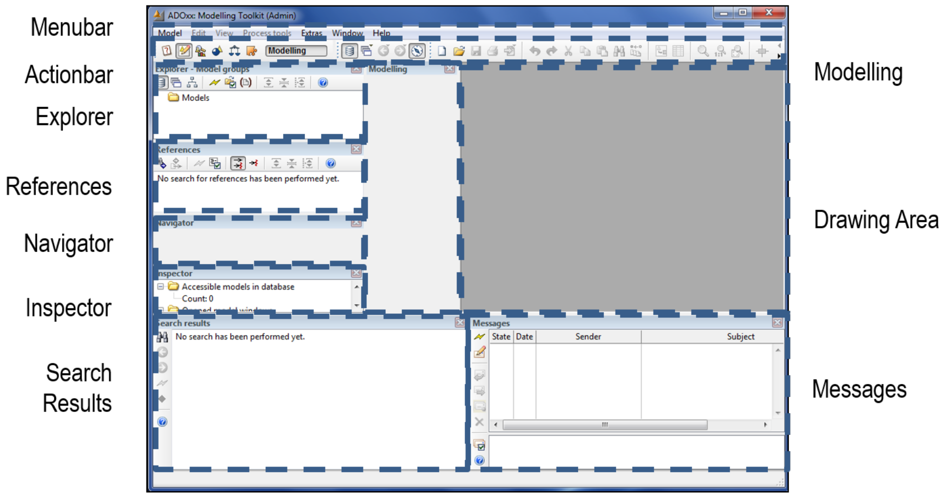
Auxiliary. Besides the previously introduced meta-concepts, auxiliaries may support method deployment or the interoperability of modeling tools. These are for example services to support the deployment of a modeling environment, or an automatic export of models in standardized formats like XML, or HTML.

The meta-concepts and the concrete tasks of the method engineer when working with AdoScript are summarised in Table 16.3 including the concrete employment of the specific platform ADOxx for performing them. This will be further outlined in the next subsection.

16.3.3 The Implementation of the AdoScript Meta-Concepts in ADOxx

ADOxx is a platform to support the realization of conceptual modeling method tools based on a *metamodeling approach.* ADOxx offers one particular implementation of the AdoScript meta-concepts – currently the most mature and widely adopted, although translatable to implementations for other platforms as Visic (2016) shows. To serve the desire for productivity or DSLs in general, the ADOxx implementation offers certain built-in functionalities that operationalize each meta-concept, i. e., it can be adapted to productively build modeling tools in the domain-specific terms of the hereby proposed MM-DSL.

Instantiation. *Design of the metamodel of a modeling method by instantiating the corresponding meta^2-model.* In the case of ADOxx this is the meta^2-model of ADOxx. This concrete



No search for references has been performed yet.

No search has been performed yet.

Figure 16.6: The deployment template of ADOxx

meta²-model offers predefined classes with pre-implemented behavior, e. g., containment, and imposes constraints on the design of the metamodel by, e. g., not allowing inheritance on relation types.

Visualization. *Enactment of the metamodel and provision of a visual interaction facility.* In the case of ADOxx, this includes the GraphRep scripting language for the vector graphics notation of the concepts in the metamodel, the deployment template for the appearance/visualization of the modeller's tool, the UI interaction facility and the deployment template of ADOxx depicted in Figure 16.6, and the Notebook functionality of ADOxx providing an interaction facility for manipulating attributes of a modeling element in a structured way. Some modeling methods also require a dynamic extension of notation, which is also supported by the GraphRep language.

Customization. *Adapt predefined structures and functionality for a diversity of domains.* These comprise querying mechanisms on models, analysing tools, and the design of the interaction facility for data structures. To meet these requirements ADOxx offers the querying language AQL, which allows for the provision of domain-specific searches in the model or model repository. Furthermore, predefined process analysis functionality exists for different perspectives, e. g., time, capabilities, or capacities. To meet the appropriate structure of model element data ADOxx allows for the customization of the Notebook that acts as a panel of editable properties, hyperlinks, and triggers associated with each model element.

Utilization. *Implement functionality that extends the pragmatics of models.* In ADOxx, there exists the scripting language for building functionality that reads or writes model content. It has a simple scripting syntax for general commands and APIs for model content manipulation, interaction with the modeling environment, calling predefined functionality, and communicating with external sources and services.

Auxiliary. ADOxx comes with a deployment service – the PDP Product Development Packaging – as well as with predefined export functionality in a platform-specific format (ADL), in HTML, and XML. An RDF generation mechanism that makes models available to semantic queries or reasoning is also available for all DSMLs developed on ADOxx (Karagiannis and Buchmann 2018).

16.4 Demonstration of the Meta-Concepts in the Bee-Up tool

The meta-concepts of AdoScript are applied with a fitting development platform to realize a modeling method operationalized as the Bee-Up modeling tool.[3] For this case the metamodeling platform ADOxx[4] version 1.5 with its own implementation of AdoScript was used. Bee-Up is a modeling tool allowing its users to create models in various languages, like BPMN, EPC, ER, UML, Petri Nets etc., and to get additional value from models by processing them: simulation of process models, execution of Petri Net semantics, transformation of model content into RDF, generating SQL code from an ER model etc. It achieves this by using the metamodeling approach driven by the AdoScript MM-DSL. In terms of motivation, the tool was born out of educational concerns, answering recent calls for tooling that can consolidate the status of Conceptual Modeling as a standalone discipline (Buchmann et al. 2019; Rosenthal et al. 2019). To realize Bee-Up, it was necessary to gather its requirements, some described in Karagiannis et al. (2016a). A few of these will be used for the following examples. To structure those requirements according to the modeling method building blocks the GMMF was used.

In Figure 16.7, a part of the requirements for an Entity-Relationship modeling language is shown as a metamodel with an example model. An example for the requirements of a

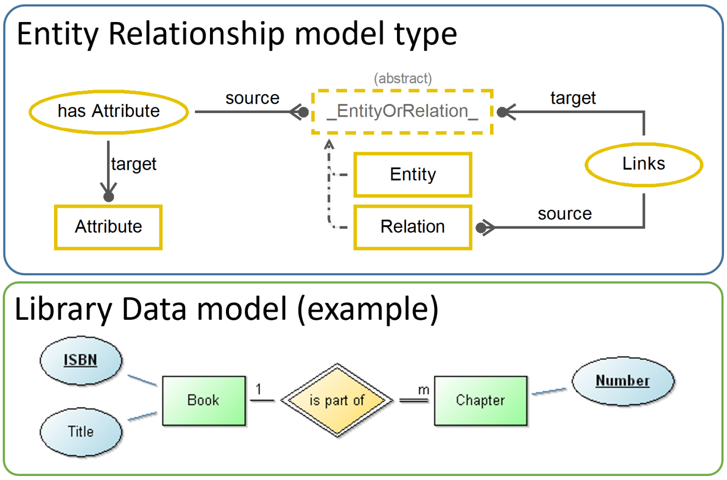
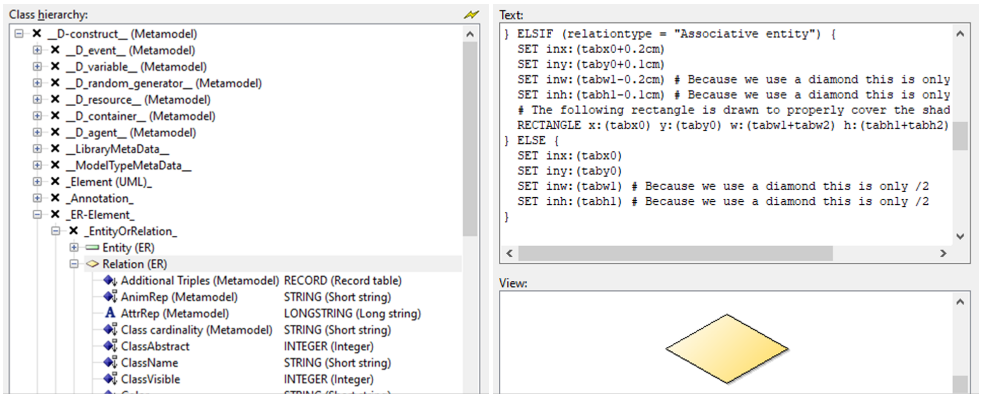


Figure 16.7: Example Entity-Relationship metamodel and model from Bee-Up

mechanism would be: ‘The contents of an ER model are transformed into the corresponding SQL CREATE statements. Entities should be transformed into tables. Attributes should be

3 See https://bee-up.omilab.org/ for details (accessed 06.03.2023).
4 See https://www.adoxx.org/fordetails (accessed 06.03.2023).


Class hierarchy:

```
⊟ ✕ __D-construct__ (Metamodel)
   ⊞ ✕ __D_event__ (Metamodel)
   ⊞ ✕ __D_variable__ (Metamodel)
   ⊞ ✕ __D_random_generator__ (Metamodel)
   ⊞ ✕ __D_resource__ (Metamodel)
   ⊞ ✕ __D_container__ (Metamodel)
   ⊞ ✕ __D_agent__ (Metamodel)
   ⊞ ✕ __LibraryMetaData__
   ⊞ ✕ __ModelTypeMetaData__
   ⊞ ✕ _Element (UML)_
   ⊞ ✕ _Annotation_
   ⊟ ✕ _ER-Element_
      ⊟ ✕ _EntityOrRelation_
         ⊞ Entity (ER)
         ⊟ Relation (ER)
            Additional Triples (Metamodel) RECORD (Record table)
            AnimRep (Metamodel)            STRING (Short string)
            AttrRep (Metamodel)            LONGSTRING (Long string)
            Class cardinality (Metamodel)  STRING (Short string)
            ClassAbstract                  INTEGER (Integer)
            ClassName                      STRING (Short string)
            ClassVisible                   INTEGER (Integer)
```

Text:

```
} ELSIF (relationtype = "Associative entity") {
  SET inx:(tabx0+0.2cm)
  SET iny:(taby0+0.1cm)
  SET inw:(tabw1-0.2cm) # Because we use a diamond this is only
  SET inh:(tabh1-0.1cm) # Because we use a diamond this is only
  # The following rectangle is drawn to properly cover the shad
  RECTANGLE x:(tabx0) y:(taby0) w:(tabw1+tabw2) h:(tabh1+tabh2)
} ELSE {
  SET inx:(tabx0)
  SET iny:(taby0)
  SET inw:(tabw1) # Because we use a diamond this is only /2
  SET inh:(tabh1) # Because we use a diamond this is only /2
}
```

View:

(a) (b)

Figure 16.8: (a) Tree-based editor for class management; (b) Editor for graphical interaction

transformed into columns of the table for the entities they belong to. The primary key of a table from an entity should be based on the attributes belonging to the entity and marked as part of the primary key …' Also a very simple example of requirements for a mostly free modeling procedure is given in this context by common sense: first create the ER model and then perform the transformation, which would use the mechanism by providing a trigger through the user interface.

The further presented examples not only show ADOxx-specific scripting code to fulfil the requirements, but also the various editors available to create and edit parts of the modeling method. In Figure 16.8, two graphical editors are shown. The first editor (a) uses a tree view for managing modeling classes, the inheritance between them and the attributes they have. Here it shows the parents, siblings and children of the Relation␣(ER) modeling class. The second editor (b) allows to define part of the interaction with a modeling class. Specifically, it uses a simple text editor and preview to define the graphical representation of Relation␣(ER). The code visible uses a conditional statement to control what type of relationship should be drawn, an associative entity or a normal relationship, and sets various variables and shapes accordingly.

Instantiation: The following code describes the structure of the metamodel covering modeling language constructs (see Figure 16.8a) using the meta-concept of instantiation.

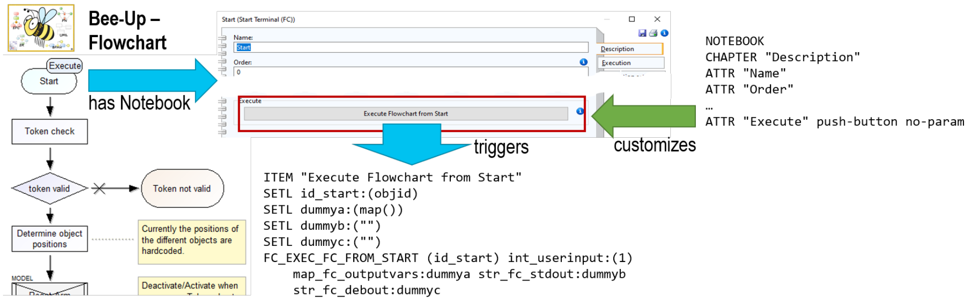
```
CLASS <_EntityOrRelation_> : <_ER-Element_>
    CLASSATTRIBUTE <ClassAbstract>
    VALUE 1

CLASS <Relation (ER)> : <_EntityOrRelation_>
    CLASSATTRIBUTE <ClassAbstract>
    VALUE 0

    CLASSATTRIBUTE <GraphRep>
    VALUE "..."

    ATTRIBUTE <Relation type>
    TYPE ENUMERATION
        FACET <EnumerationDomain>
        VALUE "Normal@IS-A@Associative␣entity@..."
        FACET <AttributeHelpText>
        VALUE "..."

    ATTRIBUTE <Relation type>
    VALUE "Normal"
```

Specifically, the beginning of the code states that the `_EntityOrRelation_` modeling class exists, that it inherits from `_ER-Element_` and that it should not be directly instantiated in models. The latter is achieved by setting the value for the inherited attribute `ClassAbstract` for this class to `1`. Furthermore the code defines the `Relation␣(ER)` modeling class, which inherits from `_EntityOrRelation_`. It also sets the values for two inherited attributes `ClassAbstract` and `GraphRep`. The attribute `Relation␣type` is defined for `Relation␣(ER)` as an enumeration and its details are provided through facets, like which values are allowed or a help-text for the attribute. Afterwards the default value for the attribute as part of the `Relation␣(ER)` modeling class is denoted.

Visualization: To specify the graphical representation of a modeling class for ADOxx (as shown in Figure 16.8b), a more imperative style of code is used. It defines a set of instructions for painting the object and allows to set temporary local variables and use control structures like conditionals and loops on top of vector graphics declarations. The following code snippet is taken from the code for `Relation␣(ER)`. It draws a polygon in the shape of a diamond, sets the font color to use and then either writes a fixed text 'IS-A' or the object's 'Name', depending on what type of relationship the object is. Most of the actual values are taken from variables set previously in the code before this snippet.

```
POLYGON 4
    x1:(inx+inw) y1:(iny)
    x2:(inx+(2*inw)) y2:(iny+inh)
    x3:(inx+inw) y3:(iny+(2*inh))
    x4:(inx) y4:(iny+inh)
FONT color:(fontcolor)
IF (relationtype = "IS-A") {
    TEXT "IS-A" x:(tabx1) y:(taby1) w:c:(tw) h:c:(th)
        line-break:rigorous
} ELSE {
    ATTR "Name" x:(tabx1) y:(taby1) w:c:(tw) h:c:(th)
        line-break:rigorous
}
```

Customization: One of the built-in functionalities provided by the ADOxx platform is the Notebook which shows the values of attributes and allows their editing. These Notebooks can be customized to show the attributes in a specific structure and provide various controls to interact with them. One of these controls is the depiction of a special type of attribute called 'Program Call' as a button. The example shown in Figure 16.9 uses the implementation of Flowcharts in Bee-Up, showing a part of a Notebook for a 'Start Terminal'. The structure



```
NOTEBOOK
CHAPTER "Description"
ATTR "Name"
ATTR "Order"
…
ATTR "Execute" push-button no-param
```

```
ITEM "Execute Flowchart from Start"
SETL id_start:(objid)
SETL dummya:(map())
SETL dummyb:("")
SETL dummyc:("")
FC_EXEC_FC_FROM_START (id_start) int_userinput:(1)
    map_fc_outputvars:dummya str_fc_stdout:dummyb
    str_fc_debout:dummyc
```

Figure 16.9: Customizing the notebook with a trigger button for executing a Flowchart

and controls of the Notebook are customized by using the platform functionality ‘AttrRep’. A part of the ‘AttrRep’ code is shown below.

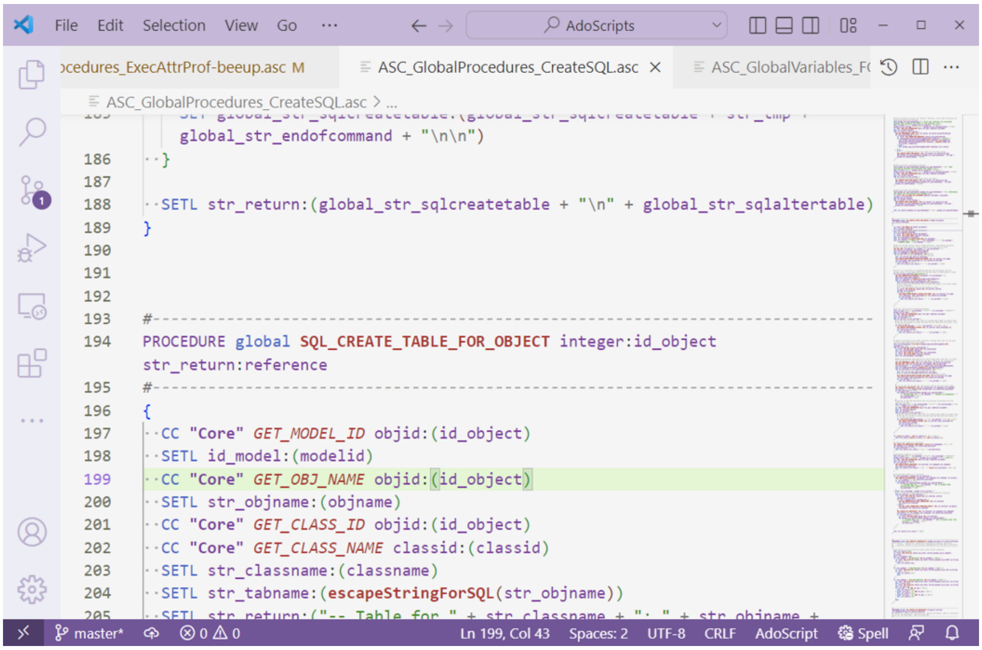
```
NOTEBOOK
CHAPTER "Description"
ATTR "Name"
ATTR "Order"
#... Other attributes added to the notebook.
ATTR "Execute" push-button no-param
```

The beginning of the code has a keyword to start the definition of the Notebook. This is followed by the specification of a chapter ‘Description’ and the attributes it should contain, like ‘Name’ and ‘Order’. The special ‘Program Call’ attribute is in this case called ‘Execute’ and added as a push-button without any additional parameters. A press of the ‘Execute’ button triggers the functionality to execute the operations along the path of a Flowchart. This functionality is implemented as procedures using AdoScript code, which processes the model elements one after the other and performs the necessary operations, like evaluating branching paths.

Utilization: Considering database engineering as the application domain of ER, an example of a domain-specific purpose is the generation of SQL statements out of ER models. The ADOxx-specific snippet below implements a mechanism for the transformation of content from models to SQL statements. Figure 16.10 shows an available code editor[5] for writing AdoScript, with one part of it being:

```
PROCEDURE global SQL_CREATE_TABLE_FOR_OBJECT
    integer:id_object str_return:reference
{
    CC "Core" GET_MODEL_ID objid:(id_object)
    SETL id_model:(modelid)
    CC "Core" GET_OBJ_NAME objid:(id_object)
    SETL str_objname:(objname)
    CC "Core" GET_CLASS_ID objid:(id_object)
```

5 The Visual Studio Code extension ADOxx AdoScript is available from https://marketplace.visualstudio.com/items?itemName=ADOxxorg.adoxx-adoscript (accessed 06.03.2023).



```
    SET global_str_sqlcreatetable:(global_str_sqlcreatetable + str_tmp +
    global_str_endofcommand + "\n\n")
  }

  SETL str_return:(global_str_sqlcreatetable + "\n" + global_str_sqlaltertable)
}



#---------------------------------------------------------------------------------
PROCEDURE global SQL_CREATE_TABLE_FOR_OBJECT integer:id_object
str_return:reference
#---------------------------------------------------------------------------------
{
  CC "Core" GET_MODEL_ID objid:(id_object)
  SETL id_model:(modelid)
  CC "Core" GET_OBJ_NAME objid:(id_object)
  SETL str_objname:(objname)
  CC "Core" GET_CLASS_ID objid:(id_object)
  CC "Core" GET_CLASS_NAME classid:(classid)
  SETL str_classname:(classname)
  SETL str_tabname:(escapeStringForSQL(str_objname))
  SETL str_return:("-- Table for " + str_classname + ": " + str_objname +
```

Figure 16.10: Procedure definition with AdoScript in Visual Studio Code

```
    CC "Core" GET_CLASS_NAME classid:(classid)
    SETL str_classname:(classname)
    SETL str_tabname:(escapeStringForSQL(str_objname))
    SETL str_return:("-- Table for " + str_classname +
        ": " + str_objname + "\nCREATE TABLE " + str_tabname + " (\n")
    #... Much more code, too much to show here
}
```

The code defines a reusable global procedure to generate the SQL code for an entity or a relationship. It has an input parameter specifying the object ID and an output parameter for the generated SQL code. Two types of commands can also be seen here: 1) 'CC' call commands that access content from the model through 'Core' which is part of the API provided by the ADOxx platform and 2) 'SETL' commands which set local variable values. Some other code of the procedure that is not shown here includes calls to other procedures, use of conditional statements, loop statements and expressions to be evaluated. Similar AdoScript code is also used to support the modeling procedure by defining the interaction of the user with the mechanism. Calls to the ADOxx API are performed to ask the user for an input model and a file location to store the result. Furthermore a trigger for the mechanism is created in form of a menu item to be able to start it. More example cases can be found in OMiLAB-Community (2022) and implementations of various modeling tools from the OMiLAB community are available on OMiLAB.[6]

6 See also `https://www.omilab.org/activities/living-paper/develop+realize-within-adoxx/`

16.5 Conclusion

In this chapter, we introduce AdoScript, a DSL for realizing modeling methods and building modeling tools, i. e., fulfilling a set of requirements defined for an MM-DSL. Through its meta-concepts, it aims to be a platform-independent language; although the community has predominantly used its specific implementation available in ADOxx, prior work (Visic 2016; Visic et al. 2015) defined EBNF syntax rules and demonstrated patterns of compilation for translating the hereby discussed AdoScript features for other metamodeling platforms. By not being limited to language specification, AdoScript emphasises the pragmatics of modeling in alignment with syntax and semantics, possibly even determining requirements that must be satisfied on syntactic or semantic level. Such dependencies can be mapped in the modeling tool CoChaCo (Karagiannis et al. 2019).

Therefore, besides MetaMorph, AdoScript is a formalism for modeling languages, and FDMM, a formal specification language for modeling languages, the third pillar in a language trilogy of MM-DSLs serving different layers of formalism for a modeling method (definition, specification, implementation). We specify four MM-DSL meta-concepts that are fundamental for AdoScript, i. e., Instantiation, Visualization, Customization, and Utilization. These concepts are outlined in detail and their concrete application in a metamodeling platform is demonstrated using ADOxx. The usage of the AdoScript meta-concepts is demonstrated on the Bee-Up tool, a modeling tool comprising the well-known standardized modeling languages BPMN, ER, EPC, UML, and Petri Nets. The advancement of all three pillars of the language trilogy for MM-DSLs is currently under progression (Döller et al. 2023).

For scientific conceptual modeling and in consequence for the realization and exploitation of these results in the area of modeling method engineering, a standard DSL would be beneficial. AdoScript, the contribution of this chapter, can be used as a starting point in this direction. Another suggestion is to approach this endeavour in an open organization such as OMiLAB. In OMiLAB, it has been experimentally shown by using the ADOxx platform that it is much easier to exchange models if the tools used to create them are employing the same underlying meta^2-model. It has also been shown that by using a DSL one does not need to depend on a single metamodeling platform because DSL code can be compiled to any platform for which a ‘transformation component’ exists.

From a management point of view, it is very important to establish common ground where models can be published, accessed, modified, and redistributed. Models are certainly not pictures. Therefore, the first requirement would be to have a common format for model sharing. Secondly, tools used to create models should allow the export to the well-known OMG standard for exchanging metadata information (XMI). It is used almost in every Eclipse-based modeling tool. XMI may be good enough for UML-based model exchange, but it is not sufficient for models that have more complex graphical representations. Some of the modeling tools do possess the means to exchange both kinds of information – metadata and graphical – but typically only between themselves.

The standardization of a DSL also includes the open elaboration of MM-DSL requirements. One of them is the possibility of opening up the engineering of modeling methods to arbitrary requirements coming from communities of modellers having domain-specific purposes and common goals. Due to the knowledge-intense task of a method engineer it is beneficial to make use of collective intelligence, e. g., via social media and use the knowledge of a crowd of domain experts.

In conclusion, we want to repeat the goal we pursue with the intention of stimulating not only the use, but also the engineering of modeling methods with the help of artifacts such as the hereby presented contribution: *We use abstraction to reduce complexity in a domain for a specific purpose.*

References

Buchmann, R. A. (2022). 'The Purpose-Specificity Framework for Domain-Specific Conceptual Modeling'. In: *Domain-Specific Conceptual Modeling*, pp. 67–92. DOI: 10.1007/978-3-030-93547-4_4.

Buchmann, R. A., Ghiran, A.-M., Döller, V. and Karagiannis, D. (2019). 'Conceptual Modeling Education as a "Design Problem"'. In: *Complex Systems Informatics and Modeling Quarterly* 21, pp. 21–33. DOI: 10.7250/csimq.2019-21.02.

Choe, Y. and Lee, M. (July 2016). 'Algebraic Method to Model Secure IoT'. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, pp. 335–355. DOI: 10.1007/978-3-319-39417-6_15.

Döller, V. (2022). 'Formalizing the four-layer metamodeling stack with MetaMorph: potential and benefits'. In: *Software and Systems Modeling* 21.4, pp. 1411–1435. DOI: 10.1007/s10270-022-00986-2.

Döller, V. and Götz, S. (2022). 'Tree Diagrams and Unit Squares 4.0 : Digitizing Stochastic Classes with the Didactic Modeling Tool ProVis'. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools.* Ed. by D. Karagiannis, M. Lee, K. Hinkelmann and W. Utz. Cham: Springer, pp. 481–501. DOI: 10.1007/978-3-030-93547-4_21.

Döller, V., Karagiannis, D. and Utz, W. (2023). 'MetaMorph: formalization of domain-specific conceptual modeling methods—an evaluative case study, juxtaposition and empirical assessment'. In: *Software and Systems Modeling* 22.1, pp. 75–110. DOI: 10.1007/s10270-022-01047-4.

Fill, H.-G., Redmond, T. and Karagiannis, D. (2012). 'FDMM: A formalism for describing ADOxx meta models and models'. In: *ICEIS 2012 - Proceedings of the 14th International Conference on Enterprise Information Systems.* Vol. 3, pp. 133–144. DOI: 10.5220/0003971201330144.

Fowler, M. (2011). *Domain-specific languages.* The Addison-Wesley signature series. Upper Saddle River, NJ: Addison-Wesley.

Frank, U. (1998). 'The MEMO META-METAMODEL'. In: *Arbeitsberichte des Instituts für Wirtschaftsinformatk Nr.9, Koblenz.*

Frank, U. (2011). *Multi-Perspective Enterprise Modelling: Background and Terminological Foundation.* ICB Research Reports 46. Universität Duisburg-Essen, Institut für Informatik und Wirtschaftsinformatik.

Frank, U. (2013). 'Domain-specific modeling languages: Requirements analysis and design guidelines'. In: *Domain Engineering: Product Lines, Languages, and Conceptual Models.* Ed. by I. Reinhartz-Berger, A. Sturm, T. Clark, S. Cohen and J. Bettin. Springer Berlin Heidelberg, pp. 133–157. ISBN: 9783642366543. DOI: 10.1007/978-3-642-36654-3_6.

Frank, U. (2014). 'Multilevel modeling: Toward a new paradigm of conceptual modeling and information systems design'. In: *Business and Information Systems Engineering* 6, pp. 319–337. DOI: 10.1007/s12599-014-0350-4.

Karagiannis, D. (2018). ‘Conceptual modelling methods: The AMME agile engineering approach’. In: *Informatics in Economy. IE 2016*. Vol. 273. Lecture Notes in Business Information Processing. Springer, pp. 3–19. DOI: 10.1007/978-3-319-73459-0_1.

Karagiannis, D. and Buchmann, R. (2018). ‘A Proposal for Deploying Hybrid Knowledge Bases: the ADOxx-to-GraphDB Interoperability Case’. In: *HICSS 2018 - Proceedings of the 51st Hawaii International Conference on System Sciences*. AIS eLibrary.

Karagiannis, D., Buchmann, R. A., Burzynski, P., Reimer, U. and Walch, M. (2016a). ‘Fundamental conceptual modeling languages in OMiLAB’. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, pp. 3–30. DOI: 10.1007/978-3-319-39417-6_1.

Karagiannis, D., Burzynski, P., Utz, W. and Buchmann, R. A. (2019). ‘A Metamodeling Approach to Support the Engineering of Modeling Method Requirements’. In: *27th IEEE International Requirements Engineering Conference*, pp. 199–210. DOI: 10.1109/RE.2019.00030.

Karagiannis, D. and Kühn, H. (2002). ‘Metamodelling Platforms’. In: *Proceedings of the Third International Conference on E-Commerce and Web Technologies - DEXA 2002*. Ed. by K. Bauknecht, A. Min Tjoa and G. Quirchmayer. Springer. Berlin, Heidelberg, p. 182. DOI: 10.1007/3-540-45705-4_19.

Karagiannis, D., Mayr, H. C. and Mylopoulos, J., eds. (2016b). *Domain-specific conceptual modeling: Concepts, methods and tools*. Springer. DOI: 10.1007/978-3-319-39417-6.

Karagiannis Dimitris, Lee Moonkun, Hinkelmann Knut and Utz Wilfrid, eds. (2022). *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools*. Springer International Publishing, p. 656. DOI: 10.1007/978-3-030-93547-4.

Kelly, S. and Tolvanen, J. P. (2007). ‘Domain-Specific Modeling: Enabling Full Code Generation’. In: *Domain-Specific Modeling: Enabling Full Code Generation*, pp. 1–427.

Mayr, H. C., Al Machot, F., Michael, J., Morak, G., Ranasinghe, S., Shekhovtsov, V. and Steinberger, C. (2016). ‘HCM-L: Domain-specific modeling for active and assisted living’. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, pp. 527–552. DOI: 10.1007/978-3-319-39417-6_24/COVER.

Michael, J. and Mayr, H. C. (2016). ‘Creating a Domain Specific Modelling Method for Ambient Assistance’. In: *15th International Conference on Advances in ICT for Emerging Regions, ICTer 2015 - Conference Proceedings*, pp. 119–124. DOI: 10.1109/ICTER.2015.7377676.

Moody, D. (2009). ‘The physics of notations: Toward a scientific basis for constructing visual notations in software engineering’. In: *IEEE Transactions on Software Engineering* 35.6, pp. 756–779. DOI: 10.1109/TSE.2009.67.

Morita, T. and Yamaguchi, T. (2022). ‘Generating ROS Codes from User-Level Workflow in PRINTEPS’. In: *Domain-Specific Conceptual Modeling*, pp. 435–455. DOI: 10.1007/978-3-030-93547-4_19.

Muck, C. and Palkovits-Rauter, S. (2022). ‘Conceptualizing Design Thinking Artefacts: The Scene2Model Storyboard Approach’. In: *Domain-Specific Conceptual Modeling*, pp. 567–587. DOI: 10.1007/978-3-030-93547-4_25.

OMiLAB-Community (2022). ‘Development of Conceptual Models and Realization of Modelling Tools Within the ADOxx Meta-Modelling Environment: A Living Paper’. In: *Domain-Specific Conceptual Modeling*, pp. 23–40. DOI: 10.1007/978-3-030-93547-4_2.

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A. and Sinz, E. J. (2011). ‘Memorandum on design-oriented information systems research’. In: *European Journal of Information Systems* 20.1, pp. 7–10.

Rosenthal, K., Ternes, B. and Strecker, S. (2019). ‘Learning Conceptual Modeling: Structuring Overview, Research Themes and Paths for Future Research’. In: *ECIS 2019 - Proceedings of the 27th European Conference on Information Systems.* AIS eLibrary.

Strecker, S., Baumöl, U., Karagiannis, D., Koschmider, A., Snoeck, M. and Zarnekow, R. (2019). ‘Five Inspiring Course (Re-) Designs’. In: *Business and Information Systems Engineering* 61.2, pp. 241–252.

Ternes, B., Rosenthal, K. and Strecker, S. (2021). ‘Automated Assistance for Data Modelers: a Heuristics-based Natural Language Processing Approach’. In: *ECIS 2021 - Proceedings of the 29th European Conference on Information Systems.* AIS eLibrary.

Thalheim, B. (2012). ‘Syntax, Semantics and Pragmatics of Conceptual Modelling’. In: *NLDB 2012 - Proceedings of the 17th International Conference on Applications of Natural Language to Information Systems.* Vol. 7337. Lecture Notes in Computer Science. Springer, pp. 1–10. doi: 10.1007/978-3-642-31178-9_1.

Visic, N. (2016). ‘Language-Oriented Modeling Method Engineering’. Doctoral dissertation. University of Vienna.

Visic, N., Fill, H.-G., Buchmann, R. A. and Karagiannis, D. (2015). ‘A domain-specific language for modeling method definition: from requirements to grammar’. In: *RCIS 2015 - Proceedings of the 9th IEEE International Conference on Research Challenges in Information Science.* IEEE CS, pp. 286–297. doi: 10.1109/RCIS.2015.7128889.

Chapter 17

Model-Driven Development: From Strategy to Code

Oscar Pastor, Rene Noel, Rosa Velasquez and José Ignacio Panach

Model-driven development puts software abstractions at the centre of the software development process. Through model-to-model transformations, business knowledge abstractions help developers to align software design with business requirements. However, the constantly changing environment and the social, cross-disciplinary nature of software development teams sets two main challenges to model-driven methods. On the one hand, modern organisations have an accelerated approach to defining, deploying, and measuring the success of business strategy plans, making them a central part of the software development process. On the other hand, business users need a shared comprehension of business processes, which might be enabled only by understandable business process models. This chapter presents a model-driven method that starts by representing the business strategy that triggers the software development initiative, then provides an assisted business process modelling method to improve models' understandability, and then connects with a sound model-driven method that generates the software code. The method aims to help design understandable business processes and software components aligned to business strategy and improve the traceability of the whole development process from strategy to code.

17.1 Introduction

Model-Driven Development (MDD) is the systematic use of software abstractions (models) as primary artefacts during a Software Engineering (SE) process (Mellor et al. 2003). Following the Model Driven Architecture (MDA) (The Object Management Group 2014) approach, software models can be connected with different system and business abstraction levels. The abstraction levels are connected through model-to-model transformations from business information at the Computation Independent Modelling (CIM) level, which can then be transformed into Platform Independent Models (PIM) that embody the design of the system and then into Platform Specific Models (PSM) with technical details to support the automatic generation of the system code.

One of the key advantages of the MDA-based methods is to allow business stakeholders from different domains to participate in the development process (The Object Management Group 2014). However, in model-driven processes, all the stakeholders must understand the models. The understandability of conceptual models is an open research problem (Houy et al. 2012) since it depends on several factors both from the audience of the models (e. g., domain knowledge, modelling language knowledge) and from the models themselves (e. g., complexity, layout) (Dikici et al. 2018).

Besides including stakeholders from different domains, MDA-based initiatives also help to include information from different organisational levels and preserve their alignment. MDA-based initiatives have pioneered the connection of business goals with business processes (Barat et al. 2017; Guizzardi and Reis 2015; Amyot et al. 2022), that can then be transformed into the conceptual schema of the information system (España 2011) and then into the code of the information system (Pastor and Molina 2007). However, recent approaches to software development (Evans and Evans 2004; Zimmermann 2017) and for scaling agile delivery of software products in digital enterprises (Mendling et al. 2019; Forsgren et al. 2018), put specific attention on knowledge about business strategy and organisational structure knowledge. While business strategy leads the design of business and software mechanisms to continuously assess the performance of strategic initiatives, the organisational structure serves to design loosely-coupled software components (Conway 1968; Thoughtworks 2016). However, business strategy and organisational structure information has not been completely included in modelling frameworks (Kitsios and Kamariotou 2019) nor in MDA-based development methods.

This article presents a model-driven software production method that includes the business strategy and organisational structure models and an assisted business process modelling method for understandability improvement. Using the MDA-based approach, we provide semantically consistent links between different abstraction levels through model-to-model transformations to ensure the alignment of software products from strategy to code.

The remeinader of the article continues with Section 17.2 which presents the related work. In Section 17.3, we describe the model-driven method, and in Section 17.4, we discuss the benefits and limitations of the proposal. Finally, we summarise the conclusions in Section 17.5.

17.2 Related work

In this section, we review some initiatives that exploit the model-driven approach to connect business domain knowledge with the information system model and thus with the code of the information system. Similarly to our proposal, these initiatives use conceptual modelling techniques, such as different modelling levels and model-to-model transformations. These initiatives prove the suitability of our approach. Yet, they do not get to the business strategy level. At the same time, we review the current studies on business process model understandability, as a key enabler of the participation of business domain people in the development process.

17.2.1 From Requirements to Code: Model-Driven Initiatives

The model-driven architecture (MDA) initiatives have focused on transforming models at different abstraction levels, helping the requirements engineering activities, and some of them support code generation. Sebastián et al. (2020) conducted a systematic mapping study to identify the principal challenges and gaps in generating code from conceptual models in

MDA-based methods. Fifteen MDA-based initiatives were identified, mostly using UML as modelling framework, although only a few of them considered CIM models.

CIM models are more used regarding requirements engineering, were the strategic alignment term has been coined to connect the system stakeholders' goals with business process models. Some initiatives focus on using goal-oriented modelling languages such as KAOS (Van Lamsweerde 2001) and i* (Yu et al. 2011). These languages have been integrated into business process models, aligning with information systems. For example, Ruiz et al. propose GOBIS (Ruiz et al. 2015) to integrate i* and business process model using Communication Analysis method (España et al. 2009). In the same line, Barat et al. proposed a method centred on complex decision-making (Barat et al. 2017). This approach allows for identifying and modelling important information related to the goals of the process. Other strategic alignment initiatives consider goal modelling with TROPOS (Guizzardi and Reis 2015) and GRL (Amyot et al. 2022), MAPs (Kraiem et al. 2014), mostly to analysis and check how business processes support and fulfil the system stakeholder's goals.

However, business strategy goals lie in a higher-abstraction level than information system stakeholders, that have been mostly addressed by Enterprise Architecture modelling frameworks, such as Business Motivation Model (The Object Management Group 2015) and ArchiMate (The Open Group 2023). These frameworks provide business strategy concepts and support connecting them with IT infrastructure components, yet not with detailed software design elements.

In summary, using existing sound conceptual modelling techniques and frameworks in combination presents an open challenge and also an opportunity to enable organisational agility with model-driven development methods.

17.2.2 Prior research on model understandability

The model understandability is a relevant criterion for measuring the quality of conceptual models. Many researchers have proposed multiple factors that influence this criterion, being of crucial importance model factors and personal factors (Dikici et al. 2018).

The impact of model factors has been widely researched in many studies. Regarding model factors, metrics of models such as complexity, number of tasks, and size, among other quantifiable measures (Mendling 2008), have been considered influential factors that impact the models understandability (Störrle 2014; Reijers and Mendling 2011). For example, Cardoso et al. confirm an important correlation between complexity and understandability (Rolón et al. 2009). Other researchers focus on non-quantifiable model characteristics such as modularization (Reijers and Mendling 2008), labelling style of the model elements (Mendling et al. 2010), or visual layout (Störrle 2014). Even though the results revealed that each factor impacts the understandability, they do not indicate the level of impact for each one. So, we can not know which factors affects more clearly the model understandability.

Personal factors have been less analysed than model factors. These factors focus on the characteristics of the user that interpret the model, such as modelling expertise, theoretical knowledge, and domain familiarity. In 2007, Mendling et al. (2007) operationalised the theoretical knowledge and practical experience in BPMs. Using demographics surveys, Recker (2010) capture user characteristics. Also, Mendling et al. (2019) operationalised a set of measurements related to the user's expertise as training, familiarity with BPMs, confidence in understanding BPMs, and competence using BPMs. The principal conclusions of this research lead to: (1) replicate prior studies incorporating the user characteristics explicitly; and (2) operationalise the user characteristics more rigorously. Another research issue is how to measure model understandability; researchers have used indicators to operationalise and quantify process model understandability. In Mendling et al. (2019), the authors defined performance as an indicator to measure task effectiveness, and completion time as an indicator to measure task efficiency. Effectiveness indicator was based on ad-hoc questions for each model, so the user can analyse and answer according to the process's flow. In recent years, several initiatives, both enterprises and industries that regard model quality, have opted for introducing the use of automation tools based on machine learning (ML) to predict software quality. There initiatives focused on two directions: Repair model (Barriga et al. 2019; Pinna Puissant et al. 2015) and Recommender systems (Fellmann et al. 2017; Koschmider and Oberweis 2010). However, both directions focus on already designed models, but they do not prevent issues during the modeling activity.

In summary, there is research that investigates factors that influence the understandability in BPMs, however, this research focuses on a limited set of factors leaving aside others. Hence, these proposals have not established an integral combination of factors that may affect the understandability. On the other hand, the initiatives that use ML to improve the quality of the model do not improve the understandability during the modelling activity.

17.3 From Strategy to Code with Understandable Business Models

17.3.1 Method Overview

Motivated by the approach of software-centric organisations for aligning their strategy and systems and achieving organisational agility, we propose a model-driven method for developing software that connects requirements from the business strategy with the software code through a series of semi-automatic model-to-model transformations. As depicted in Figure 17.1, our proposal combines three modelling levels: the organisational, business process, and the information system levels. The proposed transformations automatically connect critical business information for designing the system, providing traceability from strategy to code.

First, a business strategy model at the organisational level represents the business scenario that drives the change in the current organisation processes and systems. The business strategy model is focused on the strategic plan (goals, strategies, tactics and objectives) and the organisational structure (organisation units and roles) needed to deploy the strategy. A model-to-model transformation conveys the key concepts needed to design strategically-aligned processes: the organisation units that must deploy the strategy, their dependencies, and the key objectives to assess their performance in implementing the strategy.

The concepts generated by the transformation scaffold the re-design of independent, modular business processes, from which we focus on modelling the communicative interactions between the system actors, including the information exchanged in each interaction. In this level, users of the method and business stakeholders must commit to the requirements expressed in the business process model, so ensuring the understandability of the models is crucial for the subsequent generation of the information system model. The user of the method is a business analyst, whom we will call 'the analyst' from now on. The business process model is then transformed into the information system model. By analysing the structure of the exchanged information and the sequence of interactions, the analysts can

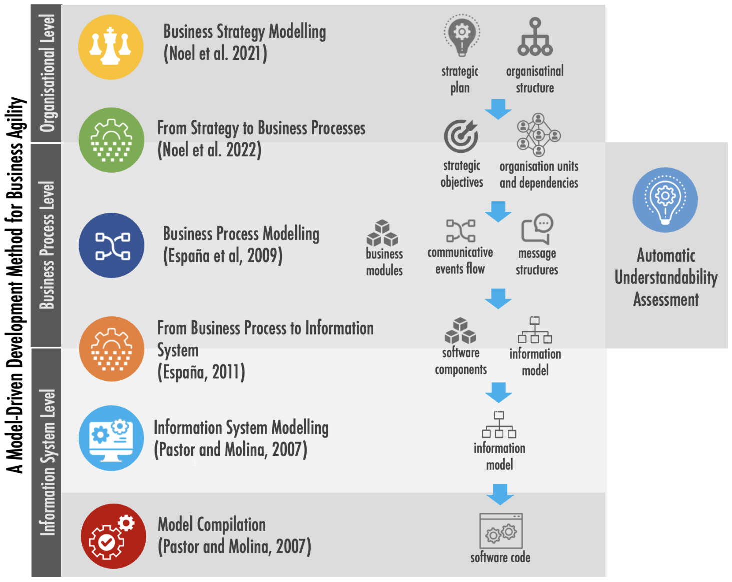

A Model-Driven Development Method for Business Agility
Organisational Level
Business Process Level
Information System Level
Business Strategy Modelling (Noel et al. 2021)
strategic plan
organisatinal structure
From Strategy to Business Processes (Noel et al. 2022)
strategic objectives
organisation units and dependencies
Business Process Modelling (España et al, 2009)
business modules
communicative events flow
message structures
From Business Process to Information System (España, 2011)
software components
information model
Information System Modelling (Pastor and Molina, 2007)
information model
Model Compilation (Pastor and Molina, 2007)
software code
Automatic Understandability Assessment

Figure 17.1: A model-driven approach from strategy to code approach

generate most of the information system model. Moreover, the modular design of the business processes is traduced into a modular design of information system models, and thus to software components. Finally, the conceptual schema of the information system is compiled to produce the working code of the system.

In the following subsections, we will present each of the stages of the above approach through a working example. We introduce Lodging Co., a fictional on-line travel agency for short-term dwelling reservations. The customers can search for properties according to a set of criteria, and after finding one of their preference, can book it for a given period. Customers must pay when confirming the reservation, and they could also pay for and additional breakfast service. The data tracked from the customer's activity have shown that recurring customers are very prone to purchase the additional breakfast service. Lodging Co. executives have envisioned the opportunity to offer other additional services, which can be a new business model. To enable this new business model, senior executives must design and deploy a new strategy.

17.3.2 Modelling Business Strategy

The first step of our proposal considers to model business strategy in order to capture business logic that is relevant for designing the software system. Though many modelling approaches have considered business strategy concepts (The Open Group 2023; The Object Management Group 2015), they lack of critical information for applying an agile approach for developing systems: the organisational structure. For this, in previous work we designed LiteStrat (Noel et al. 2021), a business strategy modelling method that provides a language with business strategy and organisational structure concepts, and a modelling procedure that ensure an outside-in approach for strategy, similarly to the SCO's approach presented

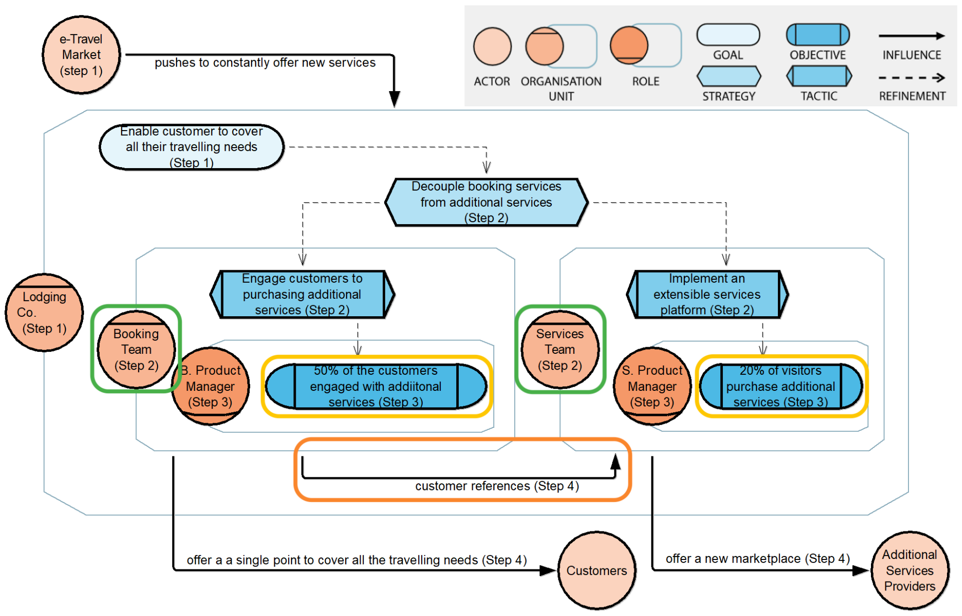

ACTOR
ORGANISATION UNIT
ROLE
GOAL
STRATEGY
OBJECTIVE
TACTIC
INFLUENCE
REFINEMENT
e-Travel Market (step 1)
pushes to constantly offer new services
Enable customer to cover all their travelling needs (Step 1)
Decouple booking services from additional services (Step 2)
Lodging Co. (Step 1)
Engage customers to purchasing additional services (Step 2)
Implement an extensible services platform (Step 2)
Booking Team (Step 2)
B. Product Manager (Step 3)
50% of the customers engaged with addiitonal services (Step 3)
Services Team (Step 2)
S. Product Manager (Step 3)
20% of visitors purchase additional services (Step 3)
customer references (Step 4)
offer a a single point to cover all the travelling needs (Step 4)
Customers
offer a new marketplace (Step 4)
Additional Services Providers

Figure 17.2: A LiteStrat model for the strategic scenario.

in Section 17.1. To show the application of LiteStrat, consider the extension of the working example written below and depicted in the LiteStrat diagram in Figure 17.2.

Lodging Co. executives have set the goal of enabling their customers to cover all the additional services needed when renting a short-term dwelling. To do this, the primary strategy is to decouple their current booking business from the additional services business. This will permit growth from the scarce current offering of additional breakfast service to new services such as car rentals, restaurant reservations, sightseeing tours, etc. To deploy the strategy, the executives decided to create a new agile cell named Services Team apart from the existing Booking Team. Under this new structure, the Booking Team must refer the customers to the additional services when the booking is confirmed; since this is critical for the effective implementation of the strategy, the booking services must engage customers into purchasing additional services, so the Booking Product Manager must fulfil the objective of successfully referring at least a 50% of its customers to the additional services area. The Services Team must design an extensible platform for continuously including new additional services, and the success of the initiative will be achieved if 20 per cent of the engaged customers purchase an additional service. With the new strategy, Lodging Co. aims to influence their customers by offering a single point for covering all they travelling needs, and also to influence the additional services providers by offering a new marketplace.

Following the LiteStrat modelling procedure, the first step is to represent the external influences that drive the change of strategy and the main goal (Step 1). Next, the courses of action needed to implement the strategy are defined in terms of strategies and tactics; the latter are assigned to the organisational units that must implement them (Step 2). In the third step, the measurable, specific objectives are modelled and assigned to roles in the organisational units responsible for achieving or tracking them (Step 3). Finally, the effects of the strategy are modelled in terms of influences between the organisation units and external actors (Step 4).

17.3.3 From Strategy to Business Process Models

The information in the business strategy model is then used for generating an initial version of the business process model through the Stra2Bis method (Noel et al. 2022). Stra2Bis is a method for performing a model-to-model transformation from the source business strategy model to a target business process model. The transformation is designed to preserve critical information for keeping business process aligned with the organisation structure and objectives. Stra2Bis proposes three guidelines in order to design business processes that are independent between the different organisation units, with well defined interfaces, and that consider requirements for measuring the strategic objectives. The resulting initial business process model is presented in Figure 17.3 using the Communication Analysis notation. The three transformation guidelines are summarised below.

- Guideline 1 - Organisation Units' Independence: Design a single business process for each organisation unit. The transformation's source and target elements are marked in green in Figures 17.2 and 17.3 respectively.
- Guideline 2 - Managed Strategic Dependencies: Design the interactions between business processes to manage the organisation units' strategic dependencies. The transformation's source and target elements are marked in orange in Figures 17.2 and 17.3 respectively.
- Guideline 3 - Strategic Objectives Measurement: Design business process elements to collect data to measure strategic objectives. The transformation's source and target elements are marked in yellow in Figures 17.2 and 17.3 respectively.

By applying the above guidelines, the analysts are leaded to design separated business processes (Booking Process and Service Process) for the different organisation units (Booking

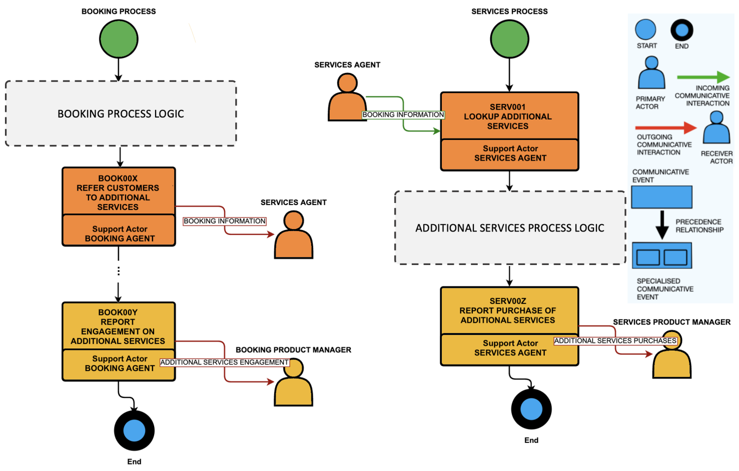

BOOKING PROCESS
BOOKING PROCESS LOGIC
BOOK00X
REFER CUSTOMERS TO ADDITIONAL SERVICES
Support Actor BOOKING AGENT
BOOKING INFORMATION
SERVICES AGENT
BOOK00Y
REPORT ENGAGEMENT ON ADDITIONAL SERVICES
Support Actor BOOKING AGENT
ADDITIONAL SERVICES ENGAGEMENT
BOOKING PRODUCT MANAGER
End
SERVICES PROCESS
SERVICES AGENT
BOOKING INFORMATION
SERV001
LOOKUP ADDITIONAL SERVICES
Support Actor SERVICES AGENT
ADDITIONAL SERVICES PROCESS LOGIC
SERV00Z
REPORT PURCHASE OF ADDITIONAL SERVICES
Support Actor SERVICES AGENT
ADDITIONAL SERVICES PURCHASES
SERVICES PRODUCT MANAGER
End
START
END
PRIMARY ACTOR
INCOMING COMMUNICATIVE INTERACTION
OUTGOING COMMUNICATIVE INTERACTION
RECEIVER ACTOR
COMMUNICATIVE EVENT
PRECEDENCE RELATIONSHIP
SPECIALISED COMMUNICATIVE EVENT

Figure 17.3: Scaffold of the business process model for the strategic scenario.

Team and the Services Team) to foster their independence. However, the integration between their processes must be clearly defined in the process model; the guidelines propose to add events in the source and the target process ('Refer Customers to Additional Services' and 'Lookup Additional Services') to support this. The Communication Analysis method helps the detailed specification of this interaction through the definition of a Message Structure (España et al. 2011). For instance, the analysts can specify the 'Booking Information' message delivered from the Booking to the Service processes by defining the message's data structure and pointing to exiting information assets. Finally, two separate events are generated to ensure the collecting and delivering information to measure the status of the strategic objectives defined in the strategy, in particular, 'Report Engaging on Additional Services' (BOOK00Y) and 'Report Purchase of Additional Services' (SEV00Z). The analyst must complete the rest of the two business processes according to the business logic.

17.3.4 Understandable Business Process Models

Generally speaking, an important goal is to ensure the models quality. One quality criterion is understandability, which can be defined as how easy it is to understand a model by readers (Reijers and Mendling 2011). As indicated in Section 17.2.2, it is essential to measure many factors that influence the understandability levels of a model. However, exploring all of them using traditional research methods is very time-consuming. Hence, manual analysing can no longer provide sound results. In Velasquez (2021), we presented a work plan to achieve the automatic understandability measurement using ML techniques to predict the level of understandability during the modelling activity. The modelling assistant (Figure 17.4)

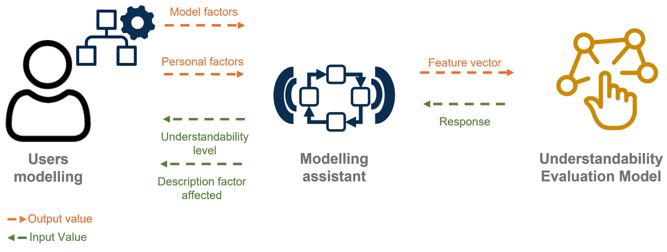
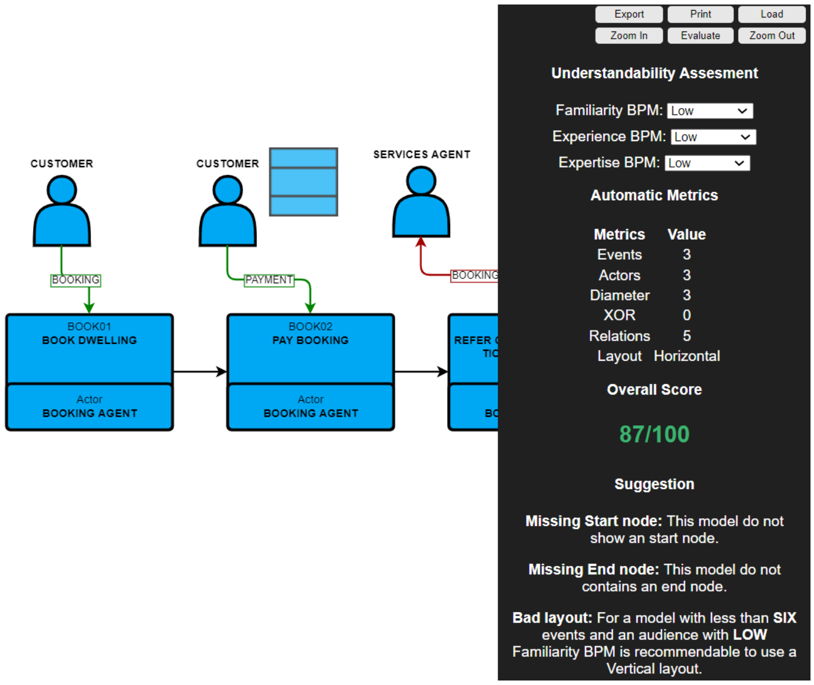


Figure 17.4: Approach to Modelling Assistant

provides early feedback during the modelling process through an automatic assessment of the models jointly based on the characteristics of the model's target audience. The assisted modelling process can be divided into three steps, detailed below:

- Step 1 - Personal factors: Before starting the modelling, the modeller user sets the target audience's characteristics. These characteristics are the audience's modelling expertise, the audience's knowledge of BPM, and the audience's familiarity with the business process model, as shown in Figure 17.4.
- Step 2 – Model factors: During the modelling process, the system will calculate the current model's metrics, such as the number of nodes, actors, events, relations, and diameter. This process is automatic and triggered each time the model changes. As shown in Figure 17.4, the model and personal factors are sent to the modelling assistant.



Understandability Assesment

Familiarity BPM: Low

Experience BPM: Low

Expertise BPM: Low

Automatic Metrics

Metrics	**Value**
Events	3
Actors	3
Diameter	3
XOR	0
Relations	5
Layout	Horizontal

Overall Score

87/100

Suggestion

Missing Start node: This model do not show an start node.

Missing End node: This model do not contains an end node.

Bad layout: For a model with less than **SIX** events and an audience with **LOW** Familiarity BPM is recommendable to use a Vertical layout.

Figure 17.5: Understandability Modelling Assistant

- Step 3 - Understandability Evaluation: In order to evaluate the understandability level continuously, the modelling assistant converts the personal and model factors information into a feature vector that is sent to the Understandability Evaluation Model (UEM). UEM has been previously trained using machine learning techniques to analyse the relationship among factors and to predict the understandability level. Hence, the UEM is in charge of responding to the modelling assistant in terms of the model's understandability level and the weight of the factors. Finally, the Modelling Assistant decodes the response and sends the user the understandability level and recommendation based on the most relevant factors. These scenario are depicted in red in Figure 17.4.

To show the application of Understandability Modelling Assistant (UMA), we will consider a part of the example detailed in Section 17.3.1. Specifically, we focus on the Booking Process. Following the procedure to evaluate the understandability, the first step is to define the target audience's characteristics in terms of personal factors: Familiarity with BPM, Experience in BPM and, Expertise in BPM (the modeller can modify the audience characteristics during all drawn processes). Second, the modeller starts the modelling process; then, the modelling assistant will automatically give feedback. This feedback consists of showing quantifiable metrics about the model, the overall understandability score, and suggestions for improving this score. The right-bar in Figure 17.5 shows the work of the modelling assistant during the modelling process. The suggestion offered by UMA are related to the factors that are most negatively affected.

17.3.5 From Business Models to Information System

Once the business processes are modelled and the message structures are detailed, the model of the information system can be derived following the transformation guidelines proposed by España (2011). The guidelines produce an information system model using the OO-Method (OOM) notation (Pastor and Molina 2007). The OO-Method is a model-driven development method for conceptual programming systems by designing four models: the Object Model (which is similar to UML's class diagram), the Dynamic Model (similar to UML's state machine diagram), the Functional Model (a specific language to specify business logic), and the Presentation Model (which supports modelling user interface components and their source data objects). OOM is supported by the INTEGRANOVA (Integranova 2020) tool, which generates working code of the information system modelled in several technologies.

Table 17.1: Message Structures for Booking and Payment

Message Structure: 'Booking'

Field	**OP**	**Domain**	**Business Object**
BOOKING=<			
Customer	i		CUSTOMER
Dwelling	i		DWELLING
Start Date	i	date	
End Date	i	date	>

Message Structure: 'Payment'

Field	**OP**	**Domain**	**Business Object**
PAYMENT=<			BOOKING (extension)
payment status	i	boolean	
payment date	d	date	>

To illustrate our proposal and for the sake of simplicity, we will limit the example to a subset of the transformation guidelines and to OOM's object model. Following the example, lets consider the events *BOOK01* and *BOOK02* in Figure 17.5. In Table 17.1, we specified the structure for messages *Booking* and *Payment.* As can be seen, the Booking message inputs the data from the customer, which has been already structured in another message, as the Business Object reference *CUSTOMER* indicates.

Similarly, the Booking message also inputs a reference to the *DWELLING* business object. Two data fields are inputted: booking's start and end date whose domain is date. By applying the transformation guidelines, this message produces the BOOKING class shown in Figure 17.6.A. By applying the transformation guidelines to the event 'B', the Payment message is processed. Since it references and extends the above Booking message, the attributes payment status and payment date are added to the booking class, as well as a method to update these data, as shown in Figure 17.6.B.

The analyst could change whether the Payment messages extends or not the booking; a different approach is shown in Figure 17.6.C, considering that the payment just references (and not extends) the booking message, and thus, a new Payment class is generated. By applying the full set of guidelines, all other events can be specified and thus transformed

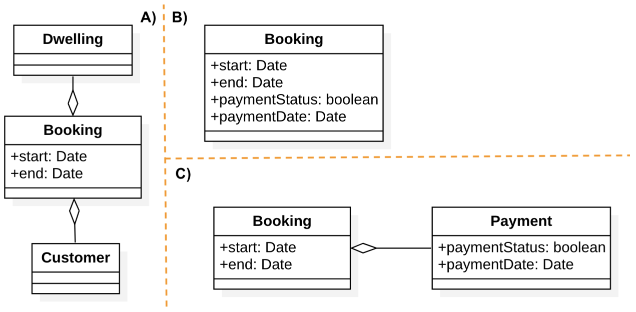


Figure 17.6: Object model produced by the transformation guidelines.

into the information system model. Finally, the information system model is compiled. To do this, platform specific requirements are configured, as shown in Figure 17.7.

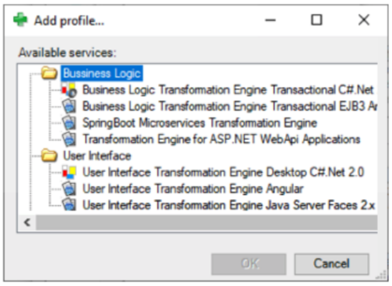

Add profile...
Available services:
Bussiness Logic
Business Logic Transformation Engine Transactional C#.Net
Business Logic Transformation Engine Transactional EJB3 A
SpringBoot Microservices Transformation Engine
Transformation Engine for ASP.NET WebApi Applications
User Interface
User Interface Transformation Engine Desktop C#.Net 2.0
User Interface Transformation Engine Angular
User Interface Transformation Engine Java Server Faces 2.x
OK
Cancel

Figure 17.7: Configuration of platform specific requirements in Integranova.

17.4 Discussion

Besides the well-known claims of model-driven development on improving the productivity and quality of the software process, the proposed method's value lies in its two main contributions: the traceability of the software system to the business strategy level and the software system's design in alignment with the business strategy.

With regard to traceability, it can be noted that the analysts can clearly trace the software components (e. g., classes) to the communicative event (or events) that generated the class, so they can easily analyse the impact of business logic changes over the system. This approach is shared with other initiatives that map business processes to information system models (Guizzardi and Reis 2015; Amyot et al. 2022); however, the fine-grained detail of messages structures supports the analysis of changes even at the level of classes' attributes. Moreover, the analysts can trace the business process to the organisation units that own them and the strategic objectives that the units (and thus the processes) must pursue. Hence, the method also supports analysing whether a change in business logic is strategically aligned. Figure 17.8.A presents an example of the traceability analysis supported by the method.

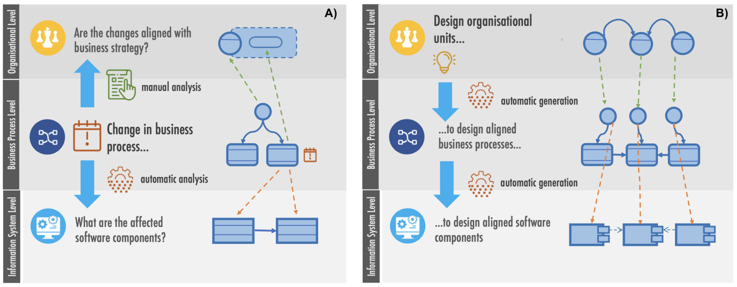

A)
Organisational Level
Business Process Level
Information System Level
Are the changes aligned with business strategy?
manual analysis
Change in business process...
automatic analysis
What are the affected software components?
B)
Organisational Level
Business Process Level
Information System Level
Design organisational units...
automatic generation
...to design aligned business processes...
automatic generation
...to design aligned software components

Figure 17.8: Scheme of Traceability and Alignment

In terms of information system design, the design of a separate business process for each organisation unit, with well-defined business interfaces, leads to the design of separate system components, with well-defined programmatic interfaces. This enables the analysts to have a complete view of the architecture of the system and its relationship with the organisational structure, helping to design the organisation structure according to the desired design of the organisation's systems. This approach is consistent with the recommendations for software-centred organisations which demand high performing software development processes. Figure 17.8.B depicts this design perspective under the method.

The limitations of the method strive on whether the organisation is committed to the digital transformation of their business. Organisations that are starting their digital transformation could exploit the method if continuously assess the performance and design of the development teams assigned to business digital transformation, using this assessments to define and deploy strategy changes. However, traditional organisations that juts leave agility as a software development issue and not involve executive level in the process, probably will not benefit from the contributions of the method. This limitation is shared by frameworks for organisational agility (Highsmith et al. 2019) and for scaling agile software development to the whole organisation (Ambler and Lines 2017; Scaled Agile 2020; Ambler and Lines 2017).

Finally, the method covers specific dimensions of business strategy (organisation structure and objectives), but it could be extended to cover other socio technical level elements which are critical to design software systems, such as privacy and security. Also, the understandability assistance could be extended to cover business process models and business strategy and information system models. This could foster growing the method models' audience that can critically examine them, so the whole organisation can be involved in the development process, from strategy to code.

17.5 Conclusions and Future Work

This article presents a new model-driven method covering from the business strategy level to code generation. Our approach permits the software system's alignment with the business strategy level, including the assurance of understandability of business process models. Using a working example, we have shown the feasibility and value of the method, which covers business strategy modelling, transformation guidelines to preserve strategy information at the business process level, and the integration of business process models with a model-driven systems development method. We also discussed the value and limitations of our proposal.

The future work focuses on extending the method to address other organisational level concerns that can set requirements for the business process level, such as privacy and security. Regarding the understandability of business process models, further work is needed to extend the analysis of factors considering multiple audience characteristics to provide more data variability. Also, we are working on training models to extend the modelling assistant to improve the understandability of business strategy and information system models.

Acknowledgement

This work has been developed with the financial support of the Spanish State Research Agency and the Generalitat Valenciana under projects MICIN/AEI/10.13039/501100011033, GV/2021/072, and INNEST/2021/57, and co-financed with ERDF and the European Union NextGenerationEU/PRTR, the National Agency for Research and Development (ANID)/ Scholarship Program/Doctorado Becas Chile/2020-72210494 and Santiago Grisolía fellowship under the project GRISOLIAP/2020/096 and in part by the project SREC (PID2021-123824OB-I00) funded by the Spanish Ministry of Science and Innovation and co-financed by FEDER.

References

Ambler, S. W. and Lines, M. (2017). *An executive's guide to disciplined Agile: winning the race to business agility*. CreateSpace Independent Publishing Platform.

Amyot, D., Akhigbe, O., Baslyman, M., Ghanavati, S., Ghasemi, M., Hassine, J., Lessard, L., Mussbacher, G., Shen, K. and Yu, E. (2022). 'Combining Goal modelling with Business Process modelling'. In: *Enterprise Modelling and Information Systems Architectures (EMISAJ)–International Journal of Conceptual Modeling* 15.2.

Barat, S., Kulkarni, V., Clark, T. and Barn, B. (2017). 'A Method for Effective Use of Enterprise Modelling Techniques in Complex Dynamic Decision Making'. In: *The Practice of Enterprise Modeling*. Cham: Springer International Publishing, pp. 319–330.

Barriga, A., Rutle, A. and Heldal, R. (2019). 'Personalized and Automatic Model Repairing using Reinforcement Learning'. In: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 175–181.

Conway, M. E. (1968). 'How do committees invent'. In: *Datamation* 14.4, pp. 28–31.

Dikici, A., Turetken, O. and Demirors, O. (2018). 'Factors influencing the understandability of process models: A systematic literature review'. In: *Information and Software Technology*, pp. 112–129.

España, S. (1st Dec. 2011). 'Methodological Integration of Communication Analysis into a Model-Driven Software Development Framework'. PhD thesis. Valencia (Spain): Universitat Politècnica de València.

España, S., González, A. and Pastor, Ó. (2009). 'Communication Analysis: a requirements engineering method for information systems'. In: *International Conference on Advanced Information Systems Engineering*. Springer, pp. 530–545.

España, S., González, A., Pastor, Ó. and Ruiz, M. (2011). '"Message Structures: a modelling technique for information systems analysis and design'. In: *Proceedings of the 14th Workshop on Requirements Engineering (WER 2011).*

Evans, E. and Evans, E. J. (2004). *Domain-driven design: tackling complexity in the heart of software.* Addison-Wesley Professional.

Fellmann, M., Zarvić, N. and Thomas, O. (2017). 'Business Processes Modelling Assistance by Recommender Functionalities: A First Evaluation from Potential Users'. In: *Perspectives in Business Informatics Research.* Springer, pp. 79–92.

Forsgren, N., Humbpotifle, J. and Kim, G. (2018). *Accelerate: The Science of Lean Software and DevOps Building and Scaling High Performing Technology Organizations.* IT Revolution Press.

Guizzardi, R. and Reis, A. N. (2015). 'A method to align goals and business processes'. In: *Proceedings of the 34th International Conference on Conceptual Modeling (ER 2015).* Berlin, Heidelberg: Springer, pp. 79–93.

Highsmith, J., Luu, L. and Robinson, D. (2019). *EDGE: Value-Driven Digital Transformation.* Addison-Wesley Professional.

Houy, C., Fettke, P. and Loos, P. (2012). 'Understanding Understandability of Conceptual Models – What Are We Actually Talking about?' In: *Conceptual Modeling: 31st International Conference ER 2012.* Berlin, Heidelberg: Springer, pp. 64–77.

Integranova (2020). *Integranova Software Solutions.* `http://www.integranova.com/es/`. (Visited on 11/05/2020).

Kitsios, F. and Kamariotou, M. (2019). 'Business strategy modelling based on enterprise architecture: a state of the art review'. In: *Business Process Management Journal* 25.4, pp. 606–624.

Koschmider, A. and Oberweis, A. (2010). 'Designing Business Processes with a Recommendation-Based Editor'. In: *Handbook on Business Process Management 1 – Introduction, Methods, and Information Systems.* Ed. by J. vom Brocke and M. Rosemann. Springer, pp. 299–312.

Kraiem, N., Kaffela, H., Dimassi, J. and Al Khanjari, Z. (2014). 'Mapping from MAP models to BPMN processes'. In: *Journal of Software Engineering* 8.4, pp. 252–264.

Mellor, S. J., Clark, A. N. and Futagami, T. (2003). 'Guest editors' introduction: Model-driven development'. In: *IEEE Software* 20.05, pp. 14–18.

Mendling, J., Reijers, H. and Recker, J. (2010). 'Activity labeling in process modeling: Empirical insights and recommendations'. In: *Information Systems* 35.4. Vocabularies, Ontologies and Rules for Enterprise and Business Process Modeling and Management, pp. 467–482.

Mendling, J. (2008). *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness.* Berlin Heidelberg: Springer-Verlag.

Mendling, J., Recker, J., Reijers, H. A. and Leopold, H. (2019). 'An Empirical Review of the Connection Between Model Viewer Characteristics and the Comprehension of Conceptual Process Models'. In: *Information Systems Frontiers* 21, pp. 1111–1135.

Mendling, J., Reijers, H. A. and Cardoso, J. (2007). 'What Makes Process Models Understandable?' In: *Business Process Management.* Springer Berlin Heidelberg, pp. 48–63.

Noel, R., Panach, I., Ruiz, M. and Pastor, O. (2021). 'The LiteStrat method: towards strategic model-driven development'. In: *Procedings of the 29th International Conference on Information Systems Development (ISD).* Universitat Politècnica de València.

Noel, R., Panach, J. I., Ruiz, M. and Pastor, O. (2022). ‘Stra2bis: A model-driven method for aligning business strategy and business processes’. In: *International Conference on Conceptual Modeling (ER 2022)*. Springer. Cham, pp. 255–270.

Pastor, O. and Molina, J. C. (2007). *Model-driven architecture in practice: a software production environment based on conceptual modeling*. Springer Science & Business Media.

Pinna Puissant, J., Van Der Straeten, R. and Mens, T. (2015). ‘Resolving model inconsistencies using automated regression planning’. In: *Software & Systems Modeling*, pp. 461–481.

Recker, J. (2010). ‘Continued use of process modeling grammars: the impact of individual difference factors’. In: *European Journal of Information Systems* 19.1, pp. 76–92.

Reijers, H. A. and Mendling, J. (2011). ‘A Study Into the Factors That Influence the Understandability of Business Process Models’. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, pp. 449–462.

Reijers, H. and Mendling, J. (2008). ‘Modularity in Process Models: Review and Effects’. In: *Business Process Management*. Springer, Berlin, Heidelberg, pp. 20–35.

Rolón, E., Cardoso, J., García, F., Ruiz, F. and Piattini, M. (2009). ‘Analysis and Validation of Control-Flow Complexity Measures with BPMN Process Models’. In: *Enterprise, Business-Process and Information Systems Modeling*. Berlin, Heidelberg: Springer, pp. 58–70.

Ruiz, M., Costal, D., España, S., Franch, X. and Pastor, Ó. (2015). ‘GoBIS: An integrated framework to analyse the goal and business process perspectives in information systems’. In: *Information Systems* 53, pp. 330–345.

Scaled Agile (2020). *SAFe 5 for Lean Enterprises*. `https://www.scaledagileframework.com/`. (Accessed on 04/10/2021).

Sebastián, G., Gallud, J. A. and Tesoriero, R. (2020). ‘Code generation using model driven architecture: A systematic mapping study’. In: *Journal of Computer Languages* 56, p. 100935.

Störrle, H. (2014). ‘On the Impact of Layout Quality to Understanding UML Diagrams: Size Matters’. In: *Model-Driven Engineering Languages and Systems*. Springer, Cham, pp. 518–534.

The Object Management Group (2014). *Model Driven Architecture (MDA)*. `https://www.omg.org/mda/`. (Accessed on 04/14/2021).

The Object Management Group (2015). *Business Motivation Model Specification Version 1.3*. `https://www.omg.org/spec/BMM/About-BMM/`. (Accesed on 02/07/2021).

The Open Group (2023). *The ArchiMate® Enterprise Architecture Modeling Language*. `https://www.opengroup.org/archimate-home`. (Accessed on 04/20/2022).

Thoughtworks (2016). *Inverse Conway Maneuver | Technology Radar | Thoughtworks*. `https://www.thoughtworks.com/es-es/radar/techniques/inverse-conway-maneuver`. (Accessed on 11/09/2021).

Van Lamsweerde, A. (2001). ‘Goal-oriented requirements engineering: a guided tour’. In: *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, pp. 249–262.

Velasquez, R. (2021). ‘A Machine Learning-based Modelling Assistant for Improving Understandability of Business Process Models’. In: *ER 2021: PhD Symposium*, p. 10.

Yu, E., Giorgini, P., Maiden, N., Mylopoulos, J. and Fickas, S. (2011). ‘Modeling Strategic Relationships for Process Reengineering’. In: *Social Modeling for Requirements Engineering*. MIT press, pp. 11–152.

Zimmermann, O. (2017). ‘Microservices tenets’. In: *Computer Science-Research and Development* 32.3, pp. 301–310.

Kapitel 18

Die Entdeckung des Plurals für die Programmierung

Friedrich Steimann

Während in natürlicher Sprache Singular und Plural gleichberechtigt nebeneinander stehen, gibt es in Programmiersprachen praktisch nur den Singular: Wo viele vorkommen, werden diese zu einem (einer Collection oder ähnlichem) verpackt. Zwar reduziert diese Reifizierung von vielen zu einem die Zahl der benötigten Sprachmittel, jedoch macht sie die Programme, die viele handhaben müssen (und das sind praktisch *alle* Programme) unnötig komplex. Mit meinem Beitrag zeige ich, dass eine Einführung des Plurals in die Programmierung nicht nur ökonomisch, sondern auch wohlfundiert sein kann. Zugleich mache ich deutlich, wie sie die Programmierung näher an die Modellierung heranrückt, in der der Plural schon immer zum Selbstverständnis gehört.

18.1 Einleitung

Alles ist ein Objekt. Diesem Leitsatz der objektorientierten Programmierung werden Anhänger der funktionalen Programmierung entgegenhalten, dass in Wahrheit alles eine Funktion ist, aber der entsprechende Paradigmenstreit soll – zumindest in diesem Beitrag – nicht weiter interessieren. Vielmehr will ich mich daran abarbeiten, dass in beiden Sätzen „alles“ über Einzelne rangiert, wie man am Singular der Kopula „ist“ ablesen kann. Obwohl „alles“ eigentlich allumfassend ist, schließt es Viele (Plurale) aus: „Zwei Objekte ist ein Objekt“ wäre grammatikalisch falsch und „zwei Objekte sind ein Objekt“ wäre es inhaltlich.

Was man sprachlich vielleicht als Spitzfindigkeit abtun könnte, ist programmiertechnisch Dogma: Während es in einem Programm zwar in der Regel viele Objekte (und viele Funktionen) gibt, so kann an jeder Stelle des Programms (also dem, was mit „alles“ gemeint ist) immer nur ein Objekt stehen — sollen es viele sein, müssen sie für diesen Zweck durch ein Objekt repräsentiert werden. So gesehen sind eben auch viele Objekte ein Objekt (ein Vieles), so dass der Satz „alles ist ein Objekt“ durch den Einschluss Vieler unter „alles“ nicht ungültig wird. Für Funktionen gilt ähnliches.

Die logische Zusammenfassung von vielen[1] zu Einem hat viele Vorteile. Historisch gesehen hat sie sogar einen Umbruch erheblicher Tragweite bewirkt: Mit Georg Cantors Einführung von Mengen als „jedes Viele, welches sich als Eines denken lässt“ (Cantor 1883, S. 587) ist die Mathematik neu begründet worden. So lassen sich Zahlen genauso als Mengen

1 Ich schreibe hier „viele“ klein, da ich nicht den Plural von einem Vielen (also mehrere Viele) meine, sondern mehrere Einzelne.

codieren wie Tupel, Vektoren, Matrizen usw. Wesentlich ist dabei die strukturbildende Eigenschaft von Mengen: Es gilt nämlich

$$\forall x : x \neq \{x\}.$$

Eine Menge ist etwas anderes als ihre Elemente – sie ist *ein* (mathematisches) Objekt.

Nicht nur die Mathematik wurde durch Cantors *Reifizierung von vielen zu einem* bereichert – auch die Programmierung profitiert erheblich davon. So bieten objektorientierte Programmiersprachen sog. Collection-Objekte, die als Container von vielen Objekten fungieren und mit deren Hilfe man diese Vielzahl (diesen Plural) gemeinsam behandeln kann, ohne dafür die Objekte in einer expliziten Schleife – der sog. externen Iteration – einzeln aufzählen zu müssen. So erlaubt beispielsweise Smalltalk den Ausdruck

```
collection select: [ :x | x \\ 2 = 0 ] ,
```

der aus den Zahlen, die in der Collection `collection` zusammengefasst sind, diejenigen auswählt, die Zahlen sind (und sie in einer neuen Collection zurückgibt). Entsprechend liefert

```
collection collect: [ :x | x * 2 ]
```

eine neue Collection, in der das Doppelte aller Zahlen aus `collection` enthalten ist, und

```
collection inject: 0 into: [ :s :x | s + x ]
```

addiert die Zahlen in `collection` zu ihrer Summe. Diese drei Funktionen, einer breiteren Gemeinde auch unter *filter*, *map* und *fold* (oder *reduce*) bekannt, stammen aus der funktionalen Programmierung und werden dort auf Listen angewendet. Tatsächlich sind Listen in der funktionalen Programmierung so prominent, dass sie (im Gegensatz zu Collections in vielen objektorientierten Programmiersprachen) eine eigene Syntax und eine in der Sprachdefinintion (und nicht in einer Bibliothek!) festgelegte Semantik haben. Bei einer der ersten funktionalen Programmiersprachen standen Listen sogar im Vordergrund: Lisp steht für „List processor“.

Die Einführung der Menge (oder Liste) und damit die Zusammenfassung von vielen zu Einem hat aber die Betrachtung von vielen nicht überflüssig gemacht. So hat die Gleichung $x^2 = 4$ zwei Lösungen, -2 und 2, aber die Lösungsmenge $\{-2, 2\}$ ist nicht selbst die Lösung. $x = \{-2, 2\}$ zu schreiben wäre also falsch (und $x \in \{-2, 2\}$ ginge an der Sache vorbei); stattdessen findet man eher $x_1 = -2; x_2 = 2$, wobei das Semikolon und die Subskripte 1 und 2 ausdrücken, dass es zwei Lösungen gibt: -2 *und* 2, eben ein Plural („-2 und 2 *sind* die Lösung*en*“, nicht „-2 und 2 *ist* die Lösung“).

Dazu kommt, dass eine Menge eine andere Art von Objekt ist als ihre Elemente. Dies drückt sich u. a. darin aus, dass auf Mengen andere Operationen definiert sind als auf ihren Elementen: Der Term $\{1, 2\} + 3$ ist genauso bedeutungslos wie der Term $1 \in 2$.[2] In Programmiersprachen mit statischer Typprüfung gelten solche Terme oder Ausdrücke als fehlgeformt und werden zurückgewiesen; in Sprachen mit dynamischer Typprüfung kommt es zum einem Laufzeit(typ)fehler, in Smalltalk etwa zu „does not understand“. Diese Inkompatibilität von Mengen und ihren Elementen führt dazu, dass man in Programmen nicht nur an der Deklarationsstelle eines Programmelements zwischen Singular (genau

2 Zieht man allerdings die Grundlegung von Zahlen durch Mengen in Betracht, kann der eine oder andere solche Ausdruck doch sinnvoll sein.

einem Objekt) und Plural (einer beliebigen Anzahl) unterscheiden muss, sondern auch bei dessen Verwendungen. Dies steht im Gegensatz zu relationalen Sprachen wie SQL (Eisenberg und Melton 1999), UML (Object Management Group 2011) oder Alloy (Jackson 2006), bei denen sich die Unterscheidung auf die Deklaration mit sog. Multiplizitäten beschränkt. Es ist nicht vollständig klar, warum Programmiersprachen auf einer Unterscheidung im Typ (Elementtyp vs. Containertyp) beharren – vermutlich hat es damit zu tun, dass die für Container eingesetzten Sprachmittel auch für andere Datenstrukturen verwendet werden können, die Sprachdefinition also ohne Sonderbehandlung von Vielen ökonomischer ist (wie in Abschnitt 18.6 dargelegt, lassen sich auch Listen als Funktionen darstellen).

18.2 Plural in der Mathematik

Dass Mengen und andere strukturbildende Container über das Ziel, viele (oder einen Plural) zu repräsentieren, hinausschießen, ist bereits dem kanadischen Kollegen Eric Hehner aufgefallen (Hehner 1981; Hehner 1985; Hehner 1993): Insbesondere die strukturbildende Eigenschaft von Mengen, also dass $x \neq \{x\}$, (die zugleich die Reifizierung bedeutet) wird für die Darstellung von vielen gar nicht gebraucht. In seiner *Bunch theory* definiert Hehner daher ein Bunch[3] als den *Inhalt einer Menge*: So ist beispielsweise

- im Fall der Menge $\{-2, 2\}$ das dazugehörige Bunch $-2, 2$ (von mir gelesen als „-2 und 2“, ein Plural[4]),
- im Fall von $\{1\}$ das dazugehörige Bunch 1 (d. h., das Bunch 1 und sein einziger Konstituent 1 sind nicht unterscheidbar!) und
- im Fall von $\{\}$ (oder $\emptyset$) das dazugehörige Bunch nichts, repräsentiert etwa durch das Zeichen ϵ (analog dazu, wie das Zeichen ε gewöhnlich den leeren String repräsentiert — nichts lässt sich eben schlecht als nichts darstellen).

Für solche Bunches definiert Hehner dann eine ganze Reihe von Operationen und Relationen, die im Wesentlichen denen der Mengentheorie (Vereinigung, Durchschnitt etc.) entsprechen, jedoch anstelle von Element ($\in$) und Teilmenge ($\subseteq$) nur eine Relation vorsehen: Die *Subbunch*-Relation, mit „ : “ als dem Relator. Für diese Relation gilt u. a.

$$\begin{array}{rl}
\epsilon : B & \text{für alle Bunches } B, \\
e : e & \text{für alle elementaren Bunches (Singulare) } e, \\
e_1, e_2 : e_2, e_1 & \text{d. h., Reihenfolge ist irrelevant, sowie} \\
e_i, e_j : e_1, \dots, e_i, \dots, e_j, \dots, e_n & \text{d. h., Zusammenhängigkeit ist irrelevant.}
\end{array}$$

Wie man leicht erkennt, gelten alle diese Aussagen auch, wenn man die Bunches durch die Mengen, deren Inhalt sie sind, und die Subbunch-Relation durch die Teilmengenrelation ersetzt: $\{\} \subseteq \{B\}$ (wobei $\{B\}$ hier für eine beliebige Menge steht), $\{e\} \subseteq \{e\}$ usw. Man könnte also die Bunch-Theorie durch eine reduzierte Mengentheorie ersetzten, in der alles flache Mengen über Elementen sind und in der es die Element-Beziehung nicht gibt (Steimann und Freitag 2022). Wer jetzt an Alloy (Jackson 2006) denkt, hat gut aufgepasst.

3 Ich erkläre das Wort „Bunch“ hier zum Neutrum, in Anlehnug an „das Viele“, das hier allerdings nicht als Eines gedacht werden soll, sondern als Vielzahl. In einer anderen Arbeit verwende ich „eine Anzahl Objekte“ (Steimann 2022), was auch grammatikalisch als Vielzahl (Plural) durchgeht.

4 Genau wie „Adam und Eva“: „Adam und Eva lebte*n* im Paradies.“

Doch damit nicht genug. Hehner erweitert in der Folge sämtliche Operationen, die für die Konstituenten eines Bunches definiert sind, auf Bunches. So ergibt sich beispielsweise

$$1,2+1,2=(1+1),(1+2),(2+1),(2+2)=2,3,3,4=2,3,4\,,$$

wobei hier das Komma von Bunches stärker bindet als das Plus und $2,3,3,4$ und $2,3,4$ dasselbe Bunch repräsentieren und deswegen gleich sind. Ist mindestens einer der beiden Operanden nichts, erhalten wir nichts zum Ergebnis:

$$1,2+\epsilon=\epsilon\,.$$

Für Boolesche Bunch-Ausdrücke erhalten wir analog beispielsweise

$$\mathit{false},\mathit{true}\wedge\mathit{true}=(\mathit{false}\wedge\mathit{true}),(\mathit{true}\wedge\mathit{true})=\mathit{false},\mathit{true}$$

sowie für den einstelligen Operator $\neg$ (der ebenfalls schwächer bindet als das Komma)

$$\neg\,\mathit{false},\mathit{true}=\neg\mathit{false},\neg\mathit{true}=\mathit{true},\mathit{false}=\mathit{false},\mathit{true}\,.$$

Es werden also die Operatoren über ihre Operanden-Bunches distributiert. Wie man leicht erkennt, gehen dabei jedoch selbst einfachste Identitäten über Bord: So gilt weder $B+B=2\cdot B$ noch $B\wedge\neg B=\mathit{false}$ mehr.

Bevor man sich aber um den Verlust algebraischer Identitäten in Domänen wie der der Zahlen oder der Booleschen Wahrheitswerte Sorgen macht, sollte man sich erst einmal fragen, was ein Bunch von Zahlen oder Wahrheitswerten überhaupt bedeutet und ob es sinnvoll ist, mit Bunches zu rechnen. Zwar zeigt

$$-2,2\,^2=-2^2,2^2=4,4=4$$

in der Tat, dass -2 und 2 Lösungen von $x^2=4$ sind, aber keineswegs, dass dies alle Lösungen sind. Ansonsten scheinen „Zahlenplurale“ in der Mathematik ausschließlich in wohldefinierten Strukturen vorzukommen: Zähler und Nenner als Paare, die eine rationale Zahl bilden, x, y und z als Koordinaten eines Punkts in einem dreidimensionalen Raum oder als Komponenten eines Vektors, a_{11}, a_{12}, a_{21} und a_{22} als Elemente einer 2×2-Matrix usw. All diese mathematischen Objekte haben ihre eigenen Operatoren, Rechenregeln und Gesetzmäßigkeiten, die sich von denen für Zahlen und damit auch denen von Bunches von Zahlen unterscheiden (so dass diese Objekte schon deswegen nicht als Bunches aufgefasst werden können).

Während also echte Plurale von Zahlen in der reinen Mathematik eher am Rande existieren („mehrere Lösungen, OK, aber was noch?“), ändert sich das Bild, sobald wir die Mathematik Computern anvertrauen. Insbesondere in der Fließkommaarithmetik müssen wir nämlich damit leben, dass das berechnete Ergebnis nicht dem mathematisch exakten entspricht, sondern fehlerbehaftet ist (wobei die genaue Größe des Fehlers gemeinerweise unbekannt bleibt). Mit der sog. *Intervallanalyse* (Moore 1979) versucht man daher, das korrekte Ergebnis mit einem Intervall einzufangen, das gerade so groß ist, dass es das korrekte Ergebnis sicher einschließt. Die Regeln der Intervallarithmetik erlauben dann, mit dem erhaltenen Intervall weiterzurechnen, als wäre es eine Zahl, wobei das Versprechen ist, dass das mathematisch korrekte Endergebnis wiederum im errechneten Endintervall liegt. Mit einem Intervall zu rechnen heißt dabei mit all den Zahlen, die es enthält, parallel zu rechnen (genau wie bei der Rechnung mit den Bunches oben), wobei man in der Praxis versucht, sich auf die Intervallgrenzen zu beschränken (wobei man dann aber immer noch mit zwei Zahlen, also einem Plural, rechnet). So liest sich denn obige Beispielrechnung $1, 2 + 1, 2$ als „eine Zahl, 1 oder 2, ich weiß nicht so genau, plus eine zweite Zahl, auch (und unabhängig von der ersten) 1 oder 2, ergibt eine dritte Zahl, 2 oder 3 oder 4, je nachdem“. Wir haben es hier mit einer Form des Nichtdeterminismus (Hußmann 1993)[5] zu tun.

Aber auch dieser Form des Plurals möchte ich hier nicht weiter nachgehen. Mich interessiert vielmehr der Plural von Entitäten oder Objekten, wie man ihn in der relationalen und objektorientierten Modellierung antrifft.

18.3 Plural in der Modellierung

In der relationalen Datenmodellierung werden Beziehungen zwischen Objekten (oder Entitäten, wie sie dort in der Regel heißen) durch Relationen ausgedrückt. Dabei ist eine Relation eine Menge von Tupeln (bei zweistelligen Relationen eine Menge von Paaren), von denen jedes einzelne das Bestehen einer Beziehung zwischen den das Tupel konstituierenden Objekten ausdrückt. Steht ein Objekt mit mehreren anderen in derselben Beziehung, kommt das Objekt in mehreren Tupeln der entsprechenden Relation vor. Grenzen der Anzahl der Vorkommen werden durch sog. Kardinalitäten ausgedrückt, die im Kontext der objektorientierten Modellierung auch Multiplizitäten genannt werden. Eine Multiplizität von 1 (für „genau ein“) steht dabei für einen Singular, andere Multiplizitäten für Plurale.

Der Charme der Verwendung von Relationen zur Darstellung von Beziehungen ist, dass sich die Beziehungen damit auf sehr einfache und von der Kardinalität unabhängige Art und Weise verfolgen lassen, und zwar mithilfe der Verkettung von Relationen mittels des sog. relativen Produkts, im Kontext relationaler Datenbanken auch als Join-Operation bekannt. So ergibt das relative Produkt der Relation

$$\{(hof_1, baum_1), (hof_1, baum_2), (hof_2, baum_3)\}$$

mit der Relation

$$\{(baum_1, apfel_1), (baum_2, apfel_2), (baum_4, apfel_3)\}$$

die Relation

$$\{(hof_1, apfel_1), (hof_1, apfel_2)\},$$

d. h., man gelangt per Navigation der beiden Relationen von hof_1 zu $apfel_1$ und $apfel_2$, aber sonst von keinem Objekt hof_i zu irgendeinem anderen Objekt $apfel_j$. Dabei ist es für den Navigationsausdruck (das relative Produkt) vollkommen unerheblich, mit wie vielen anderen Objekten ein Objekt jeweils in Beziehung steht oder auch nur stehen darf.

Während Relationen Beziehungen zwischen Objekten darstellen, werden Objekte durch sog. Attribute jedes für sich beschrieben. Dabei ordnet ein Attribut einem Objekt in der Regel genau einen Wert zu (also etwa eine Zahl, einen String oder einen Wahrheitswert); manche Modellierungssprachen (wie beispielsweise UML) erlauben aber auch die Zuordnung eines Objekts oder auch mehrerer Werte oder Objekte, wobei deren Zahl wieder durch

5 † 23. Mai 2022

eine Kardinalität (oder Multiplizität) begrenzt werden kann. Dabei werden Attribute aber nicht als Relationen, sondern als Abbildungen interpretiert, d. h. die Zuordnung ist stets gerichtet (vom Objekt zum Wert) und eindeutig. Letzteres wiederum bedeutet, dass mehrere Werte oder Objekte zu einem zusammengefasst werden müssen: Ein Attribut ist dann nicht *mehr*wertig, sondern z. B. *mengen*wertig (d. h., der Attributwert ist eine Menge). Für einen zweifarbigen Apfel schreibt man dann beispielsweise

$$\mathit{farbe}(\mathit{apfel}_1) = \{\mathit{rot}, \mathit{gelb}\},$$

wobei dies sachlich so nicht richtig ist, denn die Menge $\{\mathit{rot}, \mathit{gelb}\}$ ist keine Farbe (und das Attribut müsste entsprechend *farbmenge* heißen). Würde man den Sachverhalt als Relation (Beziehung zwischen Objekten und Farben) ausdrücken wollen, so würde man vielleicht

$$\{(\mathit{apfel}_1, \mathit{rot}), (\mathit{apfel}_1, \mathit{gelb})\}$$

schreiben, was jedoch eine andere Bedeutung hat, weil hier apfel_1 (sachlich richtig) mit zwei Farben ins Verhältnis gesetzt wird und nicht mit einer Menge von Farben.[6] Wollte man genau diesen Sachverhalt mit einem Attribut ausdrücken, dann könnte man ein Farb-Bunch verwenden:

$$\mathit{farbe}(\mathit{apfel}_1) = \mathit{rot}, \mathit{gelb},$$

wobei die Bedeutung des Bunches hier konjunktiv („die Farben von apfel_1 sind rot *und* gelb") und nicht disjunktiv („die Farben von apfel_1 ist rot *oder* gelb", also nichtdeterministisch) ist. Entsprechend würde

$$\mathit{farbe}(\mathit{apfel}_1) = \epsilon$$

bedeuten, dass apfel_1 keine Farbe hat, ein Umstand, den man im Fall von Farbe und Äpfeln wohl durch ein Kardinalitätsconstraint im Domänenmodell ausschließen würde, der aber in anderen Fällen durchaus sinnvoll ist ($\mathit{kind}(p_1) = \epsilon$ etwa heißt, dass p_1 keine Kinder hat).

Wie an anderer Stelle (Steimann und Freitag 2022) ausgeführt, können wir mittels bunchwertiger Attribute Relationen spezifizieren: Wenn wir ein Attribut *attr* nicht als Funktion, sondern als Relation auffassen, dann können wir für alle Objekte *o* festlegen:

$$\mathit{attr}(o) = o_1, \ldots, o_n \quad \Leftrightarrow \quad \{o' \mid (o, o') \in \mathit{attr}\} = \{o_1, \ldots, o_n\}$$

d. h., die bunch-wertige Funktion *attr* ist vielmehr eine Relation, die durch Attributgleichungen der Form $\mathit{attr}(o) = o_1, \ldots, o_n$ (und zwar genau eine pro Objekt *o*, das in der Relation an deren erster Stelle vertreten ist) definiert wird. Ist $\mathit{attr}(o) = \epsilon$, dann kommt *o* nicht in *attr* vor (genauer: dann gibt es kein o', so dass $(o, o') \in \mathit{attr}$).

Wir halten fest, dass durch Bunches genau wie durch Relationen Beziehungen einzelner Objekte zu mehreren Objekten und somit zu echten, d. h. *nicht reifizierten* Pluralen ausgedrückt werden können. Dies ist in Programmiersprachen bislang eher nicht der Fall.

6 Jedoch werden Farben in den allermeisten Domänen als Werte und nicht als Objekte aufgefasst, sind also nicht Gegenstand von Beziehungen; eine Ausnahme bilden Domänen, in denen Farben selbst der Gegenstand der Betrachtung sind (Steimann 2021).

18.4 Plural für die Programmierung

Die Programmierung hat (mit Ausnahme vielleicht der logischen Programmierung) traditionell einen funktionalen Einschlag: In imperativen Programmiersprachen werden Programme in Funktionen und Prozeduren (Funktionen ohne Rückgabewert) zerlegt und in der funktionalen Programmierung ist sowieso beinahe alles eine Funktion. Ein großer praktischer Vorteil von Funktionen ist dabei, dass man sie leicht verketten kann: Das Ergebnis eine Funktionsaufrufs kann selbst als Argument eines weiteren Funktionsaufrufs dienen. Durch diese Möglichkeit der Schachtelung lassen sich nicht nur komplexe Berechnungen, sondern auch lange Navigationspfade als kompakte Ausdrücke aufschreiben. Letzteres ist insbesondere für die objektorientierte Programmierung von Interesse.

18.4.1 Attribute (Felder)

Attribute oder, wie sie in der Programmierung eher heißen, Felder (Instanzvariablen in Smalltalk) ordnen Objekten andere Objekte oder Werte zu. Ähnlich wie die Attribute der Datenmodellierung oben lassen sich auch Felder als Funktionen auffassen, die das attributierte Objekt auf Objekte oder Werte abbilden. Es entspricht also der Zugriff auf ein Feld `f` eines Objekts `o`, häufig als `o.f` geschrieben,[7] der Funktionsanwendung $f(o)$, wobei das Ergebnis der Anwendung eben der Attributwert ist (der jedoch technisch nicht berechnet, sondern dem Speicher entnommen wird). Ist das Ergebnis wiederum ein Objekt, das über ein Feld `g` verfügt, dann entsprecht der Kettenausdruck `o.f.g` dem Funktionsausdruck $g(f(o))$. So weit so gut.

Ein Problem entsteht jedoch (einmal mehr), wenn ein Attribut einen Plural darstellt. Dann nämlich verlangt die funktionale Sichtweise, dass dieser Plural wieder als genau ein Objekt oder Wert dargestellt, also reifiziert wird. Bezeichnet also das Feld `f` einen Plural, so ergibt `o.f` ein Collection-Objekt. Damit wäre aber `o.f.g` kein gültiger Navigationsausdruck mehr, weil nämlich `o.f`, eine Collection, kein Feld g hat. Stattdessen muss man eine interne Iteration bemühen, also je nach Sprache so etwas wie `o.f.map(x -> x.g)` (entsprechend `o f collect: [ :x | x g ]` in Smalltalk) schreiben, was sich jedoch erheblich vom Singular-Fall `o.f.g` (bzw. `o f g`) unterscheidet. An dieser Stelle kommen nun Bunches ins Spiel. Wie wir oben gesehen haben, distributieren Operatoren über die Konstituenten eines Bunches: $(1, 2) + 3 = (1 + 3), (2 + 3) = 4, 5$. Das gilt auch für einstellige Operatoren: $-(1, 2) = -1, -2$. Wenn nun Funktionsanwendungen ebenfalls über Bunches distributieren, dann erhalten wir für $f(o) = o_1, \dots, o_n$

$$g(f(o)) = g(o_1), \dots, g(o_n)$$

oder, in Attributschreibweise, für $\mathtt{o.f} = \mathtt{o}_1, \dots, \mathtt{o}_n$

$$\mathtt{o.f.g} = \mathtt{o}_1\mathtt{.g}, \dots, \mathtt{o}_n\mathtt{.g}\,.$$

Wenn die $\mathtt{o}_i\mathtt{.g}$ nun selbst wieder Bunches sind, also $\mathtt{o}_i\mathtt{.g} = \mathtt{o}_{i,1}, \dots, \mathtt{o}_{i,m_i}$, dann haben wir

$$\mathtt{o.f.g} = \mathtt{o}_{1,1}, \dots, \mathtt{o}_{1,m_1}, \dots, \mathtt{o}_{n,1}, \dots, \mathtt{o}_{n,m_n}\,,$$

also wieder ein Bunch.

7 Genau wie ein Feldzugriff bei einem Record in Pascal und anderen Algol-Derivaten

Es fällt auf, dass im Ergebnis die Verknüpfung der Funktionen g und f, $g \circ f$, genau dem relativen Produkt der entsprechenden Relationen f und g entspricht: das relative Produkt von

$$f = \{(o, o_1), \dots, (o, o_n)\}$$

und

$$g = \{(o_1, o_{1,1}), \dots, (o_1, o_{1,m_1}), \dots, (o_n, o_{n,1}), \dots, (o_n, o_{n,m_n})\}$$

ergibt

$$\{(o, o_{1,1}), \dots, (o, o_{1,m_1}), \dots, (o, o_{n,1}), \dots, (o, o_{n,m_n})\} = g \circ f.$$

Wir sehen also, dass sich, wenn wir Bunches anstelle von Collections verwenden, die Navigationsausdrücke für Plurale nicht von denen für Singulare unterscheiden – wir können sie vielmehr einheitlich behandeln (Steimann und Freitag 2022).

18.4.2 Funktionen (Methoden)

Nachdem wir nun Attribute als (einstellige) bunch-wertige Funktionen und diese als Relationen aufgefasst sowie gezeigt haben, was die Funktionsanwendung auf Bunches bedeutet, liegt es nahe, dasselbe Prinzip auch auf Methoden anzuwenden: Ein Methodenaufruf auf einem Bunch distributiert über die Konstituenten des Bunches. Praktisch würde das bedeuten, dass der Methodenaufruf auf einem Bunch zu gleichen Methodenaufrufen auf allen das Bunch konstituierenden Objekten auflöst (oder eben auf keinem, wenn das Bunch leer ist). Die Pseudovariable `this` (oder `self` in Sprachen wie Smalltalk) im Rumpf einer derart aufgerufenen Methode verweist damit immer auf genau ein Objekt. Man nennt diese Art der Funktionsanwendung deswegen auch *Singular-Semantik* (Steimann und Freitag 2022).

Singular-Semantik

Ein Methodenaufruf `o.m()` (oder `o m` in Smalltalk) entspricht im Wesentlichen einem dynamisch gebundenen Funktionsaufruf `m(o)`, wobei die Wahl der Implementierung von `m` am Typ (der Klasse) von `o` hängt. Ist `o` nun ein Bunch, dann distributiert der Aufruf von `m` über alle Konstituenten von `o`, wobei jeder einzelne Funktionsaufruf dynamisch gebunden wird und die Funktionswerte (die Rückgabewerte der Methoden) wieder in einem Bunch eingesammelt werden (Steimann 2022). Die entspricht in etwa dem Beispiel von `collect` aus Abschnitt 18.1, wobei hier der gesamte Aufbau des Ausdrucks vollkommen regelmäßig (ohne Variationsmöglichkeit) ist:

```
o collect: [ :e | e m ]
```

in Smalltalk oder, etwas wortreicher,

```
o.stream().map(e -> e.m()).toList()
```

in Java. Der Aufruf von `o.m()` bzw. `o m` entspricht also dem der aus der funktionalen Programmierung bekannten Funktion $map(o, m)$.

Eine Filter-Funktion $filter(o, cond)$ lässt sich entsprechend als Ausdruck `o cond` bzw. `o.cond()` formulieren, wobei der Rumpf der Methode `cond` dann die Form

```
(...) ifTrue: [^self] ifFalse:[^nothing]
```

bzw.

```
return (...) ? this : nothing;
```

haben muss, die abhängig vom Ergebnis der Bedingung entweder das Objekt, auf das sie angewendet wird, oder nichts (mit `nothing` für ϵ) zurückgibt. Das Ergebnis sind, genau wie bei *map*, wieder viele (ein Bunch, also möglicherweise auch keine) Objekte. Die Umsetzung von *reduce* ist allerdings nicht so leicht; solange wir keine Dekomposition von Bunches vorgesehen haben (analog zur Dekomposition von Listen mittels *head* und *tail*; s. Abschnitt 18.6), die eine rekursive Verarbeitung von Bunches erlauben würde, brauchen wir hierfür einen globalen (d.h., über den Aufruf einer Methode hinaus bestehenden) Akkumulator (nicht unmöglich, aber auch nicht schön).

Semantik der Parameter

Betrachtet man Methoden mit Parametern und fügt einem Methodenaufruf entsprechend ein Argument hinzu, stellt sich die Frage, ob auch über dieses Argument distributiert werden soll, so wie das bei den zweistelligen Operationen in Abschnitt 18.2 der Fall war, oder ob ein Bunch an der Stelle eines Arguments als Ganzes an den formalen Parameter übergeben werden soll. Entscheidet man sich für letzteres, so kann das „Kreuzprodukt“[8] von Empfänger- und Argument-Bunch immer noch durch die Methode des *Double dispatching* (Ingalls 1986; Steimann 2022) erzeugt werden:

```
class Number {
  Number plus(Number plural) { return plural.plus2(this); }
  Number plus2(Number singular) { return singular + this; }
}
```

Wenn nun `b1` und `b2` zwei Bunches von Zahlen sind, dann führt der Aufruf von `b1.plus(b2)` zunächst zur Ausführung der Methode `plus(b2)` auf allen Konsituenten von `b1`, wobei in deren Rumpf in der Folge auf jedem der Konstituenten von `b2` die Methode `plus2` mit dem aktuellen Empfänger (je einem Konstituenten aus `b1`, einem Singular) als Argument aufgerufen wird. Im Rumpf der Methode `plus2` steht dann das Paar bestehend aus `singular` und `this` für genau ein Element aus dem kartesischen Produkt von `b1` und `b2`, dessen Argumente so alle aufgezählt werden. Umgekehrt würde eine Distributierung auch über das Argument-Bunch schon beim Aufruf von `plus` zum kartesischen Produkt aus Empfängern und Argumenten führen; dies hätte aber u. a.. den Nachteil, dass eine Methode gar nicht ausgeführt wird, wenn ein Argument das leere Bunch ist.

Plural-Semantik

Wenn wie eben dargelegt die Zuweisung eines Bunch-Arguments an einen formalen Parameter als Ganzes sinnvoll sein kann, dann kann man fragen, ob selbiges nicht auch für den Empfänger gelten soll, ob also eine Methode nicht auch auf einem Bunch als Ganzes aufgerufen werden können soll. Tatsächlich wäre es ohne dies unmöglich, Methoden wie die folgende zu implementieren, deren Zweck es ist, zwei Bunches auf Gleichheit zu testen:

8 In Anführungsstrichen, weil das Kreuzprodukt auf Mengen und nicht auf Bunches definiert ist

```
class Object {
  boolean equal(Object that) {
    return this.in(that) && that.in(this);
  }
}
```

wobei hier die Methode `in(_)` der Bunch-Inklusion „ : “ aus Abschnitt 18.2 entsprechen soll. Führt man hingegen Methoden ein, bei deren Aufruf `this` (oder besser eine Pseudovariable `these`, die den Plural repräsentiert und `this` in solchen Methoden ersetzt) auf das gesamte Bunch verweist, auf dem aufgerufen wurde, dann tut obige Methode genau das, was sie soll (Steimann 2022). Allerdings muss eine solche Plural-Methode statisch gebunden werden (schon allein deswegen, weil sie auch auf einem leeren Bunch aufgerufen werden kann; man beachte, dass leere Bunches zwar einen statischen, aber mangels Masse keinen dynamischen Typ haben können; mehr dazu unten). Eine solche sog. *Plural-Semantik* (Steimann und Freitag 2022) kann übrigens auch im mathematischen Kontext aus Abschnitt 18.2 sinnvoll sein: Während die Anwendung einer Funktion

$$\mathit{double}(x) = x + x$$

auf das Bunch 1, 2 unter Singular-Semantik zu 2, 4 auswertet und damit nicht dem (scheinbar) gleichwertigen Ausdruck $1, 2 + 1, 2$ entspricht (der ja zu 2, 3, 4 auswertet), so würde dieselbe Funktion unter Plural-Semantik ebenfalls das Ergebnis von $1, 2 + 1, 2$ haben, weil ja x im Rumpf der Definition jeweils durch das Bunch (und nicht in einer Art Schleife der Reihe nach durch seine beiden Konstituenten) ersetzt würde.

18.5 Praktischer Nutzen der Einführung des Plurals in die Programmierung

Damit man eine Vorstellung davon entwickeln kann, was die Einführung eines Plurals wie oben skizziert für die Programmierpraxis bringen kann, ist es notwendig, sich erst einmal damit zu befassen, wie der Alltag ohne einen Plural aussieht. In einem solchen Alltag müssen wir nämlich die Behandlung von vielen in Formen des Singulars ausdrücken: Anstatt einfach „wir staunen“ sagen zu können, müssen wird „jeder von uns staunt“ sagen, und das bei jeder Gelegenheit. Staune ich dagegen allein, reicht ein simples „ich staune“. Doch damit nicht genug, wie die folgenden Betrachtungen am Beispiel der Programmiersprache Java zeigen.

18.5.1 Facetten des programmiertechnischen Problems der Unterscheidung des Umgangs mit einem und vielen

Abbildung 18.1 zeigt zwei Varianten einer Implementierung des Observer-Musters in Java (aus Steimann 2022 entnommen): Auf der linken Seite hat jedes Subjekt genau einen Observer und im Rumpf des Programms wird genau ein Subjekt behandelt, während es auf der rechten Seite von beiden jeweils mehrere geben kann. Wie man leicht erkennt, sind die Unterschiede beträchtlich:

1. *Anzahl bestimmt Typ* In Zeile 25 von Abb. 18.1, links, erlaubt die Deklaration der Variable `o` die Zuweisung eines Objekts vom Typ `Observer`. Sollten es stattdessen viele sein, können wir nicht einfach `Collection<Observer> os = new Observer()`

```java
1  class Subject {
2    final IObserver observer;
3    Subject(IObserver o) {
4      assert o != null;
5      observer = o;
6      IObserver backup = o;
7      o = null;
8      if (⋯) o = backup;
9    }
10   IObserver getObserver() {
11     return observer;
12   }
13   void notify() {
14     observer.update(this);
15   }
16 }
17 interface IObserver {
18   void update(Subject s);
19   Collection<Element> getElems();
20 }
21 class Observer
22 implements IObserver { ⋯ }
23 class Element { ⋯ }
24
25 Observer o = new Observer();
26 Subject s = new Subject(o);
27 Collection<Element> es =
28   s.getObserver().getElems();
```

```java
class Subject {                                              1
  final Collection<IObserver> observers;                     2
  Subject(Collection<? extends IObserver> os)                3
        {                                                    4
    assert !os.isEmpty();                                    5
    observers = new ArrayList<>(os);                         6
    Collection<? extends IObserver> backup = os;             7
    os.clear();                                              8
    if (⋯) os = backup;                                      9
  }                                                          10
  Collection<IObserver> getObservers() {                     11
    return java.util.Collections.                            12
      unmodifiableCollection(observers);                     13
  }                                                          14
  void notify() {                                            15
    for (IObserver o : observers)                            16
      o.update(this);                                        17
  }                                                          18
}                                                            19
interface IObserver ⋯                                        20
class Observer ⋯                                             21
class Element ⋯                                              22
                                                             23
Collection<Observer> os =                                    24
  new Arrays.asList(new Observer());                         25
Collection<Subject> ss =                                     26
  new Arrays.asList(new Subject(os));                        27
Collection<Element> es = ss.stream()                         28
  .flatMap(e -> e.getObservers().stream())                   29
  .flatMap(e -> e.getElems().stream())                       30
  .toList();                                                 31
```

Abbildung 18.1: Links: Implementierung des Observer-Musters in Java, mit einem Observer pro Subjekt. Rechts: Das gleiche Programm wie links, jedoch mit vielen Observern und vielen Subjekten. Die semantischen Differenzen sind hervorgehoben; man sieht sofort, dass sie sich über das gesamte Programm erstrecken (entnommen aus Steimann 2022).

schreiben, da dies einen Typfehler darstellt. Stattdessen müssen wir das Objekt wie in Zeile 25, rechts, zunächst in eine passende Collection packen. Auch können wir auf einem Objekt Methoden direkt aufrufen (Zeile 14, links), während wir im Fall von vielen die Objekte erst auspacken müssen (Zeile 16, rechts).

2. *Anzahl bestimmt die Regeln des Subtypings* Während Variablen eines Typs Objekte seiner Subtypen zugewiesen werden können (wie am Beispiel des Methodenaufrufs aus Zeile 26, links, ersichtlich, der ein Argument vom Typ `Observer` an den formalen Parameter vom Typ `IObserver` in Zeile 3 zuweist), sind zumindest änderbare Collections bezüglich ihrer Elementtypen invariant: In Zeile 4, rechts, muss daher ein Wildcard-Typ verwendet werden, der das Hinzufügen von Objekten zur Collection verbietet.

3. *Anzahl bestimmt die Repräsentation von „kein Objekt“* Für Variablen, die ein Objekt aufnehmen können, wird „kein Objekt“ in der Regel durch den Wert `null` repräsentiert (s. Zeile 4, links, für ein Beispiel, in dem „kein Objekt“ ausgeschlossen werden soll). Für Variablen, die viele Objekte aufnehmen können sollen und deswegen einen Collection-Typ haben, wird der gleiche Umstand durch eine leere Collection ausgedrückt (Zeile 5, rechts).

4. *Anzahl bestimmt, wie mit „kein Objekt“ sicher umgegangen wird* Da die Dereferenzierung von `null` einen Laufzeitfehler bedeutet (Tony Hoares berühmt-berüchtigter „billion dollar mistake“), ist der Methodenaufruf in Zeile 14, links, unsicher: Der Java-Compiler kann nicht ableiten, dass die Zusicherung aus Zeile 4 auch hier gilt. Im

Gegensatz dazu ist der Aufruf auf `observers` in Zeile 16 f., rechts, inhärent sicher: Bei einer leeren Collection wird über nichts iteriert, so dass auch nichts passiert.

5. *Anzahl prägt Verkettung* Wie bereits zuvor bemerkt bestimmt die Anzahl, wie die Verkettung von Feldzugriffen oder Methodenaufrufen aussieht. Im aktuellen Beispiel wird dies aus dem Gegensatz von Zeile 28, links, und Zeile 28 f., rechts, noch einmal ersichtlich.

6. *Anzahl bestimmt Kapselungsstrategie* Während Felder, deren Inhalt einzelne Objekte sind, in der Regel direkt (ggf. über Konstruktoren, Setter und Getter) geschrieben und gelesen werden können (s. z.BZeilen 11 und 5, links), sind Collections in aller Regel Repräsentationsobjekte, die vor Zugriff von außen verborgen werden sollen (Zeilen 5 und 11 f., rechts).

7. *Anzahl bestimmt das Teilen von Daten* Durch die Verwendung von Containern können sich zwei Variablen logisch denselben Speicherbereich teilen, und zwar selbst in Sprachen wie Java oder Smalltalk, in denen das sonst nicht geht. Ein Beispiel hierfür findet sich in den Zeilens 7–9, rechts, wo nach der Zuweisung von Zeile 7 die Variablen `os` und `backup` dieselbe Collection bezeichnen. Die Änderung des Inhalts von `os` in Zeile 8 betrifft somit auch `backup`, so dass die Zuweisung in Zeile 9 nicht den erhofften Effekt hat. Dies ist auf der linken Seite anders.

8. *Anzahl bestimmt Aufrufsemantik* Analog zu obigem Beispiel ist es in Sprachen wie Java und Smalltalk auch nicht möglich, Methoden per Call-by-reference aufzurufen. Gleichwohl handelt es sich bei einem Aufruf, bei dem (ein Zeiger auf) eine Collection by-value übergeben wird (wie beim Aufruf von `new Subject(os)` in Zeile 27, rechts), um einen Call-by-reference, da die Methode den Inhalt des übergebenen Containers und damit den Plural ändern kann (wie durch den Aufruf von `clear()` auf dem formalen Parameter in Zeile 8). Für den Singular (Aufruf von `new Subject(o)` in Zeile 26, links) gilt das nicht (die Zuweisung von `null` zum formalen Parameter des Konstruktors in Zeile 7 hat keinen Effekt außerhalb des Konstruktors).

9. *Anzahl bestimmt die Bedeutung von* `final` Der Zugriffsmodifizierer `final` bedeutet in Java, dass einer Variable nach ihrer Initialisierung kein neuer Wert mehr zugewiesen werden kann. Während das für einzelne Objekte bedeutet, dass die Variable Zeit ihres Lebens immer auf dasselbe Objekt verweist (Zeile 2, links), ist dem für viele nicht so: Der Inhalt einer Collection kann über die Variable uneingeschränkt manipuliert werden (Zeile 2, rechts).

18.5.2 Beschränkung der Unterschiede auf die Deklaration

Das Bild ändert sich schlagartig, wenn man Bunches als neues Sprachkonstrukt einführt und die Deklaration von Programmelementen, die zu Bunches auswerten, in zwei Dimensionen auftrennt: *Typ* und *Numerus*. Um zu sehen, wie das funktioniert, denken wir uns eine Sprache Num (Steimann 2022), in der bei der Deklaration von Programmelementen neben deren Typ auch deren Numerus angegeben wird, und zwar per

! für „genau ein“ (Singular),

? für „ein oder kein“ und

* für „beliebig viele“ (Plural).

So deklariert (und initialisiert) `Observer! o = new Observer;` eine Variable `o`, die stets für genau ein Objekt vom Typ `Observer` steht (und deswegen bereits bei ihrer Deklaration initialisiert werden muss), und `Observer* os = no Observer;` eine Variable, die beliebig viele Objekte vom Typ `Observer` aufnehmen kann (einschließlich keines, mit dem sie hier initialisiert wird). Zusätzlich gibt es in Num einen Numerus

- für „kein“;

es handelt sich dabei um den Numerus von `no T`, einem Ausdruck, der für „kein Objekt vom Typ `T`“ steht und der u. a.. von Methoden, die nichts zurückgeben sollen, in ihrem Return-Statement verwendet wird (`no T` ist das Pendant von `new T`, dessen Numerus `!` ist; `Object-` ersetzt gewissermaßen `void`, wobei in Num „nichts“ nach Typ differenziert wird). Singulare und Plurale werden dabei in Num intern gleichermaßen als Bunches repräsentiert und bei der Zuweisung eines Bunches an eine Variable wird das Bunch (d. h., die Zeiger auf alle Objekte, die es konstituieren) kopiert – es gibt also keinen Alias auf einen Container, der im Beispiel von Abb. 18.1 die Ursache für die meisten Unterschiede zwischen Singular und Plural war. Zuweisungskompatibilität ist gegeben, wenn die rechte Seite einer Zuweisung einen Subtyp *und einen Subnumerus* der linken Seite aufweist, wobei die Subnumerus-Relation $\lhd^{\#}$ durch den reflexiven und transitiven Abschluss von

$$\{(- \lhd^{\#} ?), (! \lhd^{\#} ?), (? \lhd^{\#} *)\}$$

gegeben ist. Neben dem *statischen Numerus*, der ausschließlich der statischen Numerus-Prüfung dient (parallel zur statischen Typprüfung), gibt es in Num auch einen *dynamischen Numerus* (wiederum analog zum dynamischen Typ), der programmatisch durch Betragsstriche `|`*e*`|` um einen Ausdruck *e* abgefragt wird, mit der Anzahl der konstituierenden Objekte als Ergebnis (analog zur Methode `getClass()` in Java). Entsprechend wird die Anzahl bei einem Numerus-Downcasts, so z.Bvon ? zu `!`, implizit geprüft: Wertet im Ausruck `(!)` *e* der Ausdruck *e* zu keinem oder mehr als einem Objekt aus, tritt ein Laufzeit-Numerusfehler auf (in Analogie zu einem Laufzeit-Typfehler bei einem Type cast).

Die Fassung der beiden Versionen der Implementierung des Observer-Musters in Abbildung 18.1 in Num ist in abbildung 18.2 dargestellt. Wie man leicht erkennt, beschränken sich die Differenzen nunmehr nahezu ausschließlich auf die Deklarationen:

1. *Typ unabhängig von Numerus* Wie man der Gegenüberstellung der Deklarationen der Felder `observer` und `observers` sowie der Variablen `o` and `os` aus Abb. 18.1, links (Zeilen 2 und 25) und rechts (Zeilen 2 und 24), mit denen von `IObserver! observer`, `IObserver* observers`, `Observer? o` und `Observer* os` von Abb. 18.2, Zeilen 2 und 24, entnehmen kann, unterscheiden sich Felder und andere Variablen nur noch im Numerus, aber nicht mehr in ihrem Typ.

   Die Zuweisung `os = new Observer` von Zeile 24, rechts, ist daher genauso wohlgetypt wie der Methodenaufruf `observers.update(this)` in Zeile 14, rechts. Man beachte, dass die Zuweisungen in Zeile 24 auch „wohlnummeriert“ sind, da der Numerus des `new` Ausdrucks `!` ist, während der der Variablen ? und `*` ist, und $! \lhd^{\#} ?$ und $! \lhd^{\#} *$ gilt.

2. *Subtyping unabhängig vom Numerus* Da Typ und Numerus nunmehr unabhängig sind, sind die beiden Zuweisungen `this.observer = (!) o` und `this.observers = os` (Zeile 5, links und rechts, in Abb. 18.2) typrecht.

3. *Einheitliche Kodierung von „kein Objekt“* Kein Objekt wird einheitlich als leeres Bunch dargestellt, in der Sprache Num durch `|`*e*`| == 0` detektiert, unabhängig davon,

```
 1  class Subject extends Object {                   class Subject extends Object {
 2    IObserver! observer = new Observer;              IObserver* observers = new Observer;
 3    Object- init!(IObserver? o) {                    Object- init!(IObserver* os) {
 4      assert false |o| == 0;                           assert false |os| == 0;
 5      this.observer = (!) o;                           this.observers = os;
 6      IObserver! backup = (!) o;                       IObserver* backup = os;
 7      o = no Observer;                                 os = no Observer;
 8      if (...) {o = backup;} else {}                   if (...) {os = backup;} else {}
 9    }                                                }
10    IObserver! getObserver!() {                      IObserver* getObservers!() {
11      return this.observer;                            return this.observers;
12    }                                                }
13    Object- notify!() {                              Object- notify!() {
14      this.observer.update(this);                      this.observers.update(this);
15    }                                                }
16  }                                                }
17  class IObserver extends Object {                 class IObserver extends Object {
18    Object- update!(Subject! s) {...}                Object- update!(Subject! s) {...}
19    Element* getElems!() {...}                       Element* getElems!() {...}
20  }                                                }
21  class Observer extends IObserver {...}           class Observer extends IObserver {...}
22  class Element extends Object {...}               class Element extends Object {...}
23
24  Observer? o = new Observer;                      Observer* os = new Observer;
25  Subject! s = new Subject; s.init(o);             Subject* ss = new Subject; ss.init(os);
26  Element* es = s.getObserver().getElems();        Element* es = ss.getObservers().getElems();
```

Abbildung 18.2: Implementatierung der beiden Versionen des Observer-Musters aus Abb. 18.1 in Num. Die semantischen Unterschiede sind wieder hervorgehoben; man erkennt leicht, dass sie sich im Gegensatz zu Abb. 18.1 auf wenige Stellen (im wesentlichen Deklarationen) beschränken (wieder aus Steimann 2022).

ob das Argument *e* den (statischen) Numerus ? oder * hat (Zeile 4, links und rechts, in Abb. 18.2).

4. *Einheitliche Null-Sicherheit* Da es nun kein `null` (und damit keinen Nullzeiger) mehr gibt, gibt es auch keine fehlerhafte Dereferenzierung desselben mehr („kein Objekt“ wird auch nicht dereferenziert). Zugleich wird durch die Verwendung des Numerus ! in Zeile 2, links, sichergestellt, dass in Zeile 14 immer genau ein Objekt vorliegt, so dass ein fehlerhaftes Fehlen eines Empfängers des Methodenaufrufs nicht unbemerkt bleiben kann (ganz einfach, weil es nicht vorkommen kann). Im Plural-Fall (rechts) gibt es diese Garantie nicht; dafür müsste man Num einen weiteren Numerus + für „ein oder mehr“ bereitstellen.

5. *Verkettung unabhängig von Anzahl* Wie in Aussicht gestellt unterscheiden sich Kettenausdrücke nun nicht mehr nach dem Numerus der einzelnen Segmente: Der Ausdruck aus Zeilen 28 f. in Abb. 18.1, rechts, wird ersetzt durch den Ausdruck `ss.getObservers().getElems()`, der, abgesehen von den bedeutungslosen Pluralformen der Namen, dem Ausdruck für den Singular-Fall entspricht (Abb. 18.2, Zeile 26).

6. *Kapselung unabhängig von Anzahl* Da Einzahl und Mehrzahl einheitlich durch Bunches repräsentiert werden und diese bei Weitergabe immer kopiert werden, ist eine gesonderte Kapselung der Repräsentation im Plural-Fall nicht notwendig.

7. *Daten werden nie geteilt* Aus dem gleichen Grund können Bunches (als Einheit) nicht geteilt werden (die konstituierenden Objekte jedes für sich natürlich schon).

8. *Aufrufsemantik immer einheitlich* Dasselbe gilt bei Aufrufen: Auch viele werden per Call-by-value (also als Kopien der Zeiger) übergeben.

9. *Einheitliche Bedeutung von* `final` Auch hier gibt es keinen Unterschied mehr: Eine `final` deklarierte Variable mit Numerus `*` ist genauso wenig überschreibbar wie eine mit Numerus `!` oder `?` .

Man beachte, dass für den Singular-Fall die Zusicherung in Zeile 4, links, obsolet ist, da der Numerus-Cast `(!)` in der nächsten Zeile, der von der statischen Numerus-Prüfung der Zuweisung verlangt wird, ebenfalls zu einem Laufzeitfehler führt, wenn `o` nicht genau ein Objekt liefert. Der statische Numerus von `obsever` (nämlich `!`), der den Cast verlangt, garantiert im Gegenzug, dass der Aufruf von `update(this)` in Zeile 14 niemals nichts tut. Allgemein kann man sich gegen Programmierfehler, durch die fälschlicherweise nichts aufgerufen wird (ohne dass dies wie früher durch eine Null pointer exception signalisiert würde), absichern, indem man Numerus-Casts einfügt: So kann man den verketteten Zugriffsausdruck aus Abb. 18.2, Zeile 26, links, durch `(!)((!)s.getObserver()).getElems()` ersetzen, dessen Auswertung mit einem Laufzeit-Numerus-Fehler abbricht, wenn `s` oder `s.getObserver()` zu keinem Objekt auswerten (gesetzt den Fall, dies wäre qua Deklaration überhaupt erst möglich).

18.5.3 Plural nur für Objekte

Der „containerlose" Plural von Werten wie Zahlen hat, wie in Abschnitt 18.2 bemerkt, kaum eine sinnvolle Bedeutung.[9] In NUM ist der Plural daher *nur für Objekte* vorgesehen. NUM erweitert damit den Begriff der *Nullbarkeit* (*nullability*, in NUM durch den Numerus `?` repräsentiert), der in Java und verwandten Sprachen ebenfalls an Referenztypen gebunden ist, auf den Begriff der *Zählbarkeit* (*countability*). Die fallweise Sicherstellung von „not null" (Numerus `!`), die für Sprachen wie Java nach wie vor ein Forschungsthema darstellt (die in NUM jedoch bereits erreicht wird, siehe Steimann 2022), wird damit zu einem Spezialfall des allgemeineren Problems der Behandlung verschiedener Anzahlen von Objekten, eben der Einführung eines Numerus als neben den Typen stehende programmiersprachliche Dimension.

18.6 Nachtrag: Viele Funktionen als eine Funktion

Das, was hier über viele Objekte und ihre Reifizierung zu einem Objekt gesagt wurde, gilt im Prinzip auch für viele Funktionen: Sie lassen sich zu einer Funktion reifizieren. Wie kann das gehen?

Im Lambda-Kalkül sind alle Werte Funktionen. Eine (anonyme) Funktion hat dabei die Form $\langle x \to t \rangle$, wobei die Variable x dem formalen Parameter der Funktion entspricht und t deren Rumpf, der die Form eines Terms hat. Terme können Variablen, Funktionen und Funktionsanwendungen sein:

9 Siehe Steimann 2021 für einige Bemerkungen dazu, warum es nicht sinnvoll ist, Zahlen als Objekte aufzufassen, ja warum eine Unterscheidung von Werten und Objekten nichts ist, dass man beim Entwurf von Programmiersprachen zu verbergen suchen sollte. Das Eingangszitat „Alles ist ein Objekt" wird dadurch aber nicht unbedingt infrage gestellt, nämlich dann nicht, wenn man mit „alles" nur „alle identifizierbaren Dinge", also nur das, was mit einer Identität ausgestattet ist, bezeichnet. Dafür gibt es gute Gründe; nicht zuletzt (und gerade im hier gegebenen Kontext relevant) sind überhaupt nur identifizierbare Dinge auch zählbar, also der Bildung eines Plurals fähig.

t	::=		**Terme**:
		x	Variable
	|	$\langle x \to t\rangle$	anonyme Funktion (Abstraktion, üblicherweise als $\lambda x.t$ notiert)
	|	$t(t)$	Funktionsanwendung (Applikation, üblicherweise als $t\,t$ notiert)

Bei der Auswertung einer Funktionsanwendung $\langle x \to t_1\rangle(t_2)$ werden die Vorkommen von x in t_1 durch t_2 ersetzt; der durch diese (rein syntaktische) Operation erhaltene Term ist das Ergebnis der Funktionsanwendung. So wird beispielsweise die Funktionsanwendung $\langle x \to x\rangle(\langle x \to x\rangle)$ zu $\langle x \to x\rangle$ ausgewertet ($\langle x \to x\rangle$ ist die *identische Funktion*).

Mit dieser stark eingeschränkten Sprache, in der Funktionen auch Daten (Ein- und Ausgaben von Funktionen) sind, lassen sich nun, ähnlich wie mit Mengen – und ihrer Intention nach alternativ dazu! –, komplexere Datenstrukturen darstellen, so auch Listen. Dies kann z.B. per Definition von Paaren als dreistellige (gecurryte) Funktion *pair*, die als Konstruktor von Paaren fungiert, und den beiden einstelligen Zugriffsfunktionen *first* und *second*, die aus einem Paar das erste bzw. zweite Element liefern, geschehen (aus Pierce 2002, §5.2 entnommen):

$$\begin{aligned} \mathit{pair} &:= \langle f \to \langle s \to \langle b \to b(f)(s)\rangle\rangle\rangle \\ \mathit{first} &:= \langle p \to p(\langle f \to \langle s \to f\rangle\rangle)\rangle \\ \mathit{second} &:= \langle p \to p(\langle f \to \langle s \to s\rangle\rangle)\rangle \end{aligned}$$

Hierbei sind, *f*, *s*, *b* und *p* Namen von Variablen, die auf ihre jeweilige Rolle in den sie einführenden Funktionen hindeuten sollen: So steht *f* in der Definition der Funktion (des Konstruktors) *pair* für das erste Element des Paars, *s* für das zweite und *b* für eine Selektorfunktion, die eines aus ihren zwei Argumenten auswählt (ähnlich einem Konditional, deswegen *b* für Boolean); in den Definitionen der Funktionen (Selektoren) *first* und *second* hingegen stehen *p* für ein Paar, auf das der jeweilige Selektor angewendet werden soll (tatsächlich wird hier, der Codierung geschuldet, das Paar *p* auf ein Konditional angewendet), und *f* sowie *s* für das erste und zweite Argument der beiden Konditional-Funktionen, von denen die erste das erste und die zweite das zweite Argument zurückliefern. So ergibt dann für zwei Terme t_1 und t_2 die Auswertung

$$\mathit{first}(\mathit{pair}(t_1)(t_2)) = t_1$$

sowie

$$\mathit{second}(\mathit{pair}(t_1)(t_2)) = t_2\,.$$

Aus Paaren lassen sich nun Listen konstruieren, indem man in der Konstruktion eines Paares aus t_1 und t_2 dafür Sorge trägt, das t_2 selbst ein Paar ist usw. Die Selektoren *first* und *second* entsprechen dann den bekannten Dekonstruktoren für Listen, *head* und *tail*. Da Funktionen Terme sind, lassen sich so viele Funktionen (eine Liste von Funktionen) als eine Funktion darstellen. Q.e.d.[10]

Die Frage im Sinne dieses Beitrags ist nun aber, ob sich ein Plural von Funktionen in einem solchen Kalkül auch ohne Reifizierung zu *einer* Funktion darstellen lässt. Eine Einführung von Bunches würde hier eine beträchtliche Erweiterung darstellen, da das Lambda-Kalkül ja keine Mengen kennt, die sich entsprechend auch nicht zu (strukturlosen)

10 Üblicherweise werden Listen im Lambda-Kalkül als Abstraktionen über zwei Funktionen, *cons* und *nil* genannt, codiert. Dies hat den Vorteil, dass sich Listen (die selbst Funktionen sind) durch Anwendung auf entsprechende Funktionen ohne zusätzlichen Aufwand aggregieren lassen.

Bunches vereinfachen lassen. Eine relativ einfache Möglichkeit besteht darin, die Terme des Lambda-Kalküls durch Strings von Termen zu ersetzen (Steimann 2023); solche Strings erlauben zwar anders als Bunches auch Duplikate und die Reihenfolge ihrer Konstituenten ist signifikant, aber dafür sind Strings rein syntaktische Gebilde (mit der Konkatenation als Verknüpfung), was der syntaktischen Natur des Lambda-Kalküls entgegenkommt (Mengen und ihre Vereinigung lassen sich hingegen nur schlecht syntaktisch fassen).

18.7 Schluss

Programmiersprachen sind künstliche Sprachen, die von Menschen wie Maschinen gleichermaßen gelesen und interpretiert werden können sollen. Im Vergleich zu natürlichen Sprachen sind Programmiersprachen stark reduziert: Ihre Konstrukte sind in der Regel mit Bedacht ausgewählt und so aufeinander abgestimmt, dass sich mit möglichst wenigen Sprachmitteln möglichst viel möglichst ohne Umschweife ausdrücken lässt. Auf der anderen Seite haben natürliche Sprachen einem über Jahrtausende andauernden Evolutionsdruck unterlegen und die Entwicklung von Sprachmitteln wie dem der Differenzierung von Singular und Plural, die man in Programmiersprachen vergeblich sucht, hat sich offenbar bewährt. In dem vorliegenden Beitrag habe ich gezeigt, wie eine solche Differenzierung auch in der Programmierung ohne großen Aufwand möglich ist und wie das die Programmierung verändert. Insbesondere habe ich gezeigt, dass die Einführung des Plurals die Programmierung näher an die Modellierung (in der echte Plurale seit jeher zu den Sprachmitteln gehören) heranrückt.

Literatur

Cantor, G. (1883). „Ueber unendliche, lineare Punktmannichfaltigkeiten“. In: *Mathematische Annalen* 21.4, S. 545–591. DOI: 10.1007/BF01446819. URL: https://doi.org/10.1007/BF01446819.

Eisenberg, A. und Melton, J. (1999). „SQL: 1999, formerly known as SQL 3“. In: *SIGMOD Rec.* 28.1, S. 131–138. DOI: 10.1145/309844.310075. URL: https://doi.org/10.1145/309844.310075.

Hehner, E. C. R. (1981). „Bunch Theory: A Simple Set Theory for Computer Science“. In: *Inf. Process. Lett.* 12.1, S. 26–30. DOI: 10.1016/0020-0190(81)90071-5. URL: https://doi.org/10.1016/0020-0190(81)90071-5.

Hehner, E. C. R. (1985). *The logic of programming*. Prentice Hall International series in computer science. Prentice Hall. ISBN: 978-0-13-539966-8.

Hehner, E. C. R. (1993). *A Practical Theory of Programming*. Texts and Monographs in Computer Science. Springer. ISBN: 978-3-540-94106-4. DOI: 10.1007/978-1-4419-8596-5. URL: https://doi.org/10.1007/978-1-4419-8596-5.

Hußmann, H. (1993). *Nondeterminism in Algebraic Specifications and Algebraic Programs*. Boston/Basel/Berlin: Birkhäuser. URL: http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-4258-3.

Ingalls, D. H. H. (1986). „A Simple Technique for Handling Multiple Polymorphism“. In: *OOPSLA*. ACM, S. 347–349.

Jackson, D. (2006). *Software Abstractions - Logic, Language, and Analysis*. MIT Press. ISBN: 978-0-262-10114-1.

Moore, R. E. (1979). *Methods and applications of interval analysis.* SIAM studies in applied mathematics. SIAM. isbn: 978-0-89871-161-5. doi: 10.1137/1.9781611970906. url: https://doi.org/10.1137/1.9781611970906.

Object Management Group (2011). *Unified Modeling Language 2.4.1.* Specification. http://www.omg.org/spec/UML/2.4.1/. OMG (Object Management Group).

Pierce, B. C. (2002). *Types and programming languages.* MIT Press. isbn: 978-0-262-16209-8.

Steimann, F. (2021). „The kingdoms of objects and values". In: *Onward! 2021: Proceedings of the 2021 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, Virtual Event / Chicago, IL, USA, October 20-22, 2021.* Hrsg. von W. D. Meuter und E. L. A. Baniassad. ACM, S. 125–135. doi: 10.1145/3486607.3486771. url: https://doi.org/10.1145/3486607.3486771.

Steimann, F. (Sep. 2022). „Containerless Plurals: Separating Number from Type in Object-Oriented Programming". In: *ACM Trans. Program. Lang. Syst.* 44.4. issn: 0164-0925. doi: 10.1145/3527635. url: https://doi.org/10.1145/3527635.

Steimann, F. (2023). „A Simply Numbered Lambda Calculus". In: *Eelco Visser Commemorative Symposium.* Hrsg. von R. Lämmel, P. D. Mosses und F. Steimann. Open Access Series in Informatics (OASIcs) 109. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. doi: 10.4230/OASIcs.EVCS.2023.24. url: https://doi.org/10.4230/OASIcs.EVCS.2023.24.

Steimann, F. und Freitag, M. (2022). „The Semantics of Plurals". In: *Proceedings of the 15th ACM SIGPLAN International Conference on Software Language Engineering, SLE 2022, Auckland, New Zealand, December 6-7, 2022.* Hrsg. von B. Fischer, L. Burgueño und W. Cazzola. ACM, S. 36–54. doi: 10.1145/3567512.3567516. url: https://doi.org/10.1145/3567512.3567516.

Chapter 19

Steps Toward Articulating a Work System Perspective that Addresses a Grand Challenge for the IS Discipline

Steven Alter

This chapter is a contribution to an anthology in honor of Ulrich Frank. It describes some of the twists and turns in my research both before and after my 2017 visits to Ulrich's Enterprise Modelling Research Group at the University of Duisburg-Essen. Those visits helped me rethink aspects of personal research goals and directions related to one of the grand challenges for IS research.

19.1 Introduction

My visits to Ulrich's group helped me rethink aspects of personal research goals and directions related to one of the grand challenges for IS research that were identified by a Delphi study involving 143 IS academics 35 years after the first ICIS meeting (Becker et al. 2015). That challenge, 'rethink the theoretical foundations of the IS discipline,' was tied for first of 21 in being viewed as a grand challenge (74.5 % yes, 14.5 % no). It was ranked third of 21 as having an impact on the discipline if it were solved, and was ranked 13 of 21 in terms of time frame, i. e., whether it could be dealt with or solved within 10 years. I think that the work system perspective that I have developed gradually over much of my academic career addresses important parts of that challenge, as will be discussed here.

Upon reading the grand challenge article around four years ago, I was surprised that the topic of theoretical foundations was viewed as more of a grand challenge than 'developing model–driven methods and tools for the full–scale automated generation of implementation–ready IS,' which was seen as a grand challenge by only a minority of the Delphi panel (23.6 % yes, 61.8 % no). In contrast, I saw that challenge as a more difficult and consequential challenge based on my minor involvement in an automatic programming project while I was a PhD candidate. I think that challenge comes closest to categorizing the research performed by Ulrich and his Group.

I was lucky to meet Ulrich Frank around 10 years ago and to discover that we could have valuable conversations even though we pursued different research goals in quite different ways. I was trying to develop ideas and methods that could be useful in understanding systems in organizations while he was trying to develop methods and tools for automated generation of implementation-ready IS. My visit in May 2017 involved teaching a course for UDE's Visiting Scholar Academy, where the instructor teaches a short course on topics that might not otherwise be covered. The second visit was a personal attempt to do something useful during part of the time between the PoEM (Practice of Enterprise Modeling) conference in Leuven, Belgium in late November and the ICIS conference in December in Seoul, South Korea.

Why do you want to do that? A eureka moment in my research efforts occurred while giving a presentation during one of those visits. I was explaining how a specific 'minimalist metamodel' for work systems in organizations might be a step toward a modeling language that might make the work system perspective more useful. Either Ulrich or one of his group members asked a simple question: 'Wwhy do you want to do that?' I do not remember what I mumbled in response, but the question stuck with me. Ulrich and his research group had both appropriate skills and strong rationales for developing *'a method for multi-perspective enterprise modeling (MEMO) and a corresponding (meta-) modeling environment ... The method is based on an elaborate conception of multi-perspective enterprise models and on an extensible language architecture. The language architecture is comprised of a meta modeling language and an extensible set of integrated domain-specific languages (DSML)'* (Frank 2013). As will be discussed here, my goals at that time were modest by comparison. I wondered whether the minimalist metamodel would help me pursue the main purposes of my research, which were really about helping people understand systems in organizations. Since I was aware of the research community's substantial interest in modeling languages, I wondered about the possible relevance of modeling languages to my research. It was not obvious, however, why anyone would need a specialized modeling language for that purpose, especially since most people would never see or learn that language.

The theme of 'Why do you want to do that?' reappears throughout this chapter as it traces some of the twists and turns in my research both before and after my 2017 visits to UDE. It describes progress toward a work system perspective for talking and thinking about systems in organizations that tries to be semi-rigorous and thus is different in nature and scope from a formal language that Ulrich and his group were developing. The attempt to say that the work system perspective is a plausible theoretical foundation for a major part of the IS discipline could bring forth another chorus of 'Why do you want to do that?' because it may seem too ambitious. This chapter is organized around the following steps, which sometimes involve the reappearance of ideas years later whose initial versions suffered from important limitations:

- Providing minor assistance on a major project related to automatic programming,
- Writing textbooks designed to help business professionals think about systems in organizations,
- Seeing information systems as work systems that often served other work systems (that might be information systems on their own right),
- Developing the work system method and work system theory,
- Wondering about how to bring more of a service mindset to work system design,
- Wondering whether work system metamodels might provide a path to UML diagrams,
- Wondering how the realities of workarounds and nonconformance are recognized by process models and BPM,
- Trying to bring richer and more evocative concepts to systems analysis and design,
- Wondering about responsibilities of digital agents,
- Trying to make knowledge about systems easily accessible where it can be used,

- Articulating an expansive work system perspective for talking and thinking about systems in organizations,
- Seeing the work system perspective as a theoretical foundation for a significant part of the IS discipline

One of the reasons why my research seemed to wander from one topic to another is that I never had a research group or PhD or MS students during my 30 years at the University of San Francisco, where I mostly taught MBAs and Executive MBAs. Staying there instead of moving to a more research-oriented university had advantages and disadvantages. The disadvantage was the lack of a research environment and research assistance. I tried to make up for that disadvantage by engaging people in discussions at conferences and universities to get feedback about my ideas and to learn what others were thinking about. The advantages were that I could walk to work in 15 minutes, could write textbooks (which might be dismissed as merely commercial by research universities), could submit articles to venues where they fit, and could pursue topics that I found significant instead of chasing after the latest hot topics in IS. Other reasons for wandering from topic to topic included my lack of training in formal research methods, behavioral science, and computer science and my eight years of experience in a manufacturing software start-up where we used analysis and modeling capabilities extensively but made almost no use of the ideas from the academic IS literature. A key learning from the industry experience was that real world applications of knowledge about systems might not be supported well by the types of thought processes, research outputs, and publications that are prized in academia. Said in a different way, I sometimes felt as though I was partially ruined as an academic researcher by absorbing values from practice related to many of the topics that we purport to study.

19.2 Providing Minor Assistance on a Project Related to Automatic Programming

I did some work for William Martin’s Automatic Programming Group at MIT’s Laboratory for Computer Science even though my PhD thesis at the MIT Sloan School of Management was one of the first studies of decision support systems, which was a new idea at that time. The idea of automatic programming goes back to early developments in computer science and initial ideas about artificial intelligence in the 1950s. The Automatic Programming Group pursued automatic programming as a process of transforming specifications through several levels of formal languages to convert a high level specification of relatively general requirements into software in the target programming language. This was somewhat like an early attempt at model-driven development.

My work in the Automatic Programming Group is noteworthy here only because it primed my curiosity about the link between designing requirements and implementing software systems. My contribution to Martin’s group was a cost estimator that was designed to play a role in determining an efficient implementation of dataset-computation pairs. I published ‘A Model for Automating File and Program Design in Business Application Systems’ in the Communications of the ACM in 1979, but I have no idea whether the cost estimator was ever used and feel quite certain that it would not be relevant with today’s vastly more powerful hardware and software. I served largely as a technical writer in producing parts of a technical report called ‘A Very High Level Language for Business Data Processing’ (Ruth et al. 1981).

It is fortunate that I did not pursue the automatic programming work because the DSS research was more consistent with my interests and capabilities. I converted my PhD thesis into one of the first books on DSS, and produced articles that some people remembered when I returned to academia years later, in some instances with people looking at me as though they were surprised I was still alive.

19.3 Writing Textbooks Designed to Help Business Professionals Think about Systems in Organizations

After eight years at a successful manufacturing software start-up I took a faculty position at the University of San Francisco, which (at that time) was mostly a teaching institution with minimal expectations for faculty research. Nonetheless, I started thinking about research possibilities because I assumed that professors were supposed to do research. My experience in the software company told me that the type of DSS research that I had done probably would not have much influence on practice. As I thought about my experience in the software company, I remembered occasions when I and my staff supported sales and customer service presentations that focused largely on the conceptual details and operation of our software without providing substantial guidance about how our software could help our customers operate more efficiently and effectively.

A eureka moment occurred when a sales representative from a large publishing company asked for my opinion of the IS textbooks that I was using. I replied that the available introductory IS textbooks were generally OK but would not have been useful to the start-up's staff or its customers. That seemed like a worthwhile opportunity. I decided to try to write a textbook that would have been helpful to both groups. Four editions (1992, 1996, 1999, 2002) were published before a publishing merger made continuation of the series redundant for the merged company. Those textbooks for undergraduates and MBAs (not IS experts) proved to be a starting point for most of my research from that time until today.

At that time, I believed that most IS textbooks and much IS research of that time focused too much on the mechanics of IT and its direct uses and too little on achieving business results more reliably. To look for system-related ideas that might help, I spent two days at the US Library of Congress around 1990, when the retrieval technology involved slow, hand-delivered fulfillment of hand-written book requests. After finding little of value at the Library of Congress I assumed that the IS community needed a new or different paradigm for understanding IT-enabled systems. Mirroring a statement by the economist Ronald Coase in Coase (1988, p. 4) that was cited in Hovorka et al. (2019, pp. 1364–1365), I decided to 'make it up myself.'

The Preface of the textbook's second edition mentioned trying to generate 'what I think is a new way to synthesize and explain ideas about information systems.' That Preface explained that the new approach emerged from 22 book tour presentations [for the first edition] that tried to address the imagined challenge of producing a one-hour presentation that would say something genuinely useful to business professionals from various organizations who were about to go into meetings about information systems of various kinds that have various types of issues that I would not know in advance. What could I say that would be useful to them? The Preface continued, 'The goal of finding something valuable and general to say about information systems in one hour might sound crazy. However, if I could identify and express the basic ideas, then ... this text [could] achieve its goals: helping students become able to recognize, described, analyze, and design information systems from a business professional's viewpoint.' Most of my research in subsequent years was devoted to various aspects of a related research question: 'What is an organized, useful, and teachable method that business professionals could use for understanding, analyzing, designing, and evaluating IT-enabled systems in their organizations?'

Today's academic publishing expectations and standards likely would find that out-of-the-box research approach inadequately justified, poorly controlled, and at a least a bit unsavory due to its commercial associations. Those issues were not on my radar screen, especially after working for years in a firm that met with customers routinely as part of its efforts to build and sell application software that addressed customer needs. Being a faculty member at a business school where journal publications were not of great importance (at that time) provided space for working on the research question even without an accepted research paradigm and outside of established discursive rules (Hassan et al. 2022). That effort would have been more difficult or possibly even discouraged at universities that saw theory-based journal publications as the coin of the realm.

Termination of the textbook series eliminated an income stream but had two beneficial effects. Now I would not face the frustration of trying to include the latest technology in each new edition of the textbook only to find that some of that coverage was already outdated when the textbook was finally published many months later. More important, ending the textbook revisions meant that I had time to do research.

19.4 Developing the Work System Method and Work System Theory

The core ideas in the textbooks were developed further as a systems analysis method for business professionals called the work ystem method (WSM) (Alter 2006; Alter 2008). The ideas underlying WSM were formalized as work system theory (WST), and subsequent developments related to service systems, workarounds, design principles, and other topics that are extensions of WST (Alter 2013). WST applies equally to work systems in general and to ISs, projects, totally automated work systems, and other special cases of WS. WST's three components are the definition of work system, the work system framework, and the work system life cycle model.

Work systems. Work system is a natural unit of analysis for thinking about purposeful systems in organizations. A work system is a system in which human participants and/or machines perform work (processes and activities) using information, technology, and other resources to produce specific product/services for internal and/or external customers or for themselves. The work in work systems may be structured to varying degrees, e. g., unstructured (designing a unique advertisement), semi-structured (performing typical medical diagnosis), workflows (processing invoice payments), or highly structured (manufacturing semiconductors or pharmaceuticals).

The most important distinction in describing special cases of WS is the difference between a sociotechnical WS in which human participants perform some of the activities vs. totally automated WS where all activities are performed by machines and where the slot for participants is blank. Information systems, projects, service systems, self-service systems, and some supply chains (interorganizational WSs) are important special cases. For example, software development projects and other projects are WSs designed to produce specific product/services and then go out of existence.

Work system method. Most of the ideas in WSM were developed with the help of MBA and Executive MBA students who used successive versions of a work system-oriented analysis outline to produce over 700 management briefings between 2003 and 2017 (e. g. Truex et al. 2010). Different courses used different versions of WSM outlines, but the fundamental approach was always organized around the following steps.

1. Identify the smallest work system that has the problem or opportunity at hand.
2. Summarize the 'as-is' work system using a work system snapshot, a stylized one-page summary that that identifies six work system elements
3. Evaluate the work system's operation using perceived strengths and weaknesses, metrics, key incidents, social relations, and other factors.
4. Drill down further as necessary using any relevant ideas, including both WSM tools, e. g., Alter (2006), Alter (2020a) and Bork and Alter (2020), plus other tools and approaches that are not specifically related to the WSP, e. g., design thinking and Six Sigma methods.
5. Propose changes by producing a work system snapshot of a proposed 'to be' work system that should perform well.
6. Describe likely performance improvements and explain why the effort of creating the new work system or making the proposed changes seems justified.

Work System Framework. A central challenge in producing a usable version of WSM was developing a general model that outlined the elements of a basic understanding of a work system. Eventually that was called the work system framework (Figure 19.1). Its nine elements are the elements of a basic understanding of a WS's form, function, and environment during a period when it is stable enough to retain its identity even though incremental changes may occur, such as minor personnel substitutions or technology upgrades. Processes and activities, participants, information, and technologies are completely within the WS. Customers and product/services may be partially inside and partially outside because customers often participate in activities within a WS and because product/services take shape within a WS. Environment, infrastructure, and strategies are outside of the WS even though they have direct effects within a WS and may be affected by major changes in significant WSs.

It is easy to take the work system framework for granted, but Figure 19.1 is actually the eighth published version. For example, the first version from 1992 had only five elements: goals, work practices, information, people, and information technology. The second edition added customers and products but treated goals as properties of a work system as a whole and each of its six elements. The most recent version is Figure 19.1, which has nine elements. Customers refers to people or organizations that receive product/services produced by a WS. This includes internal and external customers. The term product/services replaced products and services to bypass controversies about special characteristics of products vs. services. Processes and activities replaced work practices because MBA students seemed averse to distinguishing between idealized processes and actual work practices and to recognize that activities in some WSs are less like formal sequences of steps and more like

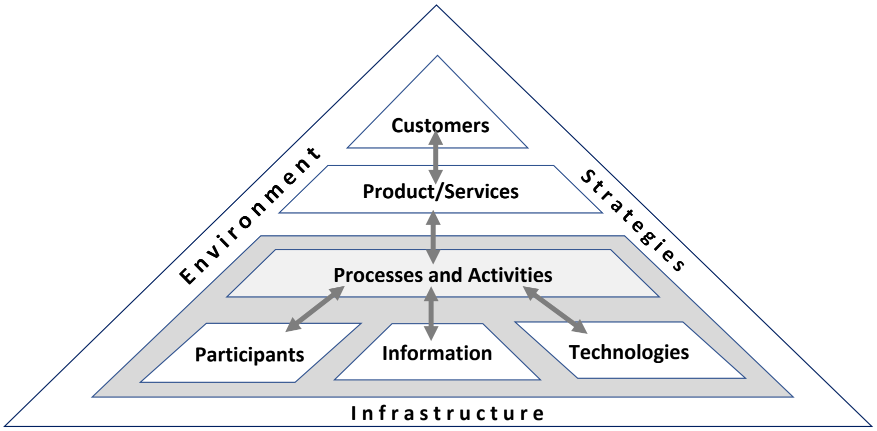


Figure 19.1: The work system framework

sets of activities that may occur in different orders, especially in knowledge-intensive work. Technologies replaced IT and technology because multiple technologies are relevant to many WSs. Infrastructure refers to human, informational, and technical resources that are viewed as shared by multiple WSs instead of being associated primarily with one WS. An example of human infrastructure is an IT group that can be viewed as a resource used by multiple WSs. Strategies had been added in 2002 because an editor of the practitioner journal CIO Insight said that no CIO would pay attention if the framework did not contain the idea of strategy – actually a useful suggestion. Also, the bi-directional arrows inside the framework represent desired alignment and do not represent flows.

Work System Life Cycle Model (WSLC). ISs and other WSs evolve through a combination of planned change through projects and unplanned change through adaptations and workarounds (Figure 19.2). WSLC phases (initiation, development, implementation, and operation and maintenance) may be performed in different ways. Both planned and unplanned changes often affect multiple WS elements, not just technologies. Typical activities and responsibilities (e. g., designing, debugging, training, etc.) associated with specific phases apply for waterfall, agile, prototyping, use of off-the-shelf applications, and shadow IT, even when several phases overlap or iterate. The WSLC's four idealized phases (and related sub-phases) express a waterfall-like approach to identifying things that should happen as a WS evolves iteratively. Many WSLC topics remain valid when agile approaches are used for developing software, such as the importance of WS changes rather than just software development, evolution over time rather than one-time projects, the simultaneous importance of planned and unplanned change, and the relevance of key activities and responsibilities within each phase.

Work System Theory. Around 2010, I was increasingly frustrated by the reaction of journal reviewers to the work system method. It was reasonably well defined, made sense, was used in many introductory courses, and seemed quite useful, but typically was viewed as trivial because it was not based on a theory. Consistent with the theory fixation discussed by Hirschheim (2019, p. 1340), the lack of underlying theory often seemed more important

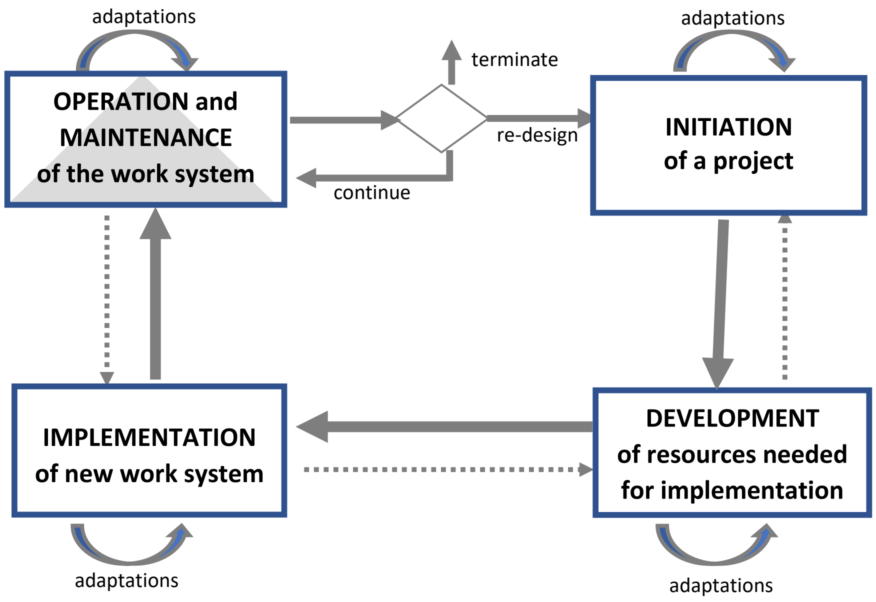

adaptations
OPERATION and MAINTENANCE of the work system
terminate
re-design
continue
adaptations
INITIATION of a project
IMPLEMENTATION of new work system
DEVELOPMENT of resources needed for implementation
adaptations
adaptations

Figure 19.2: The work system life cycle model (WSLC)

to the reviewers than the potential value of ideas related to significant pedagogical and practical problems. With substantial help and insightful criticism from the SE and reviewers I managed to produce an article on work system theory (Alter 2013) that identified WST as the theory underlying WSM. It articulated three components of WST that were mentioned above, the definition of WS, the work system framework, and the WSLC.

The scientifically questionable attempt to produce a theory for the sake of being able to say that WSM was based on a theory actually led to many benefits, especially being able to explain in 10–15 minutes how WST could form the basis of a large variety of WST extensions and applications. Another benefit is that WST could serve as the core of a broader work system perspective (WSP) that is explained later.

19.5 Seeing Information Systems as Work Systems that Often Serve Other Work Systems (that Might be Information Systems on Their Own Right)

An important conceptual breakthrough in developing WST was the realization that ISs are WSs and that most ISs exist to support other WSs that may be ISs on their own right. A set of coincidences led to that conceptual breakthrough.

An IFIP working conference on ‘organizational semiotics’ was held in Montreal just before the 2001 AMCIS meeting that I planned to attend in Boston. I was intrigued about the idea of organizational semiotics but was skeptical about what it might mean. I learned about Stamper’s six level semiotic ladder (Stamper 2001) but doubted that it would have helped the customers or staff of the software startup where I had worked. At the conference I encountered the word *ontology* for the first time, and I wondered what that might mean, as demonstrated by the strange look that a presenter gave me when I asked whether the presenter's project called for producing an ontology.

I was too distant from ontologies and semiotics to learn a lot at that conference, but I kept the conference T-shirt, which contained a bunch of symbols that were meant to represent something about the nature of semiotics. A year later I was in line at an airport in Hawaii just after the HICSS conference when Susan Sherer, who had also attended HICSS, asked me what the T-shirt was about. Eventually we wrote a paper about factors related to IS risk based on literature available at that time (Sherer and Alter 2004). We used the elements of the work system framework to organize risk factors from 46 articles. Over half of the risk factors (134 of 228) were relevant to work systems in general even though many were presented as though they were related to a more limited topic, such as a particular type of information system or project. More generally, risk factors initially associated with one type of system or situation (e. g., ERP implementation) are often equally relevant at other levels (e. g., information systems, projects, or work systems in general).

That finding is consistent with the idea that IS and project are just special cases of WS. That matters because most of the concepts and generalizations that are relevant to ISs in general are also relevant to WSs in general. (You can test that assumption by trying to identify nontrivial concepts and/or generalizations that apply to all ISs but not all WSs.)

Seeing IS as a special case of WS led to a definition of IS as a WS most of whose activities are devoted to capturing, transmitting, storing, retrieving, deleting, manipulating, and/or displaying information. This definition differs from 20 previous definitions in (Alter 2008) and was one of 34 definitions of IS noted in Boell and Cecez-Kecmanovic (2015). An example is a sociotechnical accounting IS in which accountants decide how specific transactions and assets will be handled for tax purposes and then produce monthly or yearend financial statements. This is an IS because its activities are devoted to processing information. It is also supported by a totally automated IS that performs calculations and generates reports. In both cases an IS that is an integral part of another WS cannot be analyzed, designed, or improved thoughtfully without considering how changes in the IS affect that WS.

While seeing IS as a special case of WS seems to me obvious, sufficiently rigorous, and generative in a number of ways, a prestigious journal's 2022 desk reject of an article that built on that idea showed that it is not at all obvious to other researchers. The desk reject dismissed the idea that ISs are WSs by saying 'the difference between IS and IT is well known. There are four components in IS (i. e., task, people, structure, and technology) while there are three components in IT (i. e., HW, SW, and data).' That seems to say that information is not part of an IS but is part of the technology that is part of an IS. Whoever wrote that part of the review should talk to a practitioner concerned with erroneous information in a mission-critical IS.

19.6 Wondering about How To Bring More of a Service Mindset to Work System Design

Common omissions from management briefings by MBA and Executive MBA students concerning problematic IT-enabled work systems in their own organizations led me to conclude that they focused mostly on the internal efficiency of work systems and not enough on effectiveness in satisfying customer needs and expectations. I thought it might be possible to articulate ideas that would instill more of a service mindset into work system design.

Around that time I met Jim Spohrer, a lab director at IBM who acted as chief evangelist for IBM's Service Science, Management, and Engineering (SSME) initiative that started in 2002, possibly due to recognition that over half of IBM's revenues were generated by services. Initially I thought that work system analysis and design might express more of a service mindset if it used some of the ideas from the SSME initiative. Unfortunately, many of the original SSME ideas did not ring true to me, e. g., that individual people, families, companies, and cities all might be viewed as service systems and that services are co-created consistent with service dominant logic (Vargo and Lusch 2016), which I see as a high level rationale for economic exchange rather than an operational theory about systems. Consistent with that viewpoint, early proponents of service science preferred a complex view of a *'service system as an open system (1) capable of improving the state of another system through sharing or applying its resources (i. e., the other system sees the interaction as having value), and (2) capable of improving its own state by acquiring external resources (i. e., the system itself sees value in its interaction with other systems)'* (Spohrer et al. 2008).

Despite that difference of viewpoint, I published a 2008 article in IBM Systems Journal arguing that a fundamental approach for understanding service systems includes the work system framework (Figure 19.1), work system life cycle model (Figure 19.2) and a new service value chain framework (Figure 19.3). The service value chain framework identifies typical categories of service activities and responsibilities. Every part of this framework is important for many, but not all services and service systems. The entire service value chain for a service can be viewed and analyzed as a single work system; alternatively different subsystems in Figure 19.3 (such as provider preparation or negotiation of commitments) or even specific interactions might be analyzed as separate work systems. Key points in Figure 19.3 include:

- The bilateral form of the service value chain framework is based on the widely accepted observation that services often are coproduced by service providers and service consumers (Alter 2010).
- Co-production implies attention to separate responsibilities of providers and customers (regardless of whether the customers are internal or external).
- The two vertical lines of visibility separate front stage (customer activities and resources that are visible to providers, and vice versa) versus backstage (relevant customer activities and resources that are not visible to providers, and vice versa)
- Value creation for both customers and providers may occur across the entire service value chain (e. g., the value of efficient negotiations and clear service requests for both customers and providers)
- Service encounters may occur across the entire service value chain and may have important impacts on value capture.
- Service stages of setup, service request, service fulfillment, and follow-up may be viewed as simple steps or may be subdivided into more detailed steps.

For one who first learned about the idea of ontology at an organizational semiotics conference in 2001 (mentioned earlier), it was gratifying that researchers who develop and use ontologies actually took the service value chain framework seriously as a starting point or point of comparison for their typically more complex work on service and service ontologies, i. e., for

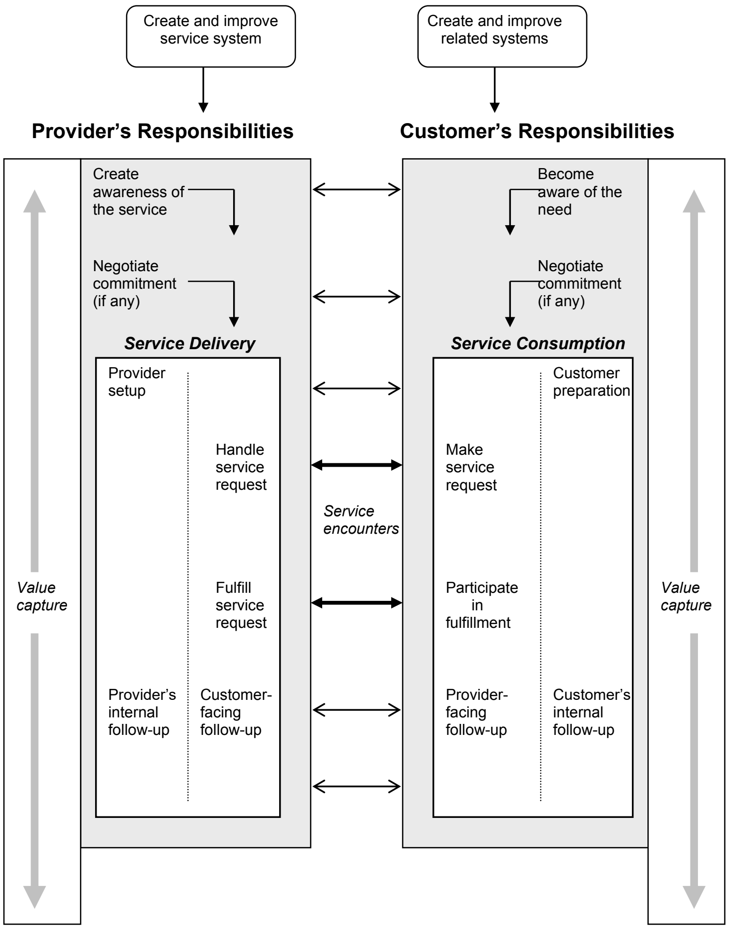

Create and improve service system
Create and improve related systems
Provider's Responsibilities
Customer's Responsibilities
Create awareness of the service
Become aware of the need
Negotiate commitment (if any)
Negotiate commitment (if any)
Service Delivery
Service Consumption
Provider setup
Customer preparation
Handle service request
Make service request
Service encounters
Fulfill service request
Participate in fulfillment
Provider's internal follow-up
Customer-facing follow-up
Provider-facing follow-up
Customer's internal follow-up
Value capture
Value capture

Figure 19.3: Service Value Chain Framework (Alter 2010)

example, Lemey and Poels (2011), Ferrario et al. (2012) and Oberle et al. (2013). Subsequent efforts proposed using 'service responsibility tables' in systems analysis (Tan et al. 2011) and using the general idea of responsibilities as the basis of an agent-responsibility framework (Figure 19.6) that will be described later.

19.7 Wondering Whether Work System Metamodels Might Provide a Path to UML Diagrams

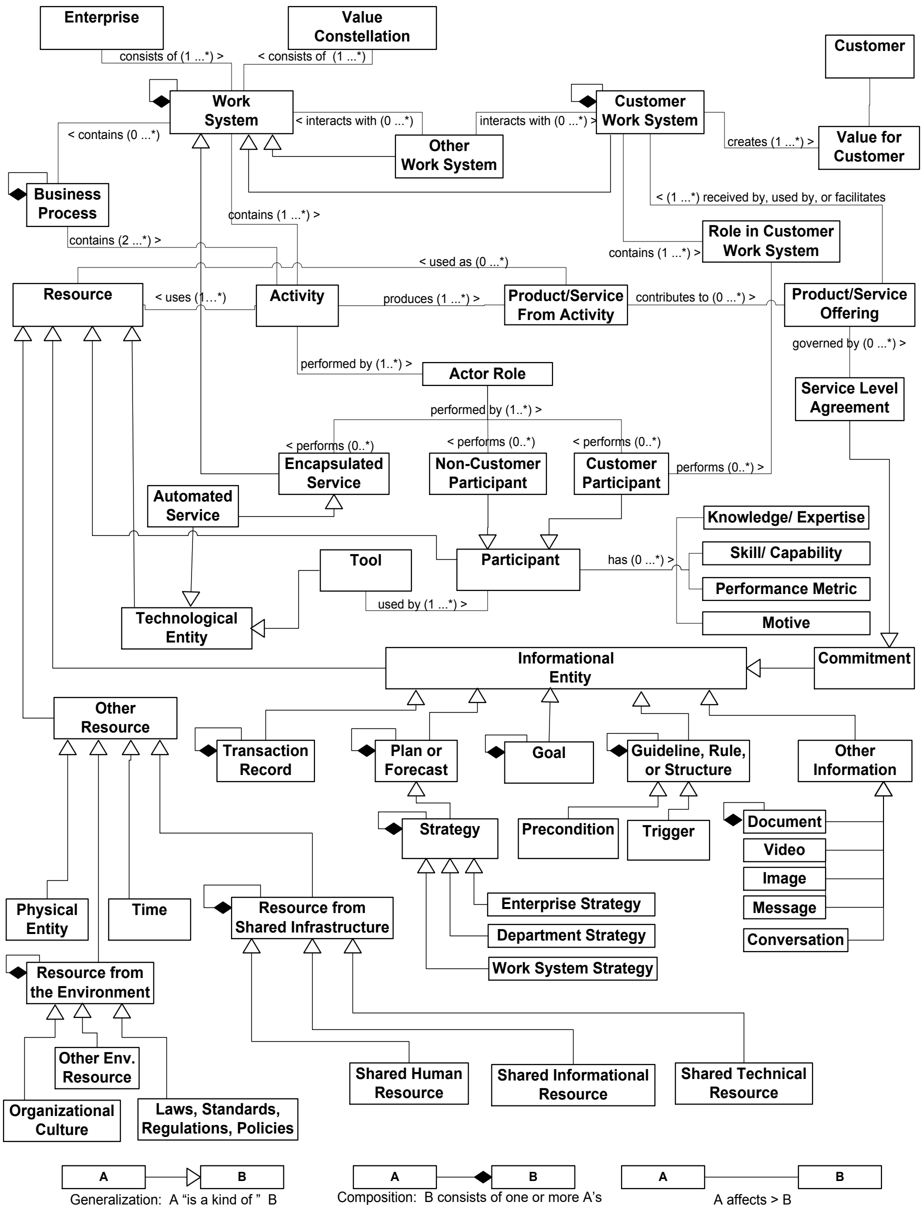
Working on the periphery of MIT's Automatic Programming Group while a PhD candidate (mentioned earlier) had primed me to consider the possibility that specifications in a higher level language could be translated automatically into executable programs. I had never used or taught UML, but I thought it might be worthwhile to imagine ways to bridge parts of the gap between UML and some type of semi-formal work system description that could be produced by MBA and EMBA students. The work system framework had been developed to help business professionals describe and understand systems in organizations. It was not designed to document executable logic that might guide work system operation. Perhaps it could be re-specified in a way that would form the beginning of a path toward automatic execution.

A necessary step in that direction was to develop a rationale for work system descriptions that were more structured and more detailed than descriptions organized around the nine elements of the work system framework (Figure 19.1). Figure 19.4 is the sixth of a series of metamodels that tried to address that limitation. That metamodel from Alter (2016) reinterprets each element of the work system framework in a more detailed way. For example, information becomes informational entity (e. g., transaction record, forecast, goal, and so on), technology is divided into tools and automated services, activities are performed by three types of actors, and so on. The comment at the bottom of Figure 19.4 notes that many properties of each entity type such as goals, performance indicators, and relevant generalizations are hidden in the one-page representation. Outputs of activities are called product/service from activity. These may serve as resources for other activities and may contribute to product/service offerings for the work system's customers. Customer work system is included in the metamodel to establish links between provider resources and value creation, thereby providing an operational interpretation of resources and value creation (inspired by the service value chain framework in Figure 19.3 and the possibility of interpreting aspects of service dominant logic in an operational way).

The metamodel in Figure 19.4 has many conceptual and practical shortcomings. Somehow it manages to be both incomplete and too complicated. Is incomplete because it fails to represent many possible configurations of work systems, especially situations in which work systems overlap or have mutual dependencies. It is too complicated because its multiplicity of entity types and relationships make it difficult to explain to someone who is not attuned to understanding complex models.

On the other hand, it proved useful in proposing a front end to object-oriented analysis and design (OOAD). Alter and Bolloju (2016) showed how two tools from WSM could be used in a step that precedes creation of use case diagrams and other UML artifacts. Those two tools are work system snapshots which summarize a work system on one page using six elements of the work system framework and a table that identifies resources used, triggers, and product/services produced for each activity. A subsequent controlled experiment conducted with 165 students in an undergraduate software engineering course in India indicated that preparation of work system snapshots resulted in a significant reduction in invalid user stories and an increase in valid user stories in the product backlog (Bolloju et al. 2017).

Around that time, I became increasingly dissatisfied with the idea of producing additional metamodel versions that would be even more complicated. The 2017 presentation to Ulrich's Information and Enterprise Modelling Research Group that was mentioned in the introduction was an attempt to elicit reactions to possible directions for making the work system perspective more useful. Those possible directions included better templates, clearer semantics, formal syntax, explicit notation, and links to complementary viewpoints. One part of the presentation involved a thought experiment of treating work system theory as the basis of a modeling language based on a minimalist metamodel that stripped as much complexity as possible from the metamodel in Figure 19.4. The minimalist metamodel would contain only nine entity types: enterprise, work system, activity, human resource, informational resource, technological resource, other resource, goal, and characteristic.



Note: Many elements in the conceptual model have goals, attributes, performance indicators, and related principles, patterns, and generalizations that do not fit into a one page representation, and that must be included in more detailed explanations.

Figure 19.4: Sixth version of a work system metamodel that re-specified elements of the work system framework

Its four types of relationships would be contains, uses, produces, and has attribute. The metamodel would say that an enterprise contains work systems; work systems contain activities; activities produce and use resources; and all entities have goals and characteristics. Since many researchers were talking about modeling languages, and since graphical aspects of some modeling languages were slightly reminiscent of the organizational semiotics T-shirt whose mysteries led to a project on IS risk (mentioned earlier), my presentation suggested that a graphic artist's major revision of the icons in Figure 19.5 might denote enterprise, work system, activity, and so on in a possible modeling language. That is when someone asked 'why do you want to do that?'

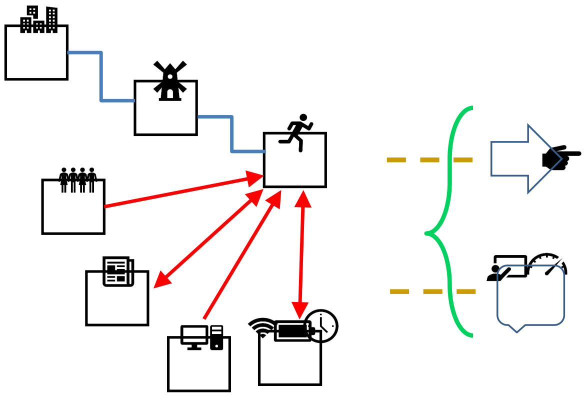


Figure 19.5: Rough indication of modeling language discussed in a thought experiment

On further reflection it became clear to me that I did not really want to develop a modeling language because that would be too restrictive. The long-term goal from my textbook writing days still applied. My real goal was to develop a set of ideas that would be rigorous enough to be genuinely useful in helping people think about systems but would be flexible enough to allow them to think about systems in ways that were relevant to their particular concerns.

Follow-on research with Dominik Bork tried to provide a more rigorous approach to pursuing that goal by suggesting that alternative work system metamodels might serve different stakeholders who had different purposes in thinking about a situation involving a work system. For example, some stakeholders might want to discuss work system capabilities without looking deeply at details; others might want an overview of how a work system operates (the level of the work system snapshots mentioned earlier); others might want to think about how specific resources are used by specific activities within the work system; yet others might want to develop detailed models of how the work system operates (e. g., BPMN models). Bork and Alter (2020) used a sequence of different work system metamodels to illustrate that using a single overarching modeling method metaphor (Ferstl and Sinz 2013) might be an approach for satisfying four requirements for more flexible modeling methods: (1) respect stakeholder diversity related to knowledge, beliefs, and roles, (2) permit different modeling techniques for different stakeholder purposes related to the same situation, (3) accommodate different modeling languages based on different metamodels, (4) recognize that a model might or might not use diagrams with rigorously defined notation and syntax or might use such diagrams for some purposes but not for others.

19.8 Wondering about Conflicts between Process Models and BPM versus Workarounds and Nonconformance

A BPM tutorial at a conference over a decade ago led to an important realization about processes and BPM. The presenter described BPM (as it existed at that time) as software that tracked steps in a process. I raised my hand and said that tracking steps in a process is not sufficient for managing high precision processes such as semiconductor manufacturing, where management requires information about details of the work that actually occurred. In essence, the presenter replied that BPM tracks steps, but not details of the activities that are performed within the steps. I raised my hand again and asked why this was called business process management when it clearly fell short of collecting information needed to manage many processes. The presenter moved on to other topics.

Thinking further, I wondered how process modeling and BPM could handle the common phenomena of workarounds and nonconformance. I wondered whether it was meaningful to model a process as a specific sequence of defined steps with full knowledge that work system participants might not do the work as prescribed. I also wondered why it made sense to do process mining if the process was already known (based on a misunderstanding of process mining that I think I have overcome).

In combination, those concerns led to a 2014 paper on a theory of workarounds, a 2015 paper on workaround design system (identify foreseeable workarounds, evaluate them as beneficial or harmful, and train staff accordingly), a 2015 paper on beneficial noncompliance and detrimental compliance, and a paper (Alter and Recker 2017) that tries to superimpose ideas about work systems and workarounds on top of many BPM research use cases identified by van der Aalst (2013). The theory of workarounds paper has proved helpful for follow-on research by others; the workaround design system paper probably has had no impact even though it might be an interesting topic for future research by a tool builder; the beneficial noncompliance and detrimental compliance paper raises interesting questions that are not typically raised in that way, and the paper about BPM research use cases might contain useful ideas.

Totally independent of the paper about superimposing work system ideas on BPM research use cases, I am pleased that some members of the BPM community have used work system ideas at least to some extent. A paper on business process redesign (Reijers and Mansar 2005) proposed a BPR framework (p. 293) that was based partly on an early version of the work system framework (Figure 19.1). Gross et al. (2021) on p. 35 describes a business process design space whose six layers overlap substantially with the six core elements of the work system framework. Both include customer, product/service, business process, information, and technology, but they differ in the sixth element, which is organization in Gross et al. (2021) and is participants in Figure 19.1, because the execution of sociotechnical work systems relies directly on participants.

19.9 Trying to Bring Richer and More Evocative Concepts to Systems Analysis and Design

The previously mentioned WSM assignments for MBA and EMBA students generally met their main educational goals, but many recommendations seemed mundane (e. g., collect currently uncollected data or train work system participants whose training is inadequate).

Based on that observation, I began to wonder whether an appropriately packaged set of system-related metaphors might provide richer and more evocative concepts that could help in producing more interesting or insightful recommendations. Several attempts to pursue that idea were inspired by earlier publications that applied metaphors for understanding complex management or system topics (e. g., Morgan 1986; Kendall and Kendall 1993; Oates and Fitzgerald 2007), but ran into roadblocks due to inadequate framing of the problem as the identification of subsystems rather than broader themes (e. g., making decisions, communicating, coordinating) that might be dispersed throughout different parts of a work system. A roundabout path (Alter 2021b, pp. 342–344) eventually led to the idea of facets of work which in turn became part of an agent-responsibility framework.

Facets of work are aspects of work that can be observed or analyzed, such as making decisions, communicating, processing information, and coordinating. The notion of 'facet' is an analogy to how a cut diamond consists of a single thing with many facets that can be observed or analyzed. All 18 of the facets of work in Table 19.1 apply to both sociotechnical and totally automated WSs, are associated with specific concepts, bring evaluation criteria and design trade-offs, have sub-facets, and bring open-ended questions for starting conversations. Some facets overlap in many situations (e. g., making decisions and communication).

Making decisions	Communicating	Providing information
Representing reality	Learning	Coordinating
Performing physical work	Providing service	Applying knowledge
Planning	Improvising	Performing support work
Creating value	Thinking	Controlling execution
Processing information	Interacting socially	Maintaining security

Table 19.1: 18 facets of work (Alter 2021b)

The selection of the 18 facets was the result of an iterative design process that might have led to some other set of facets, perhaps 14 or 27. Determination of whether or not to include a type of activity as one of the facets of work in Table 19.1 was based on the extent to which that type of activity was associated with a set of concepts, evaluation criteria, design trade-offs, sub-facets, and open-ended questions that could be useful in analysis and design. The potential contribution of facets of work is that the facets of work provide a way to be specific about requirements for many types of capabilities that might be overlooked otherwise. There is no assumption that the facets of work should be independent. To the contrary, the facet making decisions often involves other facets such as communicating, learning, and processing information. The main point is that each facet can be viewed as part of lens for thinking about where and how WSs might use information systems or digital agents.

19.10 Wondering about Responsibilities of Digital Agents

The service value chain framework in Figure 19.3 is basically about providers providing services for customers. What if the provider was a digital agent (or more broadly, an information system) and the customer was a work system? In essence the digital agent would have responsibilities in relation to the work system. Those responsibilities could be described in terms of a two dimension agent-responsibility (AR) framework shown in an abbreviated form in Figure 19.6 (Alter 2022c). The horizontal dimension of an AR framework is spectrum of roles that goes from the lowest to the highest direct involvement of a digital agent in the execution of a WS's activities. The vertical dimension is different facets of work. The reduced version of the AR framework in Figure 19.6 shows only six or the 18 facets. Presenting the reduced version makes it easier to visualize what the AR framework means.

<<< Facet of work >>>	Monitoring the work system	Providing information	Providing capabilities	Controlling activities	Coproducing activities	Executing activities
Making decisions						
Communicating						
Processing information						
Coordinating						
Creating value						
Maintaining security						
	<<<<<<< Spectrum of roles and responsibilities >>>>>>>					

Figure 19.6: Agent responsibility framework with six roles and six facets of work

Combining the AR framework's two dimensions leads to pinpointing design issues, e. g., the extent to which a digital agent should have responsibilities such as monitoring work, providing capabilities used in making decisions, or performing activities automatically. Different digital agent roles might be applied to any of the 18 facets of work, the six shown in Figure 19.6 and the other 12 shown in Table 19.1.

Uses of the AR framework do not rely on considering all combinations of roles and facets. Also, other roles and facets might be especially relevant to specific situations. Focusing on different facets of work could encourage designers or managers to wonder about the benefits of enhancing specific facets of work in a specific WS. Similarly, the spectrum of roles in the horizontal dimension encourages considering possible roles/responsibilities that digital agents might perform. It is unnecessary to consider all or even many of the 36 combinations of 6 facets of work and 6 types of roles/responsibilities (or 108 combinations of 18 facets of work presented in Table 19.1 and 6 types of roles/responsibilities). Only combinations that are important for a specific WS should be considered.

19.11 Searching for Ways to Make Knowledge about Systems Easily Accessible

A central theme in much of my work is that knowledge should be accessible and understandable by people who need to use it. That theme was a key motivator for writing textbooks, and reappeared a number of times as I started thinking about how to construct an IS body of knowledge (ISBOK) that could be accessed conveniently without using a textbook. A paper from 2005 proposed a conceptual architecture for an imagined language called Sysperanto, a play on Esperanto which is an artificial language that has received some use. A different version of a similar idea appeared as a proposed ‘knowledge cube’ in a 2012 ECIS paper. The dimensions of that cube included (1) nine elements of the work system framework, (2) different types of knowledge, (3) a loose hierarchy of special cases of work system that was included because special cases would inherit knowledge from more general cases, e. g., software development project would inherit knowledge that applies to project in general, which would inherit knowledge from work system in general. A 2013 CAIS paper proposed the idea of an ‘interpretary’ for the IS discipline, i. e., a compendium of interpretations of IS concepts and methods from different theoretical perspectives. Such an interpretary would be a step toward an organized body of knowledge for the IS discipline. Its proposed form was a two-dimensional concept interpretation matrix whose cells contain interpretations of specific concepts from specific theoretical perspectives.

A starting point for a potentially more fruitful approach to IS knowledge was a contribution to a debate in CAIS whether IS is a science (McBride 2018; Kautz 2018). The question of whether IS is a science led to thinking about what science is. My answer was that science is the creation, evaluation, accumulation, dissemination, synthesis, and prioritization of knowledge objects (KOs), including the reevaluation, improvement, or replacement of existing KOs by other KOs that are more effective for understanding the relevant domain. Just that definition is interesting in a way because I have not seen much interest in identifying IS research conclusions that are now obsolete. Figure 19.7 shows a third version of a taxonomy of KOs whose initial versions appeared in Alter (2020b) and Alter (2021a). Ideally, research results and other knowledge of all types covered by that taxonomy should be accumulated and organized in a way that makes that knowledge as accessible and useful as possible for requirements engineering, IS engineering, software development, and other relevant activities.

Figure 19.7 identifies five categories of concepts (concepts related to things, activities, characteristics, metrics, and phenomena) and indicates that all five categories of concepts are KOs that are relevant to other broad categories of KOs that include data, interpretations, generalizations, and methods. Thus, individual concepts themselves and data, interpretations, generalizations, and methods that are defined using concepts are all viewed as KOs. Alter (2022b) illustrates how combining that taxonomy with the idea of inheritance from work systems in general to special cases such as information systems and projects might provide a way to organize the KOs in an IS body of knowledge (an ISBoK). That paper also suggests that the use of tools based on knowledge graphs might make that knowledge accessible. Preliminary exploration of the possible practicality of that idea could try to apply knowledge graph techniques to limited sets of KOs that have already been compiled.

19.12 The Work System Perspective as a Theoretical Foundation for a Significant Part of the IS Discipline

The lack of a theoretical foundation and organized body of knowledge for IS have been discussed for at least four decades (e. g., Keen 1980; Hirschheim and Klein 2003; Hassan and Mathiassen 2018). As noted in the introduction, ‘rethink the theoretical foundations of the IS discipline’ was tied for first of 21 identified challenges as a grand challenge for IS research in a Delphi study about identifying grand challenges for IS research (Becker et al. 2015). It

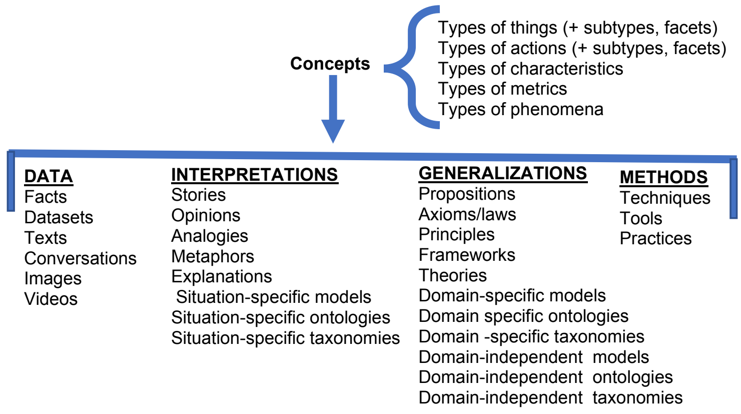

Concepts
Types of things (+ subtypes, facets)
Types of actions (+ subtypes, facets)
Types of characteristics
Types of metrics
Types of phenomena
DATA
Facts
Datasets
Texts
Conversations
Images
Videos
INTERPRETATIONS
Stories
Opinions
Analogies
Metaphors
Explanations
Situation-specific models
Situation-specific ontologies
Situation-specific taxonomies
GENERALIZATIONS
Propositions
Axioms/laws
Principles
Frameworks
Theories
Domain-specific models
Domain specific ontologies
Domain -specific taxonomies
Domain-independent models
Domain-independent ontologies
Domain-independent taxonomies
METHODS
Techniques
Tools
Practices

Figure 19.7: Taxonomy of knowledge objects (Alter 2022b)

was ranked third of 21 as having an impact on the discipline if it were solved. It was ranked 13 of 21 in terms of time frame, i. e., whether it could be dealt with or solved within 10 years. Various aspects of discussions about the possibility of a theoretical foundation for IS led me to conclude that a work system perspective could provide a theoretical foundation for the parts of the IS discipline that are about the development, operation, and maintenance of systems in organizations (and not about the parts of the discipline that focus on a very wide range of other topics such as corporate strategy, impacts of IT spending, the informal use of social media, the challenges of conceptual modeling, and so on). I discussed early versions of that idea in CAIS in 2003 in papers called '18 Reasons Why IT-Reliant Work Systems Should Replace "The IT Artifact" as the Core Subject Matter of the IS Field' and 'Sidestepping the IT Artifact, Scrapping the IS Silo, and Laying Claim to "Systems in Organizations"'. Those papers contributed to the first paper on work system theory (Alter 2013), but did not explain in depth how WST might form the core of a broader work system perspective.

Recently, I tried to define 'the work system perspective' (WSP) as more than just thinking about systems as work systems. A first step was to propose a conceptual scheme that would help in organizing and presenting a broad and deep view of a theoretical perspective. Classification schemes from the natural sciences and social sciences (e. g., biological taxonomy, the Myers-Briggs model of personality types, the periodic table of elements) exemplify ways to organize complex ideas, but a theoretical perspective for IS seemed to require a broad range of ideas of different types. Google Scholar searches on 'theoretical perspective' convinced me that a formal literature review would not be effective and that I would have to develop a new 'framework for describing a theoretical perspective' (an FDTP) for disciplines that involve systems.

Lacking an established approach for creating this type of framework, I used an iterative approach based on analogies with issues and questions from a different discipline that brings rigorous theories, frameworks, models, and methods. I selected particle physics as an object of comparison because it seeks clarity about topics in physics that are loosely analogous to topics about systems, i. e., entity types, key characteristics and phenomena, state transitions, forces, overlaps, interactions, and uncertainties. The gap between IS and physics is very large, but while writing this paper I learned that Mario Bunge used ideas in particle physics as an inspiration for trying to develop a new system-oriented ontology that might supplant the thing-oriented BWW ontology. Lukyanenko et al. (2021) called that incomplete effort the Bunge Systemist Ontology (BSO) to differentiate it from the BWW ontology. Rough analogies with topics and issues in physics proved useful in developing the FDTP, but it would be silly to make too much of any imagined similarity between ideas in IS and ideas describing the behavior of subatomic particles.

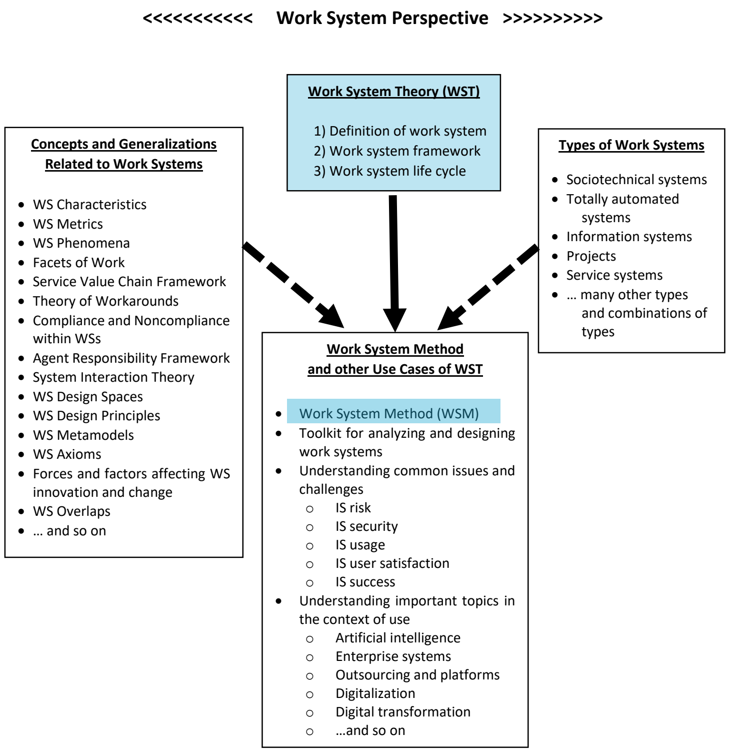
An iterative process explained in Alter (2021c) produced an FDTP that used 7 categories to organize 25 concepts that are all relevant to IS. (That paper's appendix provided a terse illustration that all of those concepts are also relevant to particle physics.) The following list shows how seven categories organize 25 concepts that can be applied to the WSP and to other system-oriented perspectives related to systems:

- Justification: 1) rationale
- Coverage: 2) domain. 3) omissions
- Focal points: 4) primary entity types, 5) special cases of entity types, 6) facets of entities 7) portrayals of entities 8) functions of entities, 9) interactions of entities, 10) overlaps of entities.
- Attributes of entities: 11) characteristics, 12) performance variables, 13) phenomena
- Change: 14) events, 15) trajectories of change, 16) forces
- Generalizations: 17) axioms, 18) design principles, 19) frameworks, 20) theories, 21) models, 22) metamodels, 23) methods
- Fundamental limitations. 24) uncertainties, 25) indeterminacies.

A conference paper on the WSP (Alter 2022a) summarized the FDTP and used it to describe WSP in a way that integrates ideas that often had been discussed in isolation from each other. While other FDTPs might be better, this FDTP helped in pointing to a variety of topics related to work systems that often are not considered fully in generalizations related to information systems. Those include the domain, omissions, alternative portrays, interactions of entities, overlaps of entities, trajectories of change, uncertainties, and indeterminacies. The application of the proposed FDTP to the WSP and digital transformation illustrates how it can help in clarifying the content of a theoretical perspective. Other possible FDTPs might help in other ways.

The four-part configuration of Figure 19.8 summarizes the WSP in a different way that emphasizes the fact that WST is its core. The shading in Figure 19.8 emphasizes the link between WST and the WSM. Notice that the WSP has a specific core but does not have a definitive boundary, as indicated by the way three parts of Figure 19.8 end with some version of 'and so on.' The four parts of Figure 19.8 express the following:

1. WST consists of the definition of work system, the work system framework (elements of a basic understanding of how a work system operates) and the work system life cycle model (how work systems evolve over time through planned and unplanned change).
2. WST applies to different types of work systems, such as sociotechnical systems, totally automated systems, information systems, projects, and so on (noting that sociotechnical vs. totally automated is the most important distinction and that other subtypes are not a strict hierarchy).
3. WST is the basis of many concepts and generalizations (see Figure 19.7) related to work systems. Concepts and generalizations related to WST start with WS characteristics,



<<<<<<<<<<< Work System Perspective >>>>>>>>>>
Work System Theory (WST)
1) Definition of work system
2) Work system framework
3) Work system life cycle
Concepts and Generalizations Related to Work Systems
• WS Characteristics
• WS Metrics
• WS Phenomena
• Facets of Work
• Service Value Chain Framework
• Theory of Workarounds
• Compliance and Noncompliance within WSs
• Agent Responsibility Framework
• System Interaction Theory
• WS Design Spaces
• WS Design Principles
• WS Metamodels
• WS Axioms
• Forces and factors affecting WS innovation and change
• WS Overlaps
• ... and so on
Types of Work Systems
• Sociotechnical systems
• Totally automated systems
• Information systems
• Projects
• Service systems
• ... many other types and combinations of types
Work System Method and other Use Cases of WST
• Work System Method (WSM)
• Toolkit for analyzing and designing work systems
• Understanding common issues and challenges
○ IS risk
○ IS security
○ IS usage
○ IS user satisfaction
○ IS success
• Understanding important topics in the context of use
○ Artificial intelligence
○ Enterprise systems
○ Outsourcing and platforms
○ Digitalization
○ Digital transformation
○ ...and so on

Figure 19.8: Work system perspective

performance variables, and phenomena that may apply to WSs as a whole or to WS elements. WS-related generalizations include inheritance from WS in general to special cases, WS design principles, a WS view of service, a WS view of workarounds, and so on.

4. WST and many related concepts and generalizations are the basis of WSM, of tools for analyzing and designing work systems, of ways to understand common issues and challenges, and of ways to understand important topics in usage contexts. WSM is the most important application of WSP, but many other applications are relevant to practice and to research.

I believe that the WSP could serve as at least a first cut at a theoretical foundation for the parts of the IS field that are about the development, operation, maintenance, and evolution of systems in organizations (thus addressing the grand challenge mentioned in the introduction). However, as I wrote several versions of a paper on that topic I kept returning something that Ulrich Frank had said to me in the past, probably about a different topic: 'That is an important topic but you will never be able to publish it.'

Rejections of two versions of that paper convinced me that his comment would apply to almost any paper on WSP as a theoretical foundation. The reviews touched on many of the same issues that I thought about when I wrote the papers, e. g., what about giving full credit to many relevant parts of the literature, what about discussions of alternative views of what might be a foundation for IS, what about the pluralistic nature of the IS field, what about the largely self-referential nature of much of the presentation, what about non-systematic use of social media, what about justifying the criteria that are used for claiming that the WSP could serve as a theoretical foundation, and what about distinguishing the paper's new ideas from older ideas that have been published previously? All of those 'what abouts' deserve discussion, but in combination cannot be addressed adequately within a single paper even if typical length limitations are relaxed substantially. Providing complete and integrated coverage of the WSP that goes beyond the coverage in Alter (2021c) and Alter (2022a) would be very difficult, because both of those papers cram a great deal of material into ten dense pages. A complete explanation would require at least one and possibly several books that probably would not have a large audience even though the Becker et al. (2015) Delphi study viewed the topic as a grand challenge for IS research.

19.13 What Comes Next?

My conclusion at this point is that future development of WSP might make it more valuable for research, teaching, and practice regardless of whether it is viewed as achieving a grand challenge for the IS field and regardless of whether it can be published in leading journals. Thus, its evolution should continue in whatever ways make sense regardless of whether reviewers believe that some of the ideas along the way are only partly justified or do not fully reflect past publications that are somewhat related to whatever is the next extension.

One approach is to drop controversial claims that WSP might provide a theoretical foundation for major parts of the IS discipline and to proceed modestly by simply illustrating that it could be viewed (metaphorically) as a 'common denominator' for much of the IS field. When someone expounds on the nature of big data or social media or artificial intelligence, the common denominator metaphor would assume that current real world applications of digital agents of any type can be described in work system terms. This includes what the system entails (who are the participants, what are the activities, what product/services are produced, etc.), how the work system life cycle model applies, and how some of the extensions of WST help in producing deeper understandings.

A recent use of that approach applied the WSP to AI in light of the common practice of generalizing about AI without defining AI in a specific way. The common denominator approach led me to think of AI applications as applications of algorithmic agents whose developers imagined they were applying AI. The real question was about the application of algorithmic agents with certain capabilities and responsibilities, and not about what the developers imagined or about providing a chain of references going back to 1955 and generalizing about capabilities of AI based on a specific example that uses an algorithm associated with AI. My most recent paper along those lines is 'How Can You Verify that I Am Using AI? Complementary Frameworks for Describing and Evaluating AI-Based Digital Agents in their Usage Contexts' (Alter 2023). It built partly on the agent-responsibility framework (Figure 19.6) and partly on an earlier paper called 'Understanding artificial intelligence in the context of usage: Contributions and smartness of algorithmic capabilities in work systems' (Alter 2022d).

The 'why are you doing that' question from my 2017 presentation to Ulrich's Group applies to those papers just as it applies to the other topics and papers mentioned here. I think those topics are interesting, valuable, and worth pursuing just as I think that Ulrich's more rigorous steps closer to the spirit of automatic programming are interesting, valuable, and worth pursuing. I hope that Ulrich would agree.

References

Alter, S. (2006). *The Work System Method: Connecting People, Processes, and IT for Business Results.* Larkspur, USA: Work System Press.

Alter, S. (2008). 'Defining information systems as work systems: implications for the IS field'. In: *European Journal of Information Systems* 17 (5), pp. 448–469.

Alter, S. (2010). 'Viewing Systems as Services: A Fresh Approach in the IS Field'. In: *Communications of the Association for Information Systems* 26 (11), pp. 195–224.

Alter, S. (2013). 'Work System Theory: Overview of Core Concepts, Extensions, and Challenges for the Future'. In: *Journal of the Association for Information Systems* 14 (2), pp. 72–121.

Alter, S. (2016). 'Encapsulation as a key concern in analysis and design for service systems'. In: *Proceedings of AMCIS 2016.*

Alter, S. (2020a). 'Ten Lightweight SA&D Tools Based on Work System Theory and Its Extensions'. In: *Proceedins of AMCIS 2020.*

Alter, S. (2020b). 'The Philosopher's Corner: Taking Different Types of Knowledge Objects Seriously: A Step toward Generating Greater Value from IS Research'. In: *ACM SIGMIS DATABASE for Advances in Information Systems* 51 (4), pp. 123–138.

Alter, S. (2021a). 'An Open-Ended Work System Knowledge Model for Visualizing, Organizing, and Accessing Knowledge about Information Systems in Organizational Settings'. In: *Proceedings of HICSS 2021.*

Alter, S. (2021b). 'Facets of Work: Enriching the Description, Analysis, Design, and Evaluation of Systems in Organizations'. In: *Communications of the Association for Information Systems* 49 (13), pp. 321–354.

Alter, S. (2021c). 'Framework for Describing a Theoretical Perspective: Application to the Bunge-Wand-Weber Ontology and General Systems Theory'. In: *Proceedings of ACIS 2021.*

Alter, S. (2022a). 'A Framework for Describing Theoretical Perspectives: Overview and Application to the Work System Perspective'. In: *Proceedings of HICSS 2022.*

Alter, S. (2022b). 'Extending a Work System Metamodel Using a Knowledge Graph to Support IS Visualization and Development'. In: *Proceedings of the ER Forum.*

Alter, S. (2022c). 'Responsibility Modeling for Operational Contributions of Algorithmic Agents'. In: *Proceedings of AMCIS 2022.*

Alter, S. (2022d). 'Understanding artificial intelligence in the context of usage: Contributions and smartness of algorithmic capabilities in work systems'. In: *International Journal of Information Management.*

Alter, S. (2023). 'How Can You Verify that I Am Using AI? Complementary Frameworks for Describing and Evaluating AI-Based Digital Agents in their Usage Contexts'. In: *Proceedings of HICSS 2023.*

Alter, S. and Bolloju, N. (2016). 'A Work System Front End for Object-Oriented Analysis and Design'. In: *International Journal of Information Technology and Systems Approach* 9 (1), pp. 1–18.

Alter, S. and Recker, J. (2017). 'Using a work system perspective to expand BPM research use cases'. In: *Journal of Information Technology and Technology Applications* 18 (1), pp. 47–71.

Becker, J., vom Brocke, J., Heddier, M. and Seidel, S. (2015). 'In search of information systems (grand) challenges'. In: *Business & Information Systems Engineering* 57 (6), pp. 377–390.

Boell, S. K. and Cecez-Kecmanovic, D. (2015). 'What is an information system?' In: *2015 48th Hawaii International Conference on System Sciences*. IEEE.

Bolloju, N., Alter, S., Gupta, A., Gupta, S. and Jain, S. (2017). 'Improving scrum user stories and product backlog using work system snapshots'. In: *Proceedings of AMCIS 2017*.

Bork, D. and Alter, S. (2020). 'Satisfying four requirements for more flexible modeling methods: Theory and test case'. In: *Enterprise Modelling and Information Systems Architectures (EMISAJ)* 15 (3), pp. 1–25.

Coase, R. H. (1988). 'The Nature of the Firm: Origin'. In: *Journal of Law, Economics & Organization* 4 (1), pp. 3–17.

Ferrario, R., Guarino, N., Trampus, R., Laskey, K., Hartman, A. and Gangadharan, G. R. (2012). 'Service System Approaches: Conceptual Modeling Approaches for Services Science'. In: *Handbook of Service Description: USDL and Its Methods*. Ed. by A. Barros and D. Oberle. Springer, pp. 75–109.

Ferstl, O. K. and Sinz, E. J. (2013). *Grundlagen der Wirtschaftsinformatik*. München: Oldenbourg Verlag.

Frank, U. (2013). 'Multi-Perspective Enterprise Modelling: Foundational Concepts, Prospects and Future Research Challenges'. In: *Software & Systems Modeling* 13 (3), pp. 941–962.

Gross, S., Stelzl, K., Grisold, T., Mendling, J., Röglinger, M. and vom Brocke, J. (2021). 'The Business Process Design Space for exploring process redesign alternatives'. In: *Business Process Management Journal* 27 (8), pp. 25–56.

Hassan, N. R., Lowry, P. B. and Mathiassen, L. (2022). 'Useful products in information systems theorizing: A discursive formation perspective'. In: *Journal of the Association for Information Systems* 23 (2), pp. 418–446.

Hassan, N. R. and Mathiassen, L. (2018). 'Distilling a body of knowledge for information systems development'. In: *Information Systems Journal* 28 (1), pp. 175–226.

Hirschheim, R. (2019). 'Against theory: With apologies to Feyerabend'. In: *Journal of the Association for Information Systems* 20 (9), pp. 1340–1357.

Hirschheim, R. and Klein, H. K. (2003). 'Crisis in the IS field? A critical reflection on the state of the discipline'. In: *Journal of the Association for Information Systems* 4 (5), pp. 237–293.

Hovorka, D. S., Rowe, F., Markus, M. L., Jarvenpaa, S., Swanson, E. B., Lacity, M., Burton-Jones, A., Venkatesh, V. and Hirschheim, R. (2019). 'Scholarly Commentaries on Hirschheim's 'Against Theory''. In: *Journal of the Association for Information Systems* 20 (9), pp. 1358–1389.

Kautz, K. K. (2018). 'Debate Section Editorial Note: Is Information Systems a Science?' In: *Communications of the Association for Information Systems* 43 (1), p. 8.

Keen, P. (1980). 'MIS Research: Reference Disciplines and a Cumulative Tradition'. In: *Proceedings of ICIS 1980*.

Kendall, J. E. and Kendall, K. E. (1993). 'Metaphors and methodologies: Living beyond the systems machine'. In: *MIS Quarterly* 17 (2), pp. 37–47.

Lemey, E. and Poels, G. (2011). 'Towards a Service System Ontology for Service Science'. In: *International Conference on Service-Oriented Computing*. Springer, pp. 250–264.

Lukyanenko, R., Storey, V. and Pastor, O. (2021). ‘Foundations of Information Technology Based on Bunge’s Systemist Philosophy of Reality’. In: *Software and Systems Modeling* (20), pp. 921–938.

McBride, N. (2018). ‘Is information systems a science?’ In: *Communications of the Association for Information Systems* 43 (9), pp. 163–174.

Morgan, G. (1986). *Images of organization.* Thousand Oaks, CA: Sage.

Oates, B. J. and Fitzgerald, B. (2007). ‘Multi-metaphor method: Organizational metaphors in information systems development’. In: *Information Systems Journal* 17, pp. 421–339.

Oberle, D., Barros, A., Kylau, U. and Heinzl, S. (2013). ‘A unified description language for human to automated services’. In: *Information Systems* 38 (1), pp. 155–181.

Reijers, H. A. and Mansar, S. L. (2005). ‘Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics’. In: *Omega* 33 (4), pp. 283–306.

Ruth, G., Alter, S. and Martin, W. (1981). *A Very High Level Language for Business Data Processing.* Technical Report MIT/LCS/TR-254. MIT Laboratory for Computer Science.

Sherer, S. A. and Alter, S. (2004). ‘Information systems risks and risk factors: are they mostly about information systems?’ In: *Communications of the Association for Information Systems* 14 (2), pp. 29–64.

Spohrer, J., Vargo, S., Caswell, N. and Maglio, P. (Jan. 2008). ‘The service system is the basic abstraction of service science’. In: *Proceedings of HICSS 2008.* IEEE, pp. 104–104.

Stamper, R. (2001). ‘Organisational semiotics: Informatics without the computer?’ In: *Information, Organisation and Technology.* Boston, MA: Springer, pp. 115–171.

Tan, X., Alter, S. and Siau, K. (2011). ‘Using service responsibility tables to supplement UML in analyzing e-service systems’. In: *Decision Support Systems* 51 (3), pp. 350–360.

Truex, D., Alter, S. and Long, C. (2010). ‘Systems analysis for everyone else: Empowering business professionals through a systems analysis method that fits their needs’. In: *Proceedings of ECIS 2010.*

van der Aalst, W. (2013). ‘Business Process Management: A Comprehensive Survey’. In: *ISRN Software Engineering*, pp. 1–37.

Vargo, S. L. and Lusch, R. F. (2016). ‘Institutions and axioms: an extension and update of service-dominant logic’. In: *Journal of the Academy of marketing Science* 44 (1), pp. 5–23.

Chapter 20

What a Wonderful Modeling World! Tackling Conceptual Diversity in Enterprise and Information Systems Modeling

Marc Frerichs and Markus Nüttgens

Starting with flow process charts, science and practice have developed a large and complex modeling landscape over the last 100 years based on a variety of modeling scenarios with an almost unmanageable number of generalistic or domain-specific methods. The resulting conceptual diversity has led to the existence of modeling methods for nearly every conceivable use case. Modeling, which is supposed to make complexity controllable through structure, has itself become a complex world. The selection of a modeling method optimal for the use case is difficult today. Due to the variety of business processes, the situation is aggravated by the circumstance that, depending on the use case, different modeling techniques are often used, resulting in inflexibility. In this chapter, we conduct a literature review on diversity in the area of conceptual modeling, and specifically in enterprise and process modeling, with the primary question of which theories, methods, and tools exist to describe the modeling landscape's diversity and make it manageable. The results of our work show that there have been different approaches over the last decades (meta modeling, standardization, and model transformation), but none of them could really prevail. As a thought-provoking teaser, we propose an approach for an independent middleware layer (process model warehouse concept) that can extract data from (enterprise) systems and enrich it using linked Business Process Management (BPM) tools and vice versa.

20.1 Introduction

'You have a character that's all of a piece, and you want the whole of life to be of a piece too—but that's not how it is. [...] All the diversity, all the charm, all the beauty of life is made up of light and shade' was one of Oblonsky's nuggets of wisdom addressed to his friend Levin in Tolstoy's work Anna Karenina. Indeed, the world is complex, diverse, full of uncertainties and it's definitely not 'all of a piece.' The interplay of things – as parts of a bigger whole – is what makes the world so exciting, but also hard to describe. The intention to master this complexity led to the development of various modeling approaches.

Models, as abstract representations of real-world objects, go back to the Stone Age. In anthropology, the ability to abstract and build models is seen as a competitive advantage of human beings. Schichl (2004) summed up the long history of modeling, providing a *'very fast tour of 30,000 years of modeling history'* and shows that modeling, as a research area, itself is very diverse. For example, the properties of a model depend heavily on the use case and the modeling goal. Models may vary, e. g., in their level of detail or formality. Furthermore, the diverse areas of application led to different approaches and requirements for modeling.

In the last few decades alone, a multitude of methods, modeling languages, and tools have been developed. And although our oldest ancestors used modeling, in whatever way, to make their lives easier, even the most basic questions are still relevant, e. g., 'what is the definition of the term *model*?' (Thomas 2005; Hesse and Mayr 2008). Also, the perspective we take concerning modeling plays a significant role. Are we dealing with syntax, semantics, or pragmatics (Thalheim 2012)? This is where true diversity manifests itself in modeling. We are also constantly expanding the field of modeling through the research and development of, e. g., new methods, modeling languages, or tools. What was once supposed to reduce complexity has increasingly developed into a very complex subject itself. On the other hand, there is an undeniable trend towards a prevalent modeling technique: The *Business Process Modeling Notation* (BPMN) became a quasi-standard in many respects. With its extensive specification, it is widely used in research, teaching and practice. BPMN is here to stay – for a lot of good reasons. Unfortunately, the focus on one 'all-encompassing' modeling technique often leads to ignoring other techniques that may be more appropriate in a given context. In many cases, the impression is given that there is nothing else in the world. And yet the modeling landscape is so diverse. In this work, we try to provide an excerpt of the beauty and richness of our beloved research field and, in doing so, want to make a call for more diversity in terms of theories, methods, and tools. We also try to summarize existing research results in the area of diversity in conceptual modeling and which approaches exist that foster or cope with conceptual diversity in modeling.

Conceptual modeling, in general, is a core discipline within (business) informatics. Although we will address methods that are also or mainly used in the software development context and topics regarding conceptual modeling in general, this work deals with *conceptual modeling in the area of Enterprise Information Systems.* In particular, we consider the fields of *enterprise modeling*, i. e., modeling techniques related to the design of corporate information systems, and *business process modeling*, i. e., modeling techniques for representing processes of an enterprise. So when we talk about modeling in the following, we usually mean modeling in the said context.

Our work is structured as follows: Section 20.2 provides a general overview of diversity in conceptual modeling and a definition of the term *diversity* to establish a common understanding. In Section 20.3, we present our research method. Section 20.4, which is the main contribution of this chapter, deals with the classification and explanation of the work we found. The results are discussed in Section 20.5. A final outlook on our vision of a better modeling world is shown in Section 20.6.

20.2 Conceptual Diversity in Enterprise and Business Process Modeling

We have already indicated why diversity is a big thing in modeling. But first things first. To further discuss on a well-founded basis, some terminology must be clarified. When talking about conceptual diversity in modeling, what do we really mean? A quick look at Google Trends shows that the topic *diversity* is generally quite trendy. Figure 20.1 shows

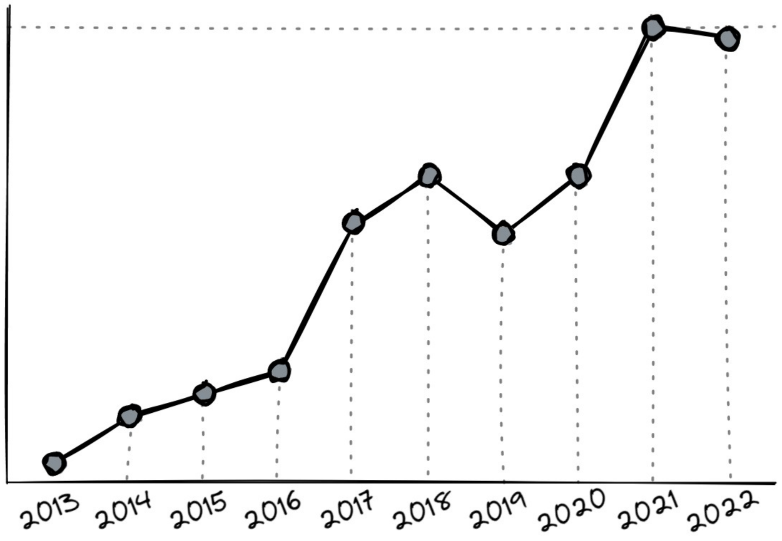


Figure 20.1: Diversity (Worldwide) – Interest over time (Source: Google Trends)

the development of the world-wide yearly average over the last ten years of the so-called *interest over time*, a term coined by Google, representing the search interest.

The term *diversity* has demonstrably gained momentum in recent years. In most cases, this is due to increased research in sociology and higher public interest and awareness for this topic. Although sociological questions regarding diversity in modeling would be certainly also exciting, we expressly do not want to pursue them at this point. We therefore do *not* define diversity as the characteristics that can be used to describe commonalities and differences between people, but in terms of *variety*, like *the absence of uniformity* regarding the *different conceptual aspects* in modeling. Thus, we refer to it as *conceptual diversity.*

Definition (Conceptual Diversity in Modeling). *Conceptual diversity in modeling describes the variety and absence of uniformity in methodologies, languages, formats, and tools in modeling.*

Conceptual modeling has always been an enabler, for example, in engineering disciplines and has contributed to better communication of complex issues. Especially due to the increasing process orientation, the business process reengineering movement (Hammer and Champy 1994), and extensive research in the 90s and early 2000s, modeling has also established itself in business administration, especially for modeling information systems (Scheer and Nüttgens 2000). The modeling of business processes is of particular importance in this context as companies are defined as the interplay of horizontally structured business processes (*process-centered organization*) (Hammer 1996; Davenport 1993). With the rise of business process modeling came the development of numerous modeling methods and tools, leading to problems in terms of standardization and intercompatibility. There has been a lot of research in conceptual modeling in the last decades while addressing a multitude of application domains which also led to a more heterogeneous landscape of process modeling.

Even the discussion about the definition of the term *model* shows the diversity in modeling. Thomas (2002) identified around 50 different kinds of models in computer science. There are mainly four approaches to modeling in business informatics (Thomas 2005), namely (1) the general model definition by Stachowiak (1992), (2) the axiomatic model definition, (3) the mapping-based model definition, and (4) the construction-oriented model definition, while there exist around 35 notions for model definitions that are commonly used in business informatics (Thalheim 2013). Thalheim (2010) and Thalheim (2013) offers extensive insights into the theory of conceptual modeling and the conception of the model. In conclusion, models have been defined through a variety of properties, purposes, and functions.

Likewise, the sheer amount of terms used in the context of modeling (besides the term *model*) is overwhelming: (business) process, process instance, workflow, (business) process model, meta model, meta-meta model, modeling method, modeling language, modeling process, modeling tool – just to provide a selection of the more basic terms. With regard to process modeling, various modeling languages can be named, from domain-specific to general-purpose, while there are mainly two paradigms of central importance: process-oriented and object-oriented modeling. In the following, we briefly summarize the most important modeling languages in the field.

The *Event-driven Process Chain* (EPC) is a semi-formal method for designing and analyzing business processes (Keller et al. 1992). It was developed as part of the framework of *Architecture of Integrated Information Systems* (ARIS) at the Institute for Business Information Systems at the University of Saarland in 1992 and focuses on the dynamic behavior of information systems (control flow analysis). In particular, it has gained popularity through the conjunction with modeling in SAP R/3 and became a widely used technique in business process modeling (Keller 1999). An EPC is an ordered graph of events, functions, various connectors (allowing alternative/parallel execution), and logical operators. The method is well-known for its simplicity, e. g., with regard to the intuitive notation. The standard set of the EPC was followed by extensions such as the eEPC (extended EPC) or the oEPC (object-oriented EPC) (Nüttgens and Zimmermann 1998).

In 2004, BPMN was published in version 1.0. BPMN is a standard for business process modeling (Object Management Group (OMG) 2008), providing a graphical notation to describe business processes in a Business Process Diagram (BPD). BPMN was initially developed by the Business Process Management Initiative (BPMI) in order to provide a notation that is understandable by all business users and was later maintained by the OMG, as the BPMI and OMG merged in 2005. BPDs are described with a limited set of graphical elements which can be grouped into these four basic categories: Flow objects, connecting objects, swim lanes, and artifacts. With the introduction of BPMN 2.0 in 2011, the standard was renamed to Business Process Model and Notation to reflect the execution semantics. In 2013, BPMN became an ISO standard (ISO/IEC 19510:2013). The current version of BPMN, at the time of writing, is 2.0.2 from January 2014.

Petri nets are one of the oldest and best-researched notations for process modeling. The theory of Petri nets goes back to the dissertation of Carl Adam Petri (Petri 1962). Originally, Petri nets were intended to model distributed systems or events that could run concurrently, i. e., independently of one another. Since then, Petri nets have spread around the world and are used in a wide variety of application areas, for example, for process modeling in operating systems, modeling of processes in systems biology, production systems, or business processes. Petri nets are not only characterized by their graphic representation, but above all by their mathematically sound specification.

Yet Another Workflow Language (YAWL) is a process modeling language based on workflow patterns (Van der Aalst et al. 2003) and, at the same time, an open-source framework for process execution (workflow management system) (Van der Aalst and ter Hofstede 2005). The visual representation of a YAWL graph is based to a large extent on the mathematical fundamentals of Petri nets. It is, so to speak, an extension of Petri nets which has been specially adapted to the workflow management world. Every process in YAWL has a start and an end state. Activities in YAWL are called tasks. Conditions in YAWL correspond to states in Petri nets, whereby – in contrast to Petri nets – it is possible to connect tasks with one another without inserting a state in between. YAWL has clearly defined join and split semantics.

The *Universal Modeling Language* (UML) was a response to the various modeling languages and methods proposed for object-oriented software engineering (Object Management Group (OMG) 2017) and is now the most important language for software systems modeling. The UML provides different diagrams for two categories: Structural diagrams, such as class diagrams or composite structure diagrams, and behavioral diagrams like activity diagrams or state machine diagrams. This makes the UML a general-purpose modeling language for the use in many different fields of application – from software systems modeling to business process modeling.

As the name suggests, *Entity-Relationship Models* (ERM) are used to describe entity types and their relationships. ERM was introduced in 1975 by Peter Chen in his publication 'The Entity-Relationship Model' (Chen 1975). ERM are the de facto standard for data modeling and are used both as a conceptual basis in application development and for database design, mainly for relational databases, in database research. ERM are primarily conceptual and are often used as ontology that expresses predicates in a domain of knowledge. Therefore, being static models, ERM can be a good foundation for purely process-oriented (dynamic) models.

There are many other modeling approaches for a wide variety of purposes, from XML-based modeling languages such as the XML Process Definition Language (XPDL) or the Business Process Execution Language (BPEL) to basic graphical diagrams like flowcharts. Figure 20.2 shows the development of several graphical modeling languages over time.

The purpose of this short summary was to give an overview of the variety of methods and languages used in modeling. It is obvious that the number of modeling languages and methods is accompanied by a large number of corresponding modeling tools. There is an enormous number of solutions for a wide variety of problems. But how can these established solutions, which are best suited for the respective field of application, be integrated in a meaningful way and thus develop their full potential?

20.3 Research Methodology

The research goal was to gather insightful information on how to deal with diversity in modeling and examine what research has already been conducted. To focus on relevant information and to provide a framework for the literature search, we have established three research questions based on aspects of the *Design Science Research* (DSR) approach. DSR involves creating artifacts or design theories and has gained increased popularity in the scientific community in the early 2000s. Gregor and Hevner (2013) distinguish three contribution types of *research deliverables* as outputs of DSR, level 1 – implementation of

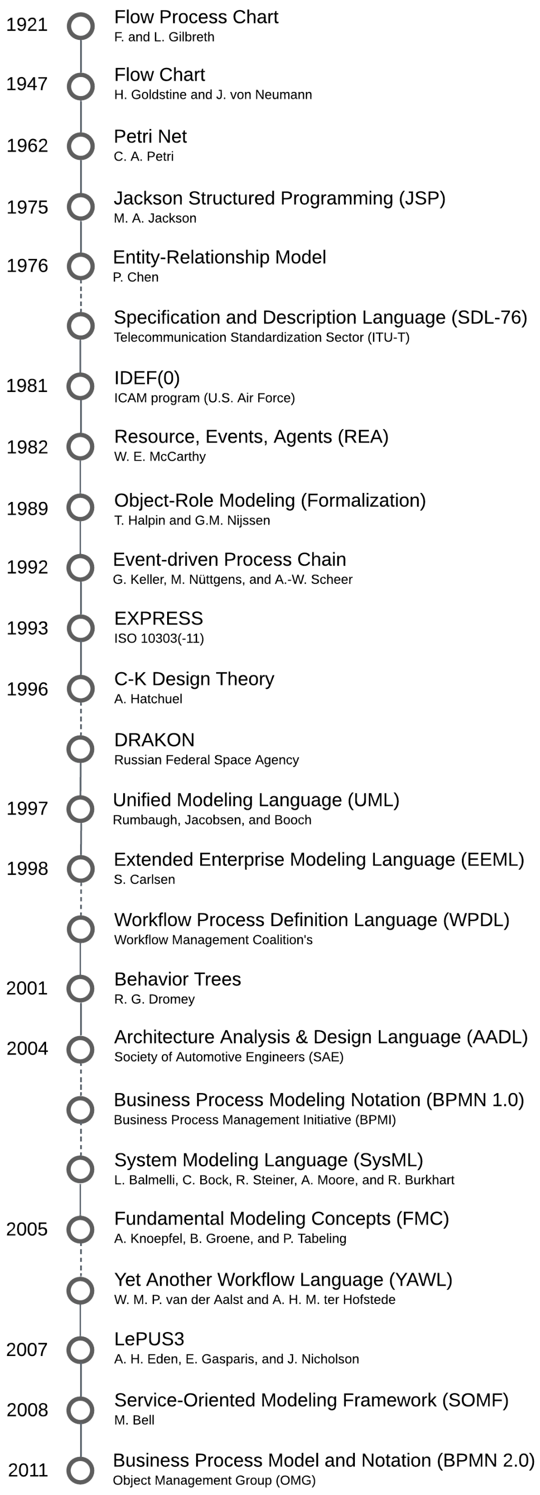

1921
Flow Process Chart
F. and L. Gilbreth
1947
Flow Chart
H. Goldstine and J. von Neumann
1962
Petri Net
C. A. Petri
1975
Jackson Structured Programming (JSP)
M. A. Jackson
1976
Entity-Relationship Model
P. Chen
Specification and Description Language (SDL-76)
Telecommunication Standardization Sector (ITU-T)
1981
IDEF(0)
ICAM program (U.S. Air Force)
1982
Resource, Events, Agents (REA)
W. E. McCarthy
1989
Object-Role Modeling (Formalization)
T. Halpin and G.M. Nijssen
1992
Event-driven Process Chain
G. Keller, M. Nüttgens, and A.-W. Scheer
1993
EXPRESS
ISO 10303(-11)
1996
C-K Design Theory
A. Hatchuel
DRAKON
Russian Federal Space Agency
1997
Unified Modeling Language (UML)
Rumbaugh, Jacobsen, and Booch
1998
Extended Enterprise Modeling Language (EEML)
S. Carlsen
Workflow Process Definition Language (WPDL)
Workflow Management Coalition's
2001
Behavior Trees
R. G. Dromey
2004
Architecture Analysis & Design Language (AADL)
Society of Automotive Engineers (SAE)
Business Process Modeling Notation (BPMN 1.0)
Business Process Management Initiative (BPMI)
System Modeling Language (SysML)
L. Balmelli, C. Bock, R. Steiner, A. Moore, and R. Burkhart
2005
Fundamental Modeling Concepts (FMC)
A. Knoepfel, B. Groene, and P. Tabeling
Yet Another Workflow Language (YAWL)
W. M. P. van der Aalst and A. H. M. ter Hofstede
2007
LePUS3
A. H. Eden, E. Gasparis, and J. Nicholson
2008
Service-Oriented Modeling Framework (SOMF)
M. Bell
2011
Business Process Model and Notation (BPMN 2.0)
Object Management Group (OMG)

Figure 20.2: Timeline of several graphical modeling languages

artifacts like software products, level 2 – knowledge as operational principles like methods, level 3 – theories. The research questions (RQ) were chosen according to these contribution types.

RQ 1. *Which theories exist of conceptual diversity in modeling?*

RQ 2. *What are the methodologies regarding conceptual diversity in modeling?*

RQ 3. *How do tools address conceptual diversity in modeling?*

According to the research questions, our research goal is to collect and evaluate relevant information about diversity in (process) modeling as well as methodological approaches to create a flexible modeling world. Additionally, we collected modeling tools that already provide implementations for model transformations. We used *Scopus* as our main data source which, according to Elsevier, is the largest abstract and citation database of peer-reviewed literature that includes journal articles, books, and conference proceedings. The phases of our literature analysis process and the number of papers per phase can be seen in Figure 20.3.

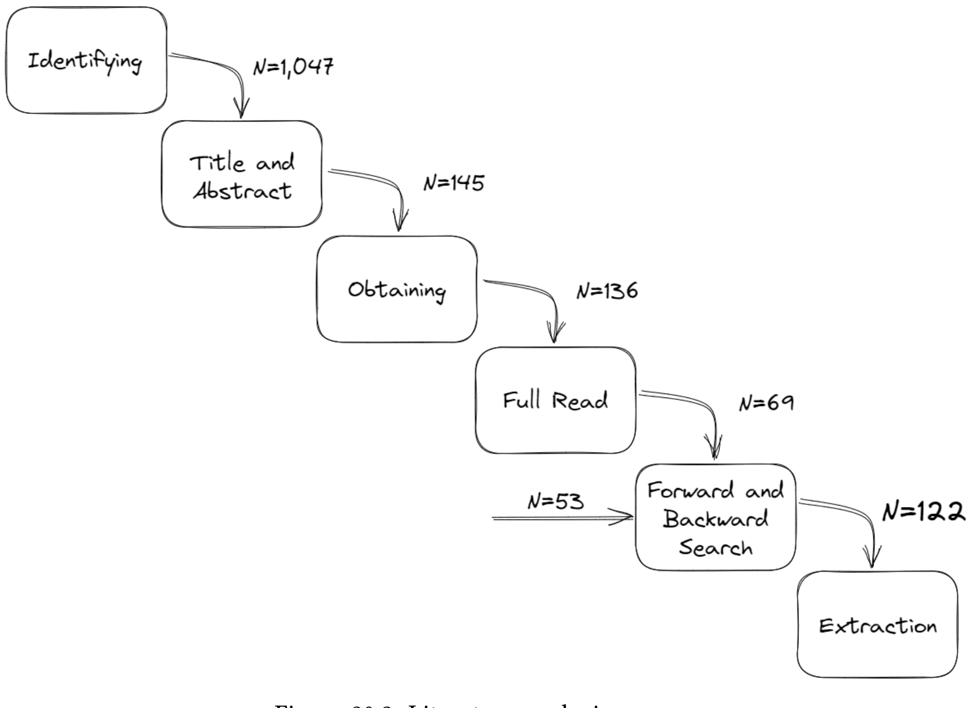


Figure 20.3: Literature analysis process

We do not claim that this literature review is complete and covers all existing and relevant articles. A limiting factor of our literature review is the use of only one database. Forward and backward searches confirmed that some relevant papers were not documented in the Scopus database, as they could not be found using either keywords or a direct title search. Specifically, this chapter is intended to be a research contribution representative of the topic of diversity in the modeling community that addresses problems in the landscape. Also, it addresses state-of-the-art and existing approaches to address the problem.

Relevant literature was identified using established research criteria found in 1,047 eligible papers. The search criteria consisted of the following search term: (*diversity* OR *heterogeneity* OR *variety*) AND (*conceptual modeling* OR *enterprise modeling* OR *process modeling*). As can be seen from the search term, we use the terms *diversity*, *heterogeneity*, and *variety* as synonyms in this work. The individual terms were allowed to appear in the title, abstract, keywords, or text. We included articles, conference papers, and books or individual book chapters. The search was not limited by publication date.

In the second phase of our literature review, all papers were examined based on title and abstract in the first step. This resulted in excluding 902 papers (including 26 entries that generally referred to a proceeding). These steps were performed directly through the Scopus platform. The exclusion of this set of papers is particularly justified because many different domains (including geology, architecture, energy engineering, healthcare) use specific modeling methods and did not add value to our questions. We also excluded specific orientations such as process mining, business management, and process management that were not relevant to our topic.

At the third stage of the evaluation, 145 contributions were left. Due to restricted access to certain databases, we were unable to obtain nine papers that met the previous criteria, so we excluded them. In the final fourth phase of the literature review, we had 136 papers remaining, which we read. We classified 69 of them as relevant to our study. In addition, through backward and forward searches of literature valuable to our topic, we identified 53 additional sources of interest that were not found through the Scopus database and our search term. We directly read these papers in full and inserted them into the overall context of this work, resulting in a total of 122 papers for data extraction.

In the data extraction phase, according to our research questions, we focused on (1) theories about diversity/heterogeneity in the modeling world (RQ1), see Section 20.4.1, (2) existing methods to handle them (RQ2), see Section 20.4.2, and (3) modeling tools that already provide implementations to transform process models (RQ3), see Section 20.4.3.

20.4 Literature Analysis and Synthesis

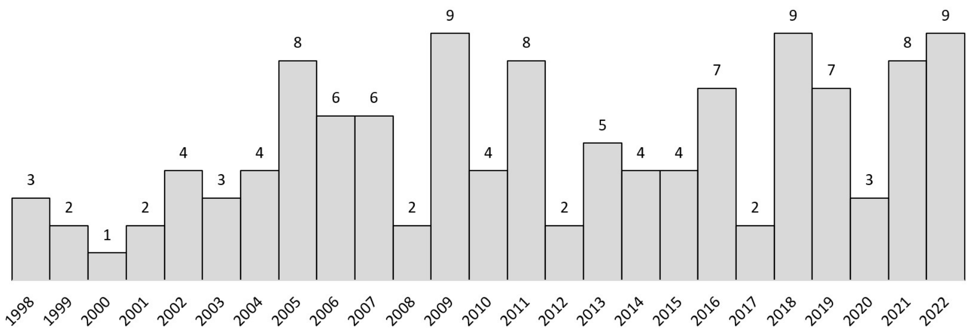


Figure 20.4: Numbers of publications in our last literature analysis phase from 1998 to 2022

Table 20.1: Structure of our literature analysis section

Main topic and *subtopics*
Theories (Section 20.4.1)
1. *Stakeholder diversity*
2. *Requirement differences*
3. *Model heterogeneity*
4. *Process variants*
5. *Method and tool variety*
Methodologies (Section 20.4.2)
1. *Model transformation*
2. *Meta modeling and ontologies*
3. *Standardization*
4. *Modeling techniques embracing diversity*
Tools (Section 20.4.3)
1. *Modeling tools supporting standards and intercompatibility*
2. *Lack of interoperability of tools*
3. *Model transformation tools*
4. *Meta modeling platforms*

The analysis shows that the distribution of articles of interest for our topic is predominantly homogeneous – but there is an upward trend. The first signs of growing publication numbers in the years 2005 to 2009 can be attributed to the increased research activities in the field of model transformation (Mendling) and meta modeling (Karagiannis, Kühn). Figure 20.4 shows the distribution of publications in our final literature analysis phase by year. The data from recent years is consistent with Google's trend analysis, shown in Figure 20.1, which could indicate that the term *diversity* has received particular attention in general. Given the papers found, it is safe to say that diversity as a research area plays a role in the modeling community and that some researchers in particular regularly publish work on the topic.

We conclude from the analysis that there is a growing interest in dealing with the plethora of different methods and tools that have emerged over the last two decades. After reviewing the literature, we identified additional subtopics to the three overarching areas of theories, methods, and tools that arise from the research questions. The resulting structure can be seen in Table 20.1. Based on this structure, the detailed results of our literature review are explained in the following.

The results of the underlying literature review show that the use of heterogeneous methods and tools as well as the different application domains and the participation of various stakeholders (leading to different views on models) led to various research approaches to deal with the diversity in conceptual modeling. Thus, this diversity leads to method and tool incompatibility. Various approaches can be derived from the literature that address the diversity and consequently the integration of the methods. There is also a variety of process modeling tools that address the diversity in modeling in different ways. In order to answer the research questions set out in Section 20.3, the aspects mentioned above are discussed in the following.

20.4.1 Theories of Diversity in Conceptual Modeling

Addressing the first research question, we used the literature to identify theories about diversity in modeling and why diversity exists in modeling. In his *Computers in Industry* paper on the evolution of BPM (Reijers 2021), Reijers describes (process) modeling as 'an import and ongoing theme' with a variety of purposes that have made it a strong tradition within BPM research. With regard to (process) modeling he reviews, among others, the following topics: (1) process models as a *direct benefit* to managers aiming for business process improvement, (2) *the process model itself* as a means to put BPM into practice, (3) *the importance of the process modeling act* itself, (4) models as assets in and by themselves, which may also serve as *reference models* in similar contexts, (5) *techniques and tools* that can be used for process modeling, (6) the importance of *context* for different purposes of process model usage. This enumeration can certainly be transferred to conceptual modeling in general. It already gives an indication of how extensive and diverse the area under consideration is and from how many different perspectives we can discuss it.

In the following, we outline five theories on diversity in modeling. The reader will find that the theories are not always clear-cut and in many cases are mutually dependent.

Stakeholder Diversity

Stakeholder diversity is based on 'the assumption that the information about the objects represented in the information system (IS) is specified and verified by domain experts and potential users' (Castellanos et al. 2020). As information systems are developed for a broader range of users, such as 'customers, suppliers, and members of the general public [...], analysts can no longer rely on a stable single group of people for the complete specification of domains.' This heterogeneous setting calls into question the efficacy of the conceptual modeling. Castellanos et al. (2020) address this problem by providing theoretical foundations 'rooted in psychology research supporting the existence and importance of special classes that are termed basic-level categories.' This theory is based on identifying basic classes in a domain.

Another theory is based on the assumption that the diversity of enterprise design stakeholders leads to the use of different modeling languages (even within a domain) due to different perspectives, personal preferences, and so on. Gray et al. (2020) therefore state that 'heterogeneity in enterprise design stakeholders generally demand for transformations between conceptual modeling languages.' Thus, their theory is based on the goal to use the strengths of different modeling languages. The main feature of the developed tool based on the *ADOxx* platform is that 'it addresses stakeholder heterogeneity by enabling transformation of a DEMO organization construction diagram (OCD) into a BPMN collaboration diagram.'

According to Rudnitckaia et al. (2019), 'the high number of participants,' i. e., the amount of stakeholders, is one of the reasons for the 'high degree of [...] diversity' of processes.

Regarding the 'variety of stakeholders that need to interpret [...] models,' Corradini et al. (2018) state that understandability of business process models is 'a fundamental quality that need to be taken into particular account by modelers.' Stakeholders are, for example, business analysts, software analysts/developers, and (other) employees. Guidelines that can help modelers to improve the understandability are proposed.

In their paper on guiding process modelers, Gassen et al. (2015) deal with 'the diversity of expertise from novice modelers to expert designers' as a challenge of process modeling. The expertise differences are discussed based on the cognitive load theory. To increase the quality of process models, they propose introducing different sorts of guidance during modeling and suggest that 'guidance on reworking models needs to take different levels of expertise into account.' Koschmider et al. (2015) also state that 'unexperienced modelers [...] do not share the same expertise as professional modelers [...] in terms of applying modeling guidelines and the correct use of the modeling language.' With their approach, they want to make 'modeling accessible for a larger audience.'

Business process models 'often grow heterogeneously with the company and are thus often terminologically divers and complex.' According to Havel et al. (2014), this is due to terminological diversity that 'originates from the fact that natural language allows an issue to be described in a large variety of ways especially when many modellers are involved.' Thus, the involvement of many stakeholders leads to terminologically ambiguous conceptual models. This heterogeneity is a problem when conceptual models are subject to, for example, model analysis techniques. Therefore, they propose the use of naming conventions to ensure the model quality. A similar theory is described by Becker et al. (2009). Fengel and Rebstock (2009) say that 'in case no predefined vocabulary or rules [...] are in place' to label model elements following the business terminology, 'terms are chosen individually on a case-by-case-basis.' Models, therefore, are 'often semantically heterogeneous concerning the domain language.' Thus, 'when comparing or integrating models, support through a consensus terminology' is necessary.

Tarkkanen (2009) states that the variety of work practices, e. g.,, due to vague work descriptions, on the same business process 'can have unexpected and harmful social and economic consequences.' The involvement of 'unexpectedly acting workers as well as those who act according to guidelines, constitute together an occurrence of non-uniformity.' The given theory concludes that modeling practices have to adapt to non-uniform work activity due to different workers.

Van Gils et al. (2022) place conceptual modeling in the context of digital transformations. They find that conceptual models are critical to understanding the key elements of the existing/desired enterprises and their context. In their paper, they address two challenges of conceptual modeling in digital transformations: (1) management of the diversity of the involved participants, and (2) the inclusion of non-experts in modeling. They explained some of the background for each challenge and outlined the key directions for future research.

Bonorden et al. (2022) use the *Decision Model and Notation* (DMN) to model decisions regarding the architecture of *Federated Digital Twins* (FDTs), i. e., digital twins that are distributed among different systems. A main aspect for them to reason about FDTs are the different stakeholders involved. In the specific use case described, there are stakeholders with different roles such as plant operators, component providers, and plant manufacturers involved. Each of the stakeholders comes with their own concerns for their realm, e. g., about sharing relevant but critical data. The authors propose to use DMN Decision Requirements Diagrams and decision tables to formalize the decisions for the partition of digital twins.

Requirement Differences

Different requirements lead to diversity in (business) processes and conceptual modeling techniques. For example, Aysolmaz et al. (2019) state that different process variants result from 'divergent business requirements.' The diversity in business contexts can lead to variants of the same process that 'may emerge in multiple cases in the same organization.' The work aims to help to select an approach on how to manage variability.

Karagiannis et al. (2019) propose a meta modeling approach that aims to support the engineering of modeling method requirements. Those requirements are the motivation for the realization of (conceptual) modeling methods and tools. Furthermore, attention is drawn to the agility in modeling method engineering: 'agility in modeling methods means enabling responsiveness to requirements that may be situational, domain-specific, enterprise-specific, evolving, etc.' (Karagiannis 2018). The 'wide diversity of requirements for conceptual modeling' and the associated need for conceptual modeling agility are manifested not only on the model contents level but also on the modeling method level.

Another theory in the field of collaborating enterprises ('borderless organizations') deals with 'processes [that] are dynamically transformed and integrated with the ones of their partners' (Bianchini et al. 2009). The collaborative design of business processes brings requirements to manage 'heterogeneity in process descriptions which are distributed across different enterprise systems.' Bouchbout et al. (2010) propose a generic meta model to support inter-organizational business processes because 'modeling business processes that span multiple organizations involves new challenges, mainly the ability to cope with autonomy, privacy, heterogeneity.' Malekan and Afsarmanesh (2013) analyze and evaluate business process modeling languages for the use in collaborative networks and propose a new evaluation approach for collaborative business process integration.

Brocke et al. (2008) point out specific requirements for enterprise content modeling. They state that 'these requirements are not met by conventional modeling languages.' Therefore, they propose the *Enterprise Content Modeling Language* (ECML), as a modeling language for the specific use case of organizations dealing with digital content.

Hofferer (2007) states that user and business requirements 'demand an integrated view on the multitude of available information resources that are processed in a variety of heterogeneous information systems.' His paper focuses on the semantic interoperability of business process models using meta modeling and ontologies.

Lukyanenko et al. (2022) find that, outside traditional database design and process modeling applications, conceptual modeling concepts have not been widely adopted. They state, while there is an increase of 'diversity and sophistication of information technologies', conceptual modeling techniques have not kept pace with technological developments. In their paper, they call for rethinking the nature of conceptual modeling and propose 'a new paradigm in which conceptual modeling is an activity of designing and managing AI-powered conceptual modeling systems (CMS)'.

With very similar motivation, Recker et al. (2021) states that research on conceptual modeling needs to be revised due to the emerging demands of a digital world. They develop a new theoretical framework that employs conceptual modeling scripts as mediators between physical and digital realities. In their paper, the authors question existing assumptions about conceptual modeling, since these are increasingly inconsistent with the IS landscape. They formulate new assumptions and research questions, and also note the need to develop new methods of conceptual modeling that accommodate the new assumptions.

Model Heterogeneity

Model heterogeneity has emerged as a major topic in our literature research. It deals with the diversity of conceptual models and the need for integration of such models. There are various theories about model heterogeneity from which a very popular one is the variety of methods and tools in modeling: 'each one presents several aspects and concepts in relation with the process' (Akkiyat and Souissi 2019). Every process modeling approach has its strengths and weaknesses (Pastor et al. 2013; Malekan and Afsarmanesh 2013), while every application domain has different requirements (see Requirement Differences). Thus, the use of different approaches leads to a wide diversity of models. This assumption is supported by the theory that due to the 'diversity and heterogeneity of real-world systems, makes it impossible to naturally model them only with existing modeling languages' (Craciunean 2019). See Method and Tool Variety for a summary of the theories on the variety of methods and tools.

Another theory about model heterogeneity has already been described in Stakeholder Diversity: The involvement of various stakeholders, e. g., process modelers or business analysts, leads to notational heterogeneity and also terminological diversity (e. g., based on the level of expertise of process modelers). The variety of drivers for business process modeling initiatives lead to models 'that differ in content due to serving different purposes' (Weidlich et al. 2009a).

Model heterogeneity can be explained particularly well using the semiotics, which we regard as the syntax, semantics, and pragmatics of models, as 'language-based models use a language as a carrier.' Zivkovic et al. (2007) provide a classification of model heterogeneity taking into account the syntactic, structural, and semantic discrepancies of models. While they focus on the heterogeneity on the meta model level, their classification can be applied to each modeling hierarchy level. The syntax of a language (sign system) describes the rules according to which the language constructs (signs in the sign system) are formed.

According to Zivkovic et al. (2007), syntactical heterogeneity 'represents the difference in formats intended for the serialization of meta models.' Therefore, the heterogeneity in terms of syntax is based on different formats and paradigms with regard to the formal structure of various meta models.

Representational and schematic heterogeneity are subsumed as structural heterogeneity. Meta models are '*represented* using different meta modeling languages, i. e., meta-meta models (meta^2 models)' (Zivkovic et al. 2007). Each of them are different in its expressive power regarding modeling primitives. Furthermore, there are reasons for *schematic* conflicts, even when agreed on a common meta^2 model. This is the case when meta models have different conceptual schemas, i. e., when the same concepts being described have been modeled differently. This divergence mainly happens due to the following situations: (1) equal concepts are modeled with different modeling primitives, or (2) requal concepts are modeled with different number of primitives.

Semantic heterogeneity deals with different meanings of meta model concepts, i. e., concepts from different meta models 'can use the same linguistic terms to describe different concepts or use different terms to describe the same concept etc.' (Zivkovic et al. 2007). According to Klein (2001), semantic mismatches between meta models can be distinguished in three possible inter-relations between them: (1) semantic equivalence: concepts hold the same meaning, even if different terms are used; (2) semantic relation: concepts are related through semantic relation types ('is-a,' 'has-a,' 'type-of,' and 'associate'); (3) non-relation:

concepts are unrelated, i. e., completely orthogonal in the meaning. Semantic heterogeneity is one of the most researched conceptual modeling problems and an approach specifically related to semantic integration of process models (Lin et al. 2006; Lin and Strasunskas 2005).

Syntactic, structural, and semantic heterogeneity can be subsumed under the term 'information heterogeneity,' (in contrast to system heterogeneity) according to Ouksel and Sheth (1999).

Business process models can differ in terms of their quality. The quality criteria can also be divided into syntax, semantics, and pragmatics. In his work, Laue (2010) presented a metric that describes the degree of unstructuredness of business process models. Based on this, various forms of unstructuredness were examined, and a catalog of patterns was designed that describes problems in EPCs. Finally, a method based on logical programming was developed that finds instances of such patterns in EPCs. The work shows that process models differ in their level of quality, which is also perceived as a kind of diversity.

With their paper 'The triptych of conceptual modeling', Mayr and Thalheim (2021) provide a framework for a better understanding of conceptual modeling. In this work, they thoroughly discuss the properties of conceptual modeling. Based on these considerations, they propose a paradigm for conceptual modeling in the form of a triptych that consists of three wings representing (1) the 'encyclopaedic' dimension, (2) the linguistic dimension, and (3) the model dimension. Consequently, conceptual models can be considered on a separation of (1) language, (2) knowledge, personal perception, and (3) modeling as a separate activity.

Process Variants

Process variants may be caused by 'various factors such as differences in delivered products, customer types, and divergent business requirements in countries' (Aysolmaz et al. 2019). Different and evolving requirements depending on the business context have been discussed in Requirement Differences. Adopting their processes to evolving business requirements is an important topic for organizations employing Business Process Management (BPM). By applying even slightly different behavior to existing processes leads to process complexity over time.

According to Rudnitckaia et al. (2019), the complexity of a domain leads to processes 'characterized by a high degree of complexity, dynamics, diversity, and human-centricity.' Reasons are product complexity, unexpected events, or the high amount of process participants. Different expertise or working practices of stakeholders can also affect process diversity. We already subsumed the influence of various stakeholders in the modeling context under the term Stakeholder Diversity.

Bjeković et al. (2014) state that if the scope of a fixed modeling language is not 'in line with the context in which it will be used, [...] purpose-specific "variants" will emerge to compensate for the missing suitability.' They also mention the need for 'modular organization, [...] situational adaptability and evolution' of a language as a 'different approach towards the standardisation of modeling languages.'

Process diversity leads to complexity in organizations. The increasing complexity of the processes leads to many process variants and also affects process modeling. Usually, process variants have to be kept in separate process models, leading to redundancy and difficult maintainability. According to Aysolmaz et al. (2016) and Aysolmaz et al. (2019), process variant modeling approaches such as the Decomposition Driven or Provop method are suitable to manage process diversity in a business context.

In order to deal with terminological and naming diversity, Havel et al. (2014), as well as Becker et al. (2009), developed solutions that ensure a standardized terminology and therefore lead to unambiguous conceptual models.

Weidlich et al. (2009b) say that the 'variety of drivers for business process modeling initiatives [...] results in multiple models that overlap in content due to serving different purposes.' To enable the propagation of changes between process models, they proposed an approach that can handle changes in pairs of models using behavioral profiles.

Adamo et al. (2018) state that 'mainstream business process modeling languages' only seem to focus on the 'activity execution order in the control flow.' In their paper, they investigate the role of 'another kind of fundamental relationship between activities' called ontological dependence and introduce a language to express it. This shows the potential diversity of processes that most modeling languages do not even capture.

Formalization and standardization efforts such as clearly defined patterns (Van der Aalst et al. 2005; Rinderle-Ma et al. 2008), guidelines (Havel et al. 2014), or conventions (Becker et al. 2009) are proposed in order to master the variety of processes.

Method and Tool Variety

The variety of modeling methods and tools is another main topic regarding diversity in conceptual modeling. Different objectives such as BPM (e. g., with the goals of business process (re)engineering, quality management, the introduction of *Enterprise Resource Planning* (ERP) systems, etc.) are the reason why a wide variety of modeling methods and tools are used in practice (Junginger et al. 2000). Many years of 'research in conceptual modeling has yielded a wealth of concepts and notations' (Wieringa 2011). Also, each domain has its own specific requirements that require appropriate solutions.

As modeling has become more important, the number of modeling tools and methods has increased, which leads to fragmentation in the field of enterprise modeling (Winter and Blaschke 2018). The main disadvantages are the lack of standardization and the variety of graphical representations available (Rosenkranz 2006). The multitude of tools and methods makes it difficult to choose a suitable solution (Araújo M. B. and Gonçalves R. F. 2016).

We already discussed some reasons for method and tool variety in this section (see Stakeholder Diversity and Requirement Differences): Stakeholders have different views on models and thus are in need of respective methods and tools. The same applies to different business contexts with individual requirements. Furthermore, there are several theories that a single method or tool is insufficient to meet the requirements (Al-Fedaghi 2018; Riggio et al. 2005).

Not only the diversity of the modeling languages is a topic to be investigated, but also the diversity and differences of the elements used in these different modeling languages. According to Kunze et al. (2011), 'BPMN models expose a higher diversity than EPCs in terms of construct heterogeneity.' However, they investigated that 'most modelers resort to a rather limited set of vocabulary, with simple activity sequences at the very core,' suggesting a 'true core set of relevant modeling concepts.'

The use of heterogeneous methods and tools led to various theories to deal with tool and method incompatibility. With Model Heterogeneity, we have already addressed the information heterogeneity that comes with the diversity of modeling methods. Regarding the variety of tools, the second category of interoperability problems comes into play, namely 'system heterogeneity' (Ouksel and Sheth 1999). System heterogeneity 'maps to the diversity of available access services, mechanisms, persistency services, and implementation technologies of each platform' (Kühn and Murzek 2006). Involved providers 'must agree on necessary interfaces, [...] standardized "protocols" [and] file formats' (Kühn and Murzek 2006).

Riggio et al. (2005) summarize five core motivations for interoperability among various platforms:

1. A single tool cannot be used in the whole Information System's (IS) development life cycle, e. g., from strategic planning to maintenance;
2. the components of a single integrated modeling environment might not be suitable for each phase of the system life cycle;
3. a single tool might not be able to satisfy all requirements, such that an organization might decide to use different methods or development processes;
4. some long-lasting projects might outlive the End-of-life (EOL) of most tools; tools might also not be backward-compatible for such a long time;
5. in projects spread over different companies, it is improbable that all participants will use the same set of tools.

Karagiannis (2018) states that the 'wide diversity of requirements for conceptual modeling [is] typically satisfied by Design Science artefacts such as modeling methods.' One of the 'open challenges' is the development of 'interoperability mechanisms at meta^{2}model level between the popular meta modeling platforms.'

In conclusion, it can be stated that there are good reasons for the diversity of methods and tools, while at the same time the need for interoperability of methods and tools is shown.

20.4.2 Methodologies Regarding Diversity in Conceptual Modeling

The second research question addresses the methodologies, i. e., research approaches regarding diversity in modeling. The previous section has shown different views on diversity in modeling. While the diversity is generally a positive development, it brings with it various challenges. Thus, we outline four main approaches to deal with diversity in modeling.

Model Transformation

Model transformation deals with translating models into a different notation that is more suitable for the intended purpose. According to Delcambre et al. (2018), 'conceptual model translation is when a conceptual model in one modeling language is translated into a conceptual model in another modeling language.' Johannsen (2007) distinguishes three forms of model transformation:

1. Transformation between graphically supported modeling languages,
2. transformation of graphical modeling languages into an exchange format, and
3. transformation between different interchange formats.

The transformation of models has been extensively researched, particularly with regard to business process models, while most approaches focus on XML-based transformation (Mendling and Nüttgens 2003; Mendling and Nüttgens 2002; Mendling and Müller 2003; Mendling and Nüttgens 2004a; Mendling and Nüttgens 2004b; Mendling and Nüttgens

2004c; Mendling et al. 2005; Mendling and Nüttgens 2006; Vanderhaeghen et al. 2005b; Vanderhaeghen et al. 2005a). An overview of respective XML-based interchange formats is provided by Mendling et al. (2004). Special attention was paid to the transformation and exchange of (extended) EPCs, BPMN, and UML models, being the most popular process modeling languages (Kožíšek and Vrana 2017). Models in XML-based interchange formats are often transformed by using Extensible Stylesheet Language Transformation (XSLT) scripts.

Model transformation can be used to exchange models between tools in order to use them in a tool with the desired functionalities. One example is the transformation of business process models into Petri nets (Van der Aalst 2015; Koschmider et al. 2015; Ouyang et al. 2007), e. g., for the purpose of model analysis. According to Murzek and Kramler (2007), there is a need for model transformation 'due to company mergers, acquisition and business to business interoperability.' In his paper on 'modeling needs in the BPM consulting process,' Reinheimer (2011) says that 'every visualization has got a key strength, a reason, why it actually exists.' While other methods need 'to be able to accept EPCs as an input,' he points out that transformation can be subject to a lack or loss of information.

Enterprises may intend to move from EPC notation to BPMN 'due to the external or internal incentives' (Levina 2012). While the focus of many research work was often put on model syntax when transforming between process modeling languages, Levina (2012) examines the transformation from EPC to BPMN with regard to the integrity of process content. Transferring business process models from EPC notation in BPMN 'does not significantly affect its information content, iff the transformation is performed according to the original model.'

Kappel et al. (2011) state that while model transformation is 'the first choice for realizing model exchange between heterogeneous tools,' it also 'forces the developer to define model transformation code again and again for certain recurring integration problems.' Therefore, they propose 'a framework for model-based tool integration which is based on well-established conceptual modeling techniques.'

Lee et al. (1998) have developed 'an interchange format to help automatically exchange process descriptions among a wide variety of business process modeling and support systems.' They introduced the so-called PIF format so that 'instead of having to write ad hoc translators for each pair of such systems, each system will only need to have a single translator for converting process descriptions.'

Karagiannis and Höfferer (2006) state that 'to bring together models that have been realized using different meta models,' the 'mapping should [...] be realized on the [...] meta^2-layer that acts as "translator" between meta models.' They refer to Zaniolo and Melkanoff (1982) who say that 'a direct mapping between different models is a formidable problem' and propose 'using a meta[^{2}] model which is easily mapped into other models.'

Meta Modeling and Ontologies

Other approaches to deal with interoperability among modeling languages are (meta-)meta modeling and ontologies. These approaches are the methods of choice in conceptual modeling regarding the integration of different meta models and research on (semantic) model heterogeneity. The set of language elements of a modeling language is nothing more than a meta model, often referred to as *model of a model*. Thus, meta modeling is the process of designing such meta models. A meta model specifies the syntax and semantics of a modeling language. Karagiannis and Höfferer (2006) state that 'to integrate artifacts that are already described by a meta model, at least a [common] meta^2 model is needed,' while Kühn and Murzek (2006) show the interoperability problems that may arise, and how integrated meta models can be realized.

Well-known meta modeling approaches are *Enterprise Model Integration* (EMI), addressing the 'diversity of models and modeling languages' (Kühn et al. 2003) and providing patterns for meta model integration, *Model Integrated Computing* (MIC), including the MIC tool architecture (Ledeczi et al. 2001), or standardized modeling languages like UML with its meta model called *Meta-Object Facility* (MOF) (Object Management Group (OMG) 2016). The *Business Process Definition Metamodel* (BPDM) is a meta model for different business process modeling languages (Object Management Group (OMG) 2008), especially for BPMN. While *BPMN Diagram Interchange* (BPMN DI) is also a meta model and schema to facilitate the interchange of BPMN diagrams between tools, BPDM is not part of the BPMN specification document but has its own OMG specification.

Kühn et al. (1999) present meta modeling concepts for BPM based on the meta^2 model of the BPM tool *ADONIS* (Junginger et al. 2000) and the definition of the semantics of process modeling techniques. Three years later, Karagiannis and Kühn (2002) introduced a generic architecture for meta modeling platforms as 'environments is that the formalism of modeling – the meta model – can be freely defined.'

Karagiannis and Visic (2011) propose a meta modeling approach that 'is considered to provide the required concepts and mechanisms to combine different modeling methods' by introducing the *Next Generation Modeling Framework* (NGMF). It supports Hybrid Method Engineering – an approach based on meta modeling that 'takes into consideration the different perspectives of modeling languages.' The underlying concept called 'hybrid modeling' addresses the integration of modeling methods and the merging of different modeling concepts. This approach is used by the *Next-Generation Enterprise Modeling* (NEMO)[1] project within the Open Models Laboratory at the University of Vienna. Fengel and Rebstock (2010) propose a 'bridge ontology' for 'linking models in differing modeling languages as well as different model types.'

In 2018, Karagiannis proposed an *Agile Modeling Method Engineering* (AMME) framework (Karagiannis 2018; Karagiannis 2022), which is an agile development process for modeling methods and addresses the diversity of requirements for conceptual modeling. To fill the gap in this development model, the *CoChaCo* method for modeling method requirements engineering was introduced.

Another contribution using meta modeling approaches addressing the integration of conceptual modeling languages is *SeMFIS*, 'a flexible engineering platform for semantic annotations of conceptual models [...] for dynamically extending the semantic representation and semantic analysis scope of conceptual modeling languages' (Fill 2017). It can be applied 'to the large variety of [...] modeling methods' because of the implementation based on the ADOxx meta modeling platform (Fill and Karagiannis 2013).

To automate the interoperability of conceptual models, Pastor et al. (2013) propose a model-driven interoperability process based on existing modeling technologies and standards such as EMOF (Essential-MOF) and UML profiles.
More solutions addressing the transformation between different models are proposed from Jarke et al. (2009) by describing 'a logic-based conceptual modeling and model management

1 https://nemo.omilab.org/nemo/

approach' by employing a 'generic meta model to facilitate mappings and transformations between heterogeneous model representations,' or Murzek and Kramler (2007) proposing a 'model morphing approach' for horizontal transformations of business process models based on the 'idea [...] to create an integrated meta model containing all concepts of the languages of a given domain.' In 2005, Grangel et al. (2005) introduced a 'meta model and a corresponding methodology that enable enterprises to exchange their enterprise models, despite the fact that they use different Enterprise Modeling Tools.' With the *Fundamental Conceptual Modeling Languages* (FCML) method, Karagiannis et al. (2016) present a 'meta modeling approach for the hybridization of BPMN, ER, EPC, UML and Petri Nets within a single modeling method' and the corresponding implementation *Bee-Up* as a proof-of-concept based on the meta^2 model provided by ADOxx (Fill and Karagiannis 2013).

List and Korherr (2006) propose a generic meta model 'that captures a wide range of process concepts.' This meta model is the basis for an evaluation framework and enables the evaluation of seven business process modeling languages (UML AD, BPDM, BPMN, EPC, IDEF3, Petri nets, and RAD). Heidari et al. (2013) also propose 'a language independent abstraction of seven mainstream BPMLs' concepts' (EPC, IDEF0, RAD, BPMN, IDEF3, UML AD, and SADT) in the form of a unified meta^2 model.

The work of Hofferer (2007) focuses on the interoperability of business processes using the 'synergy effect' of meta models and ontologies. While meta models are used as the 'provider of the syntax of a modeling language,' ontologies 'describe both the semantics of the modeling language constructs as well as the semantics of model instances.'

Conceptual models provide an abstract and simplified view of things in the real world using conceptualization, while an explicit specification of a conceptualization is called ontology (Guarino 1998). In conceptual modeling, ontologies are used to address challenges in semantic integration and the so-called understandibility problem, sometimes referred to as the 'tower of Babel problem,' for which Olivé (2017) proposes a *Universal Ontology* (UO) as a solution. 'As a basis for integrating many modeling languages and standards in different domains (e. g., UML, TOGAF, ArchiMate, RM-ODP, TROPOS/i*, AORML, ARIS, BPMN),' Guizzardi et al. (2015) present the *Unified Foundational Ontology* (UFO) as part of their research regarding ontological foundations for conceptual modeling. The research in terms of an UFO led to the 'development of an Ontology-Driven Conceptual Modeling language dubbed OntoUML, reflecting the ontological micro-theories comprising UFO' called *OntoUML* (Guizzardi et al. 2018).

The *Business Process Modeling Ontology* (BPMO) is an approach for modeling business processes at the semantic level, and part of the SUPER (Semantics Utilised for Process Management within and between Enterprises) project (Cabral et al. 2009). BPMO enables 'seamless interoperation, querying, sharing, mediation and translation of business processes.' In the context of the SUPER project, ontologies for BPMN, EPC, and BPEL as well as corresponding translators to BPMO are provided. Furthermore, an ontological representation for *Semantic BPEL* (sBPEL), and ontologies named BPMO2sBPEL and BPEL2BPMO, enabling automated translation between respective representations of business processes, are described (Norton et al. 2009).

Thomas and Fellmann (2007) provide an approach for semantic annotation of EPCs – referred to as *Semantic EPC* (sEPC) – *'by linking model elements from semi-formal languages with concepts from formal ontologies and thus, receiving a formal semantic.'*

An approach for the semantic verification of process models based on ontologies, rules, and reasoning is proposed by Fellmann et al. (2010). Model elements can be classified by using facts automatically derived from the ontology. Based on this classification of model elements, *'abstract semantic verification rules are used to decide whether the model conforms to rules and regulations.'*

MetaMorph (Döller and Karagiannis 2021; Döller et al. 2023) is a formalism for conceptual modeling languages. Döller and Karagiannis state that formalization is a crucial part in the life cycle of developing a modeling method. Their formalism provides a 'unique, unambiguous way of specifying the syntax and semantics' of any modeling language, i. e., it is 'generic and open to capture any domain and any functionality'. Their formalism considers the concepts of the meta^2 models of six established metamodeling platforms.

Standardization

Standardization is another approach to deal with the diversity in modeling. At first, there is the standardization of methods and tools.

Mendling et al. (2005) state that 'although standardization has been discussed for more than ten years, the lack of a commonly accepted interchange format is still the main encumbrance to business process management.' Various standardization efforts aim to solve the problem of 'a missing de facto standard for BPM.'

Some standardization efforts were carried out in (business process) modeling that can be specifically attributed to the OMG (Object Management Group), for which the UML (Object Management Group (OMG) 2017) is a good example. While the UML is the standard modeling language to support the software development process, its activity diagram is also suitable for modeling business processes. The UML meta model MOF (Object Management Group (OMG) 2016) is intended to bridge the gap between different meta models. Thus, if two different meta models are MOF-compliant, respective models are interoperable and could, for example, reside in the same model repository.

While the UML is the quasi-standard in software systems modeling, this applies to BPMN in terms of business process modeling with its rich specification and extensive ecosystem. Furthermore, both languages have been standardized by the ISO. As stated in Meta Modeling and Ontologies, the BPMN specification contains a meta model and schema. To build standard-compliant BPMN tools, the *BPMN Model Interchange Working Group* (BPMN MIWG), a joint effort of several vendors, was set up (Object Management Group (OMG) 2021).

With *XML Metadata Interchange* (XMI), the OMG also provides a standard for exchanging metadata information via XML (Object Management Group (OMG) 2015). It can be used for any meta model that is expressable in MOF.

Since the focus regarding business process modeling languages shifts towards standardized modeling languages such as BPMN, Karhof et al. (2016) identified and evaluated BPM tools that implement the EPC modeling language. Thus, they derived consensus about important language constructs in order to prepare EPC standardization. The evaluation showed that tools implement EPCs in very different ways, especially by neglecting the syntax, semantics, and pragmatics of the EPC language.

Standardized modeling languages, e. g., BPMN, can be extended to provide solutions for specific use cases (Yousfi et al. 2016; Amdah and Anwar 2019; Ben Hassen et al. 2019). Al-Fedaghi (2018) proposes a new approach based on BPMN that 'includes a diagrammatic modeling language to implement [the] combination of "simple business process modeling notation + BPMN + Petri nets"' called 'Flowthing Machine' (FM).

Modeling Techniques Embracing Diversity

In the following, we address specific modeling techniques that are, for example, aware of different aspects of a domain, or that reflect the inherent variety of modeling or the models (to be) created. Of course, the following techniques are also subject to a theory, which is why they could certainly also be assigned to the previous Section 20.4.1. However, we are focusing here on methods that are already more concrete or implemented.

Frank (2014) presents a method for multi-perspective enterprise modeling (MEMO) and a corresponding (meta) modeling environment. Multi-perspective enterprise modeling represents a holistic perspective on an enterprise with three generic perspectives – strategy, organization, and information system. Each can be further detailed into four different aspects – resource, structure, process, and goal. The work shows the multitude of perspectives that can be taken into account regarding enterprise models. MEMO is composed of domain-specific modeling languages (DSMLs) – namely MEMO GoalML (Goal Modeling Language) (Bock and Frank 2016), MEMO DecisionML (Decision Modeling Language) (Bock et al. 2014), MEMO MetricML (Performance Measurement Modeling Language) (Strecker et al. 2012), MEMO OrgML (Organization Modeling Language) (Frank 2011a; Frank 2011b), and MEMO ITML (IT Modeling Language) (Frank et al. 2021) – to describe and interrelate different aspects of an enterprise. MEMO also comes with an associated tooling: MEMO4ADO (Bock et al. 2022) is a modeling tool that implements a subset of MEMO and encompasses the listed DSMLs. It is implemented using the meta modeling environment ADOxx.

Another enterprise modeling method that takes different perspectives of an enterprise into account is 4EM – 'for enterprise modeling' (Lantow et al. 2022). The 4EM method consists of three core elements: a defined procedure for modeling using a fixed notation, project organization and roles, and stakeholder participation. These elements are supported by appropriate tools and resources. 4EM comes with a set of sub-models and perspectives, such as goals, business rules, concepts, business processes, actors and resources, products and services, as well as technical components of information systems.

Fonseca et al. (2021) state that 'in many important subject domains, there are central real-world phenomena that span across multiple classification levels'. With their work on 'Multi-Level Modeling' they address a modeling approach involving multiple levels of classification. They emphasize that, e. g., 'many [existing] approaches do not support domain relations between elements of different classification levels'. Furthermore, they state that other methods had severe restrictions on the arrangement of elements in strictly stratified levels, effectively preventing the representation of real-world domain models. To adress these issues, they introduce a new multi-level conceptual modeling language called ML2 (Multi-Level Modeling Language). They compare their approach with other multi-level representation approaches such as UML. For example, while ML2 implements all eight features considered, UML only (partially) supports four of them.

In their paper, Schützenmeier et al. (2021) describe the differences between imperative (procedural) and declarative process modeling languages. They discuss the advantages and disadvantages of these approaches using the examples BPMN (procedural) and MP-Declare (declarative). Based on these considerations, they develop a new concept for a hybrid modeling language based on BPMN. While their approach leads to better readable models, the authors also acknowledge that there is still a need for further elaboration.

With their modeling approach Heraklit, Fettke and Reisig (2022) introduce 'The Next Generation of Enterprise Modeling'. They state that they disagree with Dijkstra's view that there is a 'firewall' between the technological and the applied face of computer science. With regard to modeling, they say that it is an activity that should allow 'a seamless transition between formally and informally given or asserted facts of a computer-integrated system'. From their point of view, technology and applications can be connected by using shared models built on the same principles. Their Petri net-based approach promises a smooth transition between the two sides, which can be accomplished through a framework that includes architecture, statics, and dynamics as the three pillars of modeling computer-integrated systems.

Frerichs and Nüttgens (2022) propose a reference architecture called *Enterprise Digital Twin (EDT) architecture* to support the creation of a Digital Twin of an Organization based on data from business information systems such as ERP systems. They state that one principle of an EDT should be the method- and tool-agnostic design, i. e., an EDT should 'at least provide reasonable generic interfaces so the respective tools can benefit from the data [...] available in the EDT'.

20.4.3 How Tools Address Diversity in Conceptual Modeling

Research question three focuses on the tools and how they address diversity in conceptual modeling. In the following, we will outline if and how tools in the field deal with it. There are various tools in conceptual modeling, while most of the tools in this literature review focus on (business) process modeling.

According to Nägele and Schreiner (2002), there are five types of BPM tools: Visualization tools, modeling tools, simulation tools, workflow management systems, and Computer-Aided Software Engineering (CASE) tools. Gadatsch (2020) distinguishes between (1) tools for modeling, analyzing, and designing processes (BPM tools) and (2) tools for control, automation, and automated analysis of processes such as workflow management systems, robotic process automation tools, and process mining tools.

Fellmann et al. (2018) conducted a survey of 47 participants with practical modeling skills: The respondents have used 'the ARIS platform (40.4%), MS Visio (31.9%), Signavio (14.9%), Adonis (8.5%), and various office products.'

Modeling Tools Supporting Standards and Intercompatibility

In the following, we take a brief look at some popular modeling tools supporting standard modeling methods and compatibility efforts. Of course, this is just a small excerpt based on our literature review since we are not aiming for a complete list of modeling tools in this work.

Taking a look at the technical product matrix of ARIS[2] (Software AG 2018a; Scheer 1998; Scheer 2002), we get an overview of the intercompatibility across various ARIS products. Also, the compatibility with specific non-ARIS environments such as SAP or MS SharePoint (respective 'extension packs' are required) is depicted. There are different types of compatibility, such as 'Web Service', 'Repository', 'Sync', 'Import', and 'Link' (Software AG 2018a).

Thus, the intercompatibility of ARIS products seems to be addressed in some ways. Especially ARIS Architect and ARIS Designer provide different data exchange interfaces

2 https://www.scheer-group.com/products-solutions/aris/

such as BPMN 2.0 export/import, DMN 1.1/1.2 import/export, Microsoft Visio (VSDX/VDX), ArchiMate® Model Exchange File Format import/export, and more. They also support various methods and standards such as EPC, VACD, BPMN 2.0, DMN 1.2, CXM, and IoT (Software AG 2018b; Software AG 2018a).

ARIS is not only the foundation for a wide range of products for process modeling, but it is also, from a conceptual point of view, a meta model for systematic process modeling (Software AG 2016).

The 'ARIS method,' also referred to as the 'ARIS house', follows an integrated approach with five different views on processes, each representing the model of a business process under a specific aspect (Scheer 1998; Scheer 2002; Software AG 2016).

While ARIS provides a holistic approach by offering a variety of views regarding the business processes of a company, the split in views reduces complexity and ensures a systematic approach (Software AG 2016). With ARIS Express[3], there is also a free (basic) version providing a set of nine model types.

Signavio Process Manager[4] is another established process modeling tool with support for multiple modeling languages (Signavio 2021b; Signavio 2021a): BPMN, EPC, DMN, *Value Chains*, *ArchiMate*, *Case Management Model and Notation* (CMMN), customer journey maps, organizational charts, BPMN choreography diagrams, BPMN conversation diagrams, UML use case diagrams, and UML class diagrams. Signavio offers BPMN 2.0 XML as a standard for process model interchange (import/export) and also allows importing models from other workspaces or third-party tools in the form of, for example, XPDL 2.1 diagrams, ARIS Markup Language (AML) diagrams, or Microsoft Visio diagrams. Like ARIS, the tool is also able to automatically transform EPC models into BPMN models. Standardization of process models is supported with in-built universal modeling conventions (Signavio 2021a).

While ADONIS[5] provides a complete illustration of BPMN 2.0 and a bidirectional BPMN DI and XPDL interface, it follows a modeling method-independent approach (BOC Group 2021): The meta model of ADONIS can be configured to suit any particular requirements by customizing the so-called Application Library, making it a meta modeling platform (Junginger et al. 2000). Thus, while other modeling tools provide one or more fixed modeling methods that can only be customized to a limited extent, the flexibility of ADONIS enables the use of methods for different application scenarios. In addition to the commercial editions of ADONIS, there is also a free cloud BPM tool (ADONIS Community[6]) where the customization of the BPM suite, method, and the adaptation of the meta model is not possible.

The *bflow** toolbox[7] is an open-source modeling tool for business process models based on the *Graphical Modeling Framework*[8] (GMF). The GMF runtime is an application framework for creating graphical editors using the *Eclipse Modeling Framework*[9] (EMF; see Meta Modeling Platforms) and the *Eclipse Graphical Editing Framework*[10] (GEF), providing a framework and end-user components related to graphical applications. bflow* supports modeling

3 https://www.ariscommunity.com/aris-express/
4 https://www.signavio.com/products/process-manager/
5 https://www.boc-group.com/en/adonis/
6 https://www.adonis-community.com
7 http://www.bflow.org
8 https://www.eclipse.org/modeling/gmp/
9 https://www.eclipse.org/modeling/emf/
10 https://www.eclipse.org/gef/

languages like EPCs, oEPCs, and BPMN (Bflow.org 2017). It also provides model validation for EPC and oEPC models. In order to achieve the best possible flexibility when exchanging models with other tools, the blow* toolbox provides functionalities for converting different formats. These conversions are implemented using import and export scripts. There are various existing implementations, e. g., for the import/export of Visio models, BPMN models, EPC models, models in AML, and more. bflow* also offers extensibility of the toolbox by providing an add-on interface (Bflow.org 2015).

Lack of Interoperability of Tools

Kern (2014) provides a study of interoperability between tools, investigating the degree of model exchange between meta modeling tools in further detail and looking for typical exchange approaches. By analyzing the import and export functionalities of different tools, he observes that the degree of interoperability can be considered low with values between 0.8 and 7.9 per cent (depending on the approach considered). In addition, it can be observed that most of the tools offers options for importing data and connecting data sources for import purposes. However, there is often little support for exporting or extracting data.

The standardization efforts of modeling tools help to increase the quality and uniformity in modeling. However, in terms of business process modeling, these are often limited exclusively to the BPMN ecosystem, which also continues to face challenges in terms of standardization. Dirndorfer et al. (2013) show that 'there are significant difficulties to do a cross-platform exchange of BPMN business process models in praxis.' All tested BPM tools were rated 'poor' or 'very poor / fail' concerning their import functionality of models from other tools. Furthermore, three of the eight tools were rated 'poor' for not offering BPMN 2.0 exports and providing 'only a little amount of alternative export possibilities,' from which some are only proprietary.

Martino et al. (2018) state that several tools 'for the design and execution of business processes have been proposed.' However, 'there is still a lack of integration with cloud services, mostly due to the high variety of currently available offers, which creates confusion among customers.'

Model Transformation Tools

TOOLBUS[11] provides a commercial solution for the continuous conversion of models to integrate them across different tools. According to the TOOLBUS website, their tool supports more than 70 tool formats and 'almost any notation,' e. g., BPMN, UML, SysML, ER, (extended) EPC, and more. TOOLBUS promises a tool-independent exchange of various modeling formats.

TransWare's *BPM-X converter*[12] is also a commercial model transformation tool. It translates models and diagrams between different tools, formats, and standards. Adapters to connect to tools and repositories are also offered. The 'universal translator' supports the 'exchange and translation of model and meta-data including the graphical information for all the different modeling languages (EPC, BPMN, ...) and frameworks (DoDAF, NAF, Zachmann, ArchiMate, ...).' The use cases mentioned are, for example, the integration or migration of tools.

11 https://www.toolbus.de

12 https://www.transwareag.com/products/middleware-for-universal-process-transformation.html

The *Atlas Transformation Language*[13] (ATL) is a programming language and toolkit for transforming models, i. e., for performing automatic model-to-model transformations. In ATL, a transformation essentially consists of a set of rules that convert individual elements of the original model into elements of the target model. The ATL Development Tools (ADT) were developed on the Eclipse platform and provide a set of convenient functionalities for the development of ATL transformations.

Meta Modeling Platforms

The ADOxx meta modeling platform[14] is the underlying development platform for the commercial tools developed by the BOC Group, e. g., ADONIS (Junginger et al. 2000), and can be used for implementing individual modeling methods (Fill and Karagiannis 2013; Efendioglu et al. 2016). The specification of meta models can be done in the ADOxx library language (ALL).

Meta modeling platforms are generally used to develop specific meta models, i. e., modeling languages (Karagiannis and Kühn 2002; Kühn and Murzek 2006). The Eclipse Modeling Framework (EMF) is a set of Eclipse plug-ins which can be used to model a data model and to generate code or other output based on this model, e. g., HTML. Like other meta modeling platforms, EMF distinguishes between the meta model and the actual model. The meta model, consisting of the *ecore* (information about the defined classes) and the *genmodel* (additional information for code generation) description files, can be developed in different ways: XMI, Java annotations, UML, or XML scheme.

MetaEdit+[15] is another (commercial) meta modeling platform, consisting of the MetaEdit+ Workbench (to define a modeling method) and the MetaEdit+ Modeler (as the modeling tool for the defined modeling methods). The MetaEdit+ Workbench follows an object-oriented modeling approach without having to produce a single line of code.

The Generic Modeling Environment[16] (GME) is a configurable toolset that supports the creation of domain-specific modeling and program synthesis environments (Ledeczi et al. 2001). This can be done through meta models specifying the modeling language of the application domain. Meta modeling in the GME is based on UML class diagrams to model the syntactic definitions and specifying the static semantics with constraints with the Object Constraint Language (OCL).

20.5 Discussion

With the help of the identified sources, various insights could be obtained. With regard to the research questions from Section 20.3, corresponding answers regarding the diversity in modeling could be extracted. We have to emphasize that at times this work has put a clear focus on enterprise and process modeling. The results of the work may therefore not cover all suitable approaches. With regard to the theories related to diversity in modeling, five main topics were identified. These deal with the reasons and effects of diversity in the field.

In the context of the first research question (RQ 1), we observed from the literature examined that the participation of individuals with different skill levels and the various

13 https://www.eclipse.org/atl/
14 https://www.adoxx.org
15 https://www.metacase.com/de/products.html
16 https://www.isis.vanderbilt.edu/projects/gme/

requirements in varying business contexts have a significant impact on heterogeneity in conceptual modeling. For example, borderless organizations (fueled by the increasing digitization) or the growing individualization due to special customer requests are reasons for the diversification of both processes and their models. Special use cases or business models and activities require specialized methods and tools for design, analysis, or optimization. This need also applies specifically to the area of conceptual modeling. All five identified diversity areas are intertwined, and we see potential in future work to explore specific connections and effects on each other.

Methodological approaches and solutions with respect to diversity in modeling are manifold. We have identified three main areas in conceptual modeling that classify existing approaches as part of the second research question (RQ 2). One area is the transformation of models, as this is the basis for the compatibility of different methods and tools. The intercompatibility of methods and systems is made possible in particular by extensive research in the areas of meta modeling and ontologies. Furthermore, standards and standardization efforts are a possible way to address the diversity in modeling. We see great potential for further research here, e. g., in the form of a holistic analysis and synthesis of existing methods.

The diversity in modeling is fueled by the existence of a wide variety of tools. This diversity was investigated in the third research question (RQ 3) and is expressed in the methodological and functional variety of the respective tools. Thus, it is helpful that many tools support certain standards and allow intercompatibility. However, the literature also shows that tool interoperability is not trivial, since even the implementation of standards is faced by challenges. In addition to the modeling tools, there are solutions that specifically address the individuality of tools and methods. The automated transformation of conceptual models enables integration across different tools. Extensive research in the field also enables tool-supported meta modeling to successfully integrate methods and tools.

The literature review has shown that there will always be tools and methods for specific purposes that require the integration of different tools. Despite efforts to standardize the methods, for example, through the broad support of BPMN, actual method and tool integration is not guaranteed. Manufacturers implement the standards differently, e. g., due to unspecified details. In addition, 'one size fits all' does not apply to conceptual modeling, which is clearly demonstrated by the multitude of methods and tools. Thus, the question quickly arises: How many 'standards' must tool manufacturers implement (uniformly) in order to enable method and tool interoperability? Although the standardization approaches are good for creating a common basis in modeling, it does not (fully) serve the purpose of interoperability. That is why we say that a completely decentralized approach has no prospect of success.

Therefore, the extensibility of modeling tools is a good approach. A suitable example at this point is the bflow* toolbox which inherently supports various modeling languages and exchange formats. Still, it is also open for extensions and the integration of external tools. The fact that it is open-source is also an advantage, as a large number of people can further develop the tool in a participatory manner, but also developed solutions are available to the community. So far, however, the solution offered has not been practice-oriented enough. The tool already provides support for various formats – but it does not really integrate common and state-of-the-art tools. For this, not only the specifics in the implementation of methods in the respective tools would have to be addressed, but also the peculiarities of the tools independently of the methods (e. g., APIs, data structures, etc.) would have to be examined. The capabilities of the commercial tools are difficult to assess, as they are often hidden behind a 'wall of sales and marketing' and an evaluation is beyond the scope of this work. However, such an evaluation is another good starting point for future research.

Direct transformation of models is an appropriate way to show the feasibility of the transformation of a specific *model A* created with *modeling language B* into a *model C* according to the *modeling language D*. But, as Zaniolo and Melkanoff (1982) state, the 'direct mapping between different models is a formidable problem.' We agree that an approach based solely on direct mapping scales very poorly when it comes to enabling universal transformation (many-to-many). Research on the fields of meta modeling and ontologies has shown that these approaches are good enablers for the integration of models. Therefore, these approaches pave the way to methodological independence. The integration of data structures, as is common in database research, is also possible with these approaches. Thus, they address both information and system heterogeneity and provide opportunities for further research regarding the goal of method and tool-independent modeling.

20.6 What a Wonderful (Modeling) World

In this work, we conducted a literature review to show how diversity is expressed in conceptual modeling and which approaches exist to deal with this variety. For this purpose, three research questions, based on the three contribution types in DSR, were considered. The results of the literature analysis were evaluated, classified and discussed accordingly.

In summary, it can be said that there are many influencing factors for diversity in modeling – from different stakeholder perspectives to the various requirements in the business contexts, and the resulting variety of methods, tools, formats, or systems. Since the world, especially in companies, is very complex and therefore information and system heterogeneity will steadily increase – despite the multiple promising standardization efforts – it is not foreseeable that actual standards will ever prevail that effectively do justice to the conceptual diversity.

This shows the need for solutions for the interoperability of systems and tools and transformation between modeling languages and formats. To ensure that the modeling of information systems is not an end in itself, simple solutions are required to establish interoperability between different tools. As described in Section 20.5, a decentralized approach in which the various tools have to implement standards is not sustainable. Therefore, we propose a concept (based on a central instance) that enables method and tool-independent modeling (Frerichs et al. 2021).

Definition (Method- and tool-independence). *Method- and tool-independent modeling describes the ability to deal with the diversity in process-aware information systems and business process modeling tools and make these systems and tools (and the respective business process models) interoperable where applicable.*

While modeling methods and tools are still necessary, it no longer matters which ones are used since the resulting process models can be flexibly transformed. This enables the use of the best tool for the right purpose.

Our concept proposes an integrated method- and tool-independent architecture interconnecting *Process-Aware Information Systems* (PAIS) such as ERP systems, *Workflow*

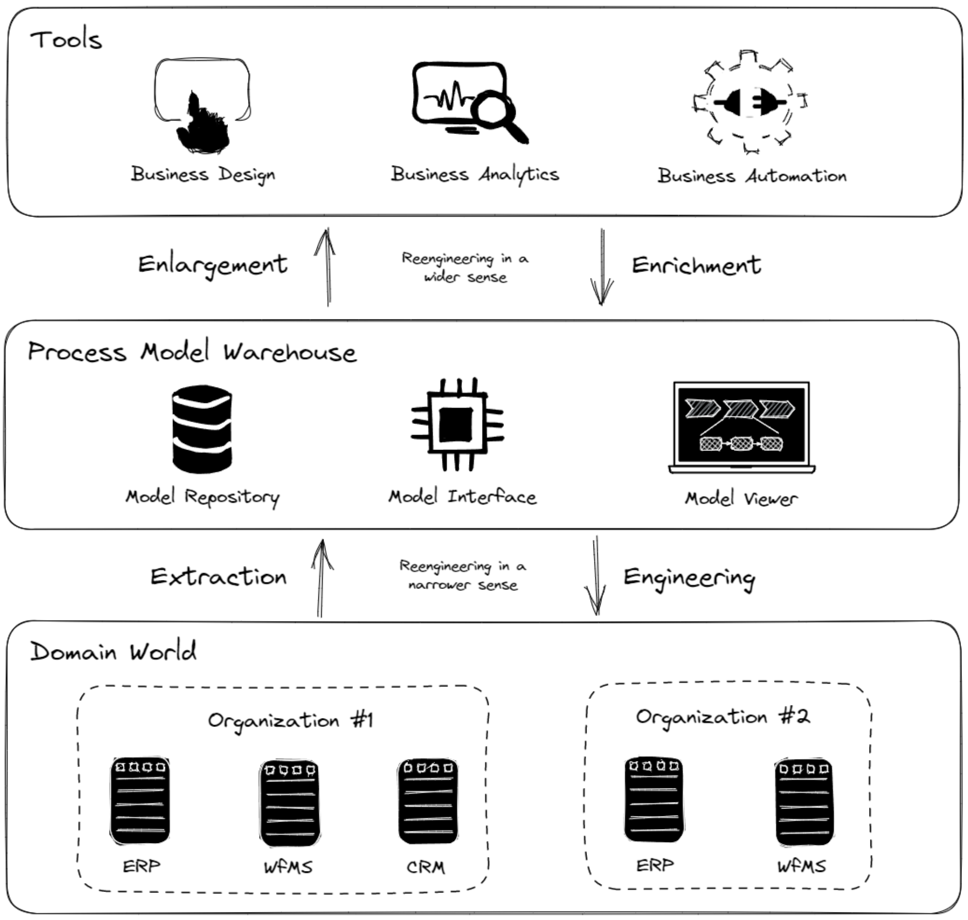


Figure 20.5: Process Model Warehouse concept

Management Systems (WfMS), *Customer Relationship Management Systems* (CRM), etc., and BPM tools, e. g., for business process design, analytics, monitoring, or automation.

Figure 20.5 shows an illustration of our concept. The aim is to be able to *extract* data from different sources of the 'domain world' (e. g., PAIS), transform it into a suitable format by using the approaches described in this work (transformation, meta modeling, and ontologies) and make it accessible for other applications (e. g., BPM tools). Also, we intend to import or load back data into corresponding systems (engineering). We refer to this step as *reengineering in a narrower sense*, as it relates to business process engineering. The *process model warehouse* is the core of our concept. It can be divided into the following three components: *model repository*, *model interface*, and *model viewer*. The model repository forms the common underlying basis for all process models that are located in the model warehouse and also serves to ensure the persistence of the models. The model interface serves as an interface to the outside world and must be able to handle the corresponding interfaces and formats of the tools. The data that is passed through the model interface must be transformed accordingly. The model viewer is an additional component that is used to obtain an overview of the process models in the process model warehouse and to manage them (e. g., viewing process models, deleting process models, etc.)

Reengineering in a wider sense means on the one hand making process data and models available for tools and thus expanding the data basis of these tools (enlargement). On the other hand, data can be fed back into the model warehouse at this level in order to expand the process models with additional data and perspectives (enrichment).

Thus, our approach enables the interaction of the relevant methods and tools of the BPM life cycle (Dumas et al. 2018) so that in each phase, the advantages of the existing BPM solutions can be leveraged. Furthermore, it also enables inter-organizational BPM.

Based on the plans described above, there are several challenges in implementing the concept. Beginning with the extraction of data from PAIS, it is not a trivial task to retrieve the relevant process data from these systems. The same applies to the connection of the various BPM tools. Coping with heterogeneous underlying data models and interfaces is a challenge in itself. This shows the need to deal with many peculiarities and characteristics of different tools and their respective formats and interfaces. A meta model to be designed will have to map the semantic and syntactic properties of the various modeling languages as well as the tool-specific features which have not been addressed sufficiently in existing approaches. We want to ensure the universality of the proposed solution, so extensibility is also provided. The connection of the many BPM tools is heavily dependent on the openness of the respective tools. In practice, it can be seen that especially newer software offers options for importing data and connecting data sources for import purposes. However, there is often little support for exporting or extracting data. Regardless of this, our concept focuses on supporting as many tools and methods as possible to achieve a high level of practical benefit and interoperability.

Instead of trying to propose the next (pseudo) standard, our approach tries to do the opposite by accepting the conceptual diversity in process modeling (and the tooling around it). Therefore, tools do not have to adapt to a given standard and we are finally able to use the best tool for the right purpose. We are aware of the fact that the implementation of the proposed artifact(s) is an ambitious task. With this very high-level concept we want to make a first small step towards the 'pious wish' of method- and tool-independent modeling but also increase awareness for this Grand Challenge in enterprise and business process modeling. This chapter is obviously thought-provoking and a bit reckless – but it is also a call to action on this long-held dream of universal interoperability. So that maybe one day we can say: 'What a wonderful modeling world!'

References

Adamo, G., Borgo, S., Di Francescomarino, C., Ghidini, C., Guarino, N. and Sanfilippo, E. M. (2018). 'Business Process Activity Relationships: Is There Anything Beyond Arrows?' In: *Business Process Management Forum. BPM 2018.* Springer, Cham.

Akkiyat, I. and Souissi, N. (2019). 'Building a Process Meta Model Extended for Cycles'. In: *Proceedings of the ArabWIC 6th Annual International Conference Research Track on - ArabWIC 2019.* New York, USA: ACM Press.

Amdah, L. and Anwar, A. (2019). 'BPMN4 collaboration: An Extension for collaborative Business Process'. In: *Advances in Science, Technology and Engineering Systems* 4.6.

Araújo M. B. and Gonçalves R. F. (2016). 'Selecting a Notation to Modeling Business Process: A Systematic Literature Review of Technics and Tools'. In: *Advances in Production Management Systems. Initiatives for a Sustainable World. APMS 2016.* IFIP Advances in Information and Communication Technology. Springer, Cham.

Aysolmaz, B., Schunselaar, D. M. M., Reijers, H. A. and Yaldiz, A. (2019). 'Selecting a process variant modeling approach: guidelines and application'. In: *Software & Systems Modeling* 18.2.

Aysolmaz, B., Yaldiz, A. and Reijers, H. (2016). 'A Process Variant Modeling Method Comparison: Experience Report'. In: *Enterprise, Business-Process and Information Systems Modeling.* Springer International Publishing.

Becker, J., Delfmann, P., Herwig, S., Lis, L. and Stein, A. (2009). 'Towards Increased Comparability of Conceptual Models – Enforcing Naming Conventions through Domain Thesauri and Linguistic Grammars'. In: *ECIS 2009 Proceedings.*

Ben Hassen, M., Turki, M. and Gargouri, F. (2019). 'A Multi-criteria Evaluation Approach for Selecting a Sensitive Business Process Modeling Language for Knowledge Management'. In: *Journal on Data Semantics* 8.3.

Bflow.org (2015). *bflow* Toolbox - Anpassen und Erweitern von bflow*.* URL: `http://bflow.org/media/files/Anpassen-und-Erweitern-von-bflow-1.2.7-01.zip`.

Bflow.org (2017). *bflow* Benutzerhandbuch.* URL: `http://bflow.org/media/files/bflow_benutzerhandbuch1_3_1-02.pdf`.

Bianchini, D., Cappiello, C., De Antonellis, V. and Pernici, B. (2009). 'Semantic Service Design for Collaborative Business Processes in Internetworked Enterprises'. In: *Advances in Conceptual Modeling - Challenging Perspectives.* Springer, Berlin, Heidelberg.

Bjeković, M., Proper, H. and Sottet, J.-S. (2014). 'Enterprise Modelling Languages – Just Enough Standardisation?' In: *Business Modeling and Software Design.* Springer, Cham.

BOC Group (2021). *The ADONIS NP 11.0 User Manual.* URL: `https://docs.boc-group.com/adonis/en/docs/11.0/user_manual/`.

Bock, A. and Frank, U. (2016). 'MEMO GoalML: A Context-Enriched Modeling Language to Support Reflective Organizational Goal Planning and Decision Processes'. In: *Conceptual Modeling.* Cham: Springer International Publishing.

Bock, A., Frank, U. and Kaczmarek-Heß, M. (2022). 'MEMO4ADO: A Comprehensive Environment for Multi-Perspective Enterprise Modeling'. In: *Modellierung 2022 Satellite Events.* Bonn: Gesellschaft für Informatik e.V.

Bock, A., Kattenstroth, H. and Overbeek, S. (2014). 'Towards a modeling method for supporting the management of organizational decision processes'. In: *Modellierung 2014.* Bonn: Gesellschaft für Informatik e.V.

Bonorden, L., Frerichs, M., Riebisch, M., Von Riegen, S., Hartke, F., Herzog, R., Hotz, L., Jürgensen, D., Kiele-Dunsche, M., Schottler, S. and Schroeder, R. (2022). 'Decision-Making About Federated Digital Twins-–How to Distribute Information Storage and Computing'. In: *Modellierung 2022.* Bonn: Gesellschaft für Informatik.

Bouchbout, K., Akoka, J. and Alimazighi, Z. (2010). 'Proposition of a Generic Metamodel for Interorganizational Business Processes'. In: *EOMAS '10: Proceedings of the 6th International Workshop on Enterprise & Organizational Modeling and Simulation.*

Brocke, J. v., Becker, J., Simons, A. and Fleischer, S. (2008). 'Towards the Specification of Digital Content - The Enterprise Content Modeling Language (ECML).' In: *14th Americas Conference on Information Systems, AMCIS 2008.* Vol. 2.

Cabral, L., Norton, B. and Domingue, J. (2009). ‘The business process modelling ontology’. In: *ACM International Conference Proceeding Series.*

Castellanos, A., Tremblay, M., Lukyanenko, R. and Samuel, B. (2020). ‘Basic Classes in Conceptual Modeling: Theory and Practical Guidelines’. In: *Journal of the Association for Information Systems* 21.4.

Chen, P. P.-S. (1975). ‘The entity-relationship model’. In: *ACM SIGIR Forum* 10.3.

Corradini, F., Ferrari, A., Fornari, F., Gnesi, S., Polini, A., Re, B. and Spagnolo, G. O. (2018). ‘A Guidelines framework for understandable BPMN models’. In: *Data & Knowledge Engineering* 113.

Craciunean, D.-C. (2019). ‘Categorical Grammars for Processes Modeling’. In: *International Journal of Advanced Computer Science and Applications* 10.1.

Davenport, T. (1993). *Process Innovation: Reengineering Work Through Information Technology.* Harvard Business Press.

Delcambre, L. M. L., Liddle, S. W., Pastor, O. and Storey, V. C. (2018). ‘A Reference Framework for Conceptual Modeling’. In: *Conceptual Modeling.*

Dirndorfer, M., Fischer, H. and Sneed, S. (2013). ‘Case Study on the Interoperability of Business Process Management Software’. In: *S-BPM ONE - Running Processes.*

Döller, V. and Karagiannis, D. (2021). ‘Formalizing Conceptual Modeling Methods with MetaMorph’. In: *Enterprise, Business-Process and Information Systems Modeling.* Cham: Springer International Publishing.

Döller, V., Karagiannis, D. and Utz, W. (2023). ‘MetaMorph: formalization of domain-specific conceptual modeling methods—an evaluative case study, juxtaposition and empirical assessment’. In: *Software and Systems Modeling* 22 (1).

Dumas, M., La Rosa, M., Mendling, J. and Reijers, H. A. (2018). *Fundamentals of Business Process Management.* 2nd. Springer-Verlag Berlin Heidelberg.

Efendioglu, N., Woitsch, R. and Utz, W. (2016). ‘A toolbox supporting agile modelling method engineering: ADOxx.org modelling method conceptualization environment’. In: *The Practice of Enterprise Modeling.*

Al-Fedaghi, S. (2018). ‘Flow-Based Process Modeling: Application in BPMN and Process-Oriented Software Systems’. In: *Proceedings of the Computational Methods in Systems and Software.*

Fellmann, M., Hogrebe, F., Thomas, O. and Nüttgens, M. (2010). ‘What’s inside the Box ? Prospects and Limitations of Semantic Verification in Process Modeling’. In: *EMISA 2010. Einflussfaktoren auf die Entwicklung flexibler, integrierter Informationssysteme. Beiträge des Workshops der GI-Fachgruppe EMISA (Entwicklungsmethoden für Informationssysteme und deren Anwendung).* Bonn.

Fellmann, M., Metzger, D., Jannaber, S., Zarvic, N. and Thomas, O. (2018). ‘Process Modeling Recommender Systems: A Generic Data Model and Its Application to a Smart Glasses-based Modeling Environment’. In: *Business and Information Systems Engineering* 60.1.

Fengel, J. and Rebstock, M. (2009). ‘Model-based domain ontology engineering’. In: *SBPM ’09: Proceedings of the 4th International Workshop on Semantic Business Process Management.*

Fengel, J. and Rebstock, M. (2010). ‘Linking Heterogeneous Conceptual Models through a Unifying Modeling Concepts Ontology’. In: *Proceedings of the 5th International Workshop on Semantic Business Process Management SBPM 2010.* Heraklion, Greece.

Fettke, P. and Reisig, W. (2022). ‘Breathing Life into Models: The Next Generation of Enterprise Modeling’. In: *CoRR* abs/2205.09591.

Fill, H.-G. (2017). 'SeMFIS: A flexible engineering platform for semantic annotations of conceptual models'. In: *Semantic Web* 8.5.

Fill, H.-G. and Karagiannis, D. (2013). 'On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform'. In: *Enterprise Modelling and Information Systems Architectures - An International Journal* 8.1.

Fonseca, C. M., Almeida, J. P. A., Guizzardi, G. and Carvalho, V. A. (2021). 'Multi-level conceptual modeling: Theory, language and application'. In: *Data & Knowledge Engineering* 134.

Frank, U. (2011a). 'MEMO Organisation Modelling Language (1) – Focus on Organisational Structure'. In: *ICB Research Report 48*. Essen: University of Duisburg-Essen.

Frank, U. (2011b). 'MEMO Organisation Modelling Language (2) – Focus on Business Processes'. In: *ICB Research Report 49*. Essen: University of Duisburg-Essen.

Frank, U. (2014). 'Multi-perspective enterprise modeling: Foundational concepts, prospects and future research challenges'. In: *Software and Systems Modeling* 13.3.

Frank, U., Kaczmarek-Heß, M. and De Kinderen, S. (2021). 'IT Infrastructure Modeling Language (ITML): A DSML for Supporting IT Management'. In: *ICB Research Report 72*. Essen: University of Duisburg-Essen.

Frerichs, M., Leible, S. and Nüttgens, M. (2021). 'Towards method- and tool-independent business process modeling'. In: *INFORMATIK 2021*. Bonn: Gesellschaft für Informatik.

Frerichs, M. and Nüttgens, M. (2022). 'Modeling the Enterprise Digital Twin: Towards an Open Platform for Analytics & Compliance Operations'. In: *Modellierung 2022*. Bonn: Gesellschaft für Informatik.

Gadatsch, A. (2020). 'IT-Unterstützung für das Prozessmanagement'. In: *Grundkurs Geschäftsprozess-Management: Analyse, Modellierung, Optimierung und Controlling von Prozessen*. Springer Vieweg, Wiesbaden.

Gassen, J. B., Mendling, J., Thom, L. H. and Oliveira, J. P. M. de (2015). 'Towards Guiding Process Modelers Depending upon Their Expertise Levels'. In: *Advanced Information Systems Engineering Workshops. CAiSE 2015*. Springer, Cham.

Grangel, R., Chalmeta, R., Schuster, S. and Peña, I. (2005). 'Exchange of Business Process Models Using the POP* Meta-model'. In: *Business Process Management Workshops. BPM 2005*. Springer, Berlin, Heidelberg.

Gray, T., Bork, D. and De Vries, M. (2020). 'A New DEMO Modelling Tool that Facilitates Model Transformations'. In: *Enterprise, Business-Process and Information Systems Modeling*. Cham: Springer International Publishing.

Gregor, S. and Hevner, A. R. (2013). 'Positioning and Presenting Design Science Research for Maximum Impact'. In: *MIS Quarterly* 37.2.

Guarino, N. (1998). 'Formal Ontologies and Information Systems'. In: *Formal Ontology in Information Systems: Proceedings of the first international conference (FOIS'98)*. Trento, Italy.

Guizzardi, G., Fonseca, C. M., Benevides, A. B., Almeida, J. P. A., Porello, D. and Sales, T. P. (2018). 'Endurant Types in Ontology-Driven Conceptual Modeling: Towards OntoUML 2.0'. In: *Conceptual Modeling. ER 2018*. Springer, Cham.

Guizzardi, G., Wagner, G., Almeida, J. and Guizzardi, R. (Feb. 2015). 'Towards Ontological Foundations for Conceptual Modeling: The Unified Foundational Ontology (UFO) Story'. In: *Applied Ontology* 10.3-4.

Hammer, M. (1996). *Beyond Reengineering: How the Process-Centered Organization is Changing Our Lives*. New York, USA: Harper Business.

Hammer, M. and Champy, J. (1994). *Business Reengineering: Die Radikalkur für das Unternehmen.* Campus-Verl.

Havel, J.-M., Steinhorst, M., Dietrich, H.-A. and Delfmann, P. (2014). 'Supporting Terminological Standardization in Conceptual Models - a Plugin for a Meta-Modelling Tool'. In: *Proceedings of the European Conference on Information Systems (ECIS) 2014.* Tel Aviv, Israel.

Heidari, F., Loucopoulos, P., Brazier, F. and Barjis, J. (2013). *A Unified View of Business Process Modelling Languages.* Tech. rep.

Hesse, W. and Mayr, H. C. (2008). 'Modellierung in der Softwaretechnik: eine Bestandsaufnahme'. In: *Informatik-Spektrum* 31.5, pp. 377–393.

Hofferer, P. (2007). 'Achieving Business Process Model Interoperability Using Metamodels and Ontologies'. In: *ECIS 2007 Proceedings.* Vol. ECIS 2007.

Jarke, M., Jeusfeld, M. A., Nissen, H. W. and Quix, C. (2009). 'Heterogeneity in Model Management: A Meta Modeling Approach'. In: *Conceptual Modeling: Foundations and Applications.* Springer, Berlin, Heidelberg.

Johannsen, F. (2007). *Transformation von Prozessmodellen: Bewertung XML-basierter Ansätze.* Vol. 4. BDVB-Award Geschäftsprozess- und Projektmanagement / Bundesverband Deutscher Volks- und Betriebswirte e.V. (bdvb), Fachgruppe für Geschäftsprozess- und Projektmanagement. Bremen: Salzwasser-Verlag.

Junginger, S., Kühn, H., Strobl, R. and Karagiannis, D. (Oct. 2000). 'Ein Geschäftsprozessmanagement-Werkzeug der nächsten Generation: ADONIS: Konzeption und Anwendungen'. In: *Wirtschaftsinformatik* 42.5.

Kappel, G., Wimmer, M., Retschitzegger, W. and Schwinger, W. (2011). 'Leveraging Model-Based Tool Integration by Conceptual Modeling Techniques'. In: *The Evolution of Conceptual Modeling.* Springer, Berlin, Heidelberg.

Karagiannis, D. (2018). 'Conceptual Modelling Methods: The AMME Agile Engineering Approach'. In: *Informatics in Economy.* Springer, Cham.

Karagiannis, D. (2022). 'Conceptual Modelling Methods: The AMME Agile Engineering Approach'. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools.* Cham: Springer International Publishing.

Karagiannis, D., Buchmann, R. A., Burzynski, P., Reimer, U. and Walch, M. (2016). 'Fundamental Conceptual Modeling Languages in OMiLAB'. In: *Domain-Specific Conceptual Modeling.* Springer, Cham.

Karagiannis, D., Burzynski, P., Utz, W. and Buchmann, R. A. (2019). 'A Metamodeling Approach to Support the Engineering of Modeling Method Requirements'. In: *2019 IEEE 27th International Requirements Engineering Conference (RE).* IEEE.

Karagiannis, D. and Höfferer, P. (2006). 'Metamodels in action: An overview'. In: *ICSOFT 2006 - 1st International Conference on Software and Data Technologies, Proceedings.* Vol. 1. Setúbal, Portugal.

Karagiannis, D. and Kühn, H. (2002). 'Metamodelling Platforms'. In: *E-Commerce and Web Technologies.* Springer, Berlin, Heidelberg.

Karagiannis, D. and Visic, N. (2011). 'Next Generation of Modelling Platforms'. In: *Perspectives in Business Informatics Research.* Springer, Berlin, Heidelberg.

Karhof, A., Jannaber, S., Riehle, D. M., Thomas, O., Delfmann, P. and Becker, J. (2016). 'On the de-facto Standard of Event-driven Process Chains: Reviewing EPC Implementations in Process Modelling Tools'. In: *Modellierung.*

Keller, G., Nüttgens, M. and Scheer, A.-W. (1992). 'Semantische Prozeßmodellierung auf der Grundlage 'Ereignisgesteuerter Prozeßketten (EPK)''. In: *Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes* Heft 89.

Keller, G. (1999). *SAP R/3 prozeßorientiert anwenden. Iteratives Prozeß-Prototyping mit Prozeßketten.* 3rd ed. Addison-Wesley.

Kern, H. (2014). 'Study of Interoperability between meta-modeling tools'. In: *2014 Federated Conference on Computer Science and Information Systems.* Warsaw, Poland: IEEE.

Klein, M. (2001). 'Combining and Relating Ontologies: An Analysis of Problems and Solutions'. In: *Workshop on Ontologies and Information Sharing (IJCAI'01).* Vol. 47. Seattle, USA.

Koschmider, A., Caporale, T., Fellmann, M., Lehner, J. and Oberweis, A. (2015). 'Business Process Modeling Support by Depictive and Descriptive Diagrams'. In: *Enterprise modelling and information systems architectures.* Bonn: Gesellschaft für Informatik e.V.

Kožíšek, F. and Vrana, I. (2017). 'Business Process Modelling Languages'. In: *Agris on-line Papers in Economics and Informatics* 9.3.

Kühn, H., Bayer, F., Junginger, S. and Karagiannis, D. (2003). 'Enterprise Model Integration'. In: *E-Commerce and Web Technologies.* Springer, Berlin, Heidelberg.

Kühn, H., Junginger, S., Karagiannis, D. and Petersen, C. (1999). 'Metamodellierung im Geschäftsprozeßmanagement: Konzepte, Erfahrungen und Potentiale'. In: *Modellierung '99.* Vieweg+Teubner Verlag, Wiesbaden.

Kühn, H. and Murzek, M. (2006). 'Interoperability Issues in Metamodelling Platforms'. In: *Interoperability of Enterprise Software and Applications.* Springer London.

Kunze, M., Luebbe, A., Weidlich, M. and Weske, M. (2011). 'Towards Understanding Process Modeling – The Case of the BPM Academic Initiative'. In: *Business Process Model and Notation. BPMN 2011.* Springer, Berlin, Heidelberg.

Lantow, B., Sandkuhl, K. and Stirna, J. (2022). 'Enterprise Modeling with 4EM: Perspectives and Method'. In: *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools.* Cham: Springer International Publishing.

Laue, R. (2010). 'Musterbasierte Überprüfung der Qualitätseigenschaften von Geschäftsprozessmodellen'. PhD thesis. Universität Leipzig.

Ledeczi, A., Maroti, M., Bakay, A., Karsai, G., Garrett, J., Thomason, C., Nordstrom, G., Sprinkle, J. and Volgyesi, P. (2001). 'The Generic Modeling Environment'. In: *International Workshop on Intelligent Signal Processing (WISP'2001).* Budapest, Hungary.

Lee, J., Gruninger, M., Jin, Y., Malone, T., Tate, A., Gregg, Y. and Other Members of the PIF Working Group (1998). 'The Process Interchange Format and Framework'. In: *The Knowledge Engineering Review* 13.1.

Levina, O. (2012). 'Measuring information content change in EPC to BPMN business process model transformation'. In: *AMCIS 2012 Proceedings.* Vol. 4.

Lin, Y. and Strasunskas, D. (2005). *Ontology-based Semantic Annotation of Process Templates for Reuse.* Tech. rep.

Lin, Y., Strasunskas, D., Hakkarainen, S., Krogstie, J. and Sølvberg, A. (2006). 'Semantic annotation framework to manage semantic heterogeneity of process models'. In: *Advanced Information Systems Engineering. CAiSE 2006.* Vol. 4001 LNCS. Springer, Berlin, Heidelberg.

List, B. and Korherr, B. (2006). 'An evaluation of conceptual business process modelling languages'. In: *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing.* New York, USA: Association for Computing Machinery.

Lukyanenko, R., Samuel, B. M., Storey, V. C. and Sturm, A. (2022). 'Conceptual Modeling Systems: A Vision for the Future of Conceptual Modeling'. In: *ER'2022 Forum and Symposium, CEUR Workshop Proceedings.*

Malekan, H. S. and Afsarmanesh, H. (2013). 'Analysis and evaluation of Business Process modeling adoption in collaborative networks'. In: *Proceedings of the Third International Symposium on Business Modeling and Software Design - Volume 1: BMSD.* Noordwijkerhout, Netherlands: SciTePress.

Martino, B. D., Esposito, A. and Cretella, G. (2018). 'From business process models to the cloud: a semantic approach'. In: *International Journal of High Performance Computing and Networking* 12.4.

Mayr, H. C. and Thalheim, B. (2021). 'The triptych of conceptual modeling'. In: *Software and Systems Modeling* 20 (1).

Mendling, J. and Müller, M. (2003). 'A Comparison of BPML and BPEL4WS'. In: *Berliner XML Tage 2003.*

Mendling, J., Neumann, G. and Nüttgens, M. (2004). 'A Comparison of XML Interchange Formats for Business Process Modelling'. In: *EMISA 2004 – Informationssysteme im E-Business und E-Government.*

Mendling, J. and Nüttgens, M. (2002). 'Event-Driven-Process-Chain-Markup-Language (EPML): Anforderungen zur Definition eines XML-Schemas für Ereignisgesteuerte Prozessketten (EPK)'. In: *EPK 2002 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten.* Trier, Germany.

Mendling, J. and Nüttgens, M. (2003). 'XML-basierte Geschäftsprozessmodellierung'. In: *Wirtschaftsinformatik 2003/Band II.* Physica, Heidelberg.

Mendling, J. and Nüttgens, M. (2004a). 'Exchanging EPC Business Process Models with EPML'. In: *Proc. of the 1st GI Workshop "XML4BPM - XML Interchange Formats for Business Process Management.*

Mendling, J. and Nüttgens, M. (2004b). 'Transformation of ARIS Markup Language to EPML'. In: *Proceedings of the 3rd GI Workshop on Event-Driven Process Chain.*

Mendling, J. and Nüttgens, M. (2004c). 'XML-based Reference Modelling: Foundations of an EPC Markup Language'. In: *Referenzmodellierung.* Physica, Heidelberg.

Mendling, J. and Nüttgens, M. (July 2006). 'EPC markup language (EPML): an XML-based interchange format for event-driven process chains (EPC)'. In: *Information Systems and e-Business Management* 4.

Mendling, J., Pérez De Laborda, C. and Zdun, U. (2005). 'Towards an Integrated BPM Schema: Control Flow Heterogeneity of PNML and BPEL4WS'. In: *Professional Knowledge Management. WM 2005.* Springer, Berlin, Heidelberg.

Murzek, M. and Kramler, G. (2007). 'The Model Morphing Approach-Horizontal Transformations between Business Process Models'. In: *Proceedings of the 6th International Conference on Perspectives in Business Information Research – BIR 2007.* Tampere, Finnland: University of Tampere.

Nägele, R. and Schreiner, P. (2002). 'Bewertung von Werkzeugen für das Management von Geschäftsprozessen'. In: *Führung und Organisation (ZfO)* 4.71.

Norton, B., Cabral, L. and Nitzsche, J. (2009). 'Ontology-based translation of business process models'. In: *ICIW '09: Proceedings of the 2009 Fourth International Conference on Internet and Web Applications and Services.* Venice/Mestre, Italy: IEEE.

Nüttgens, M. and Zimmermann, V. (1998). 'Geschäftsprozeßmodellierung mit der objektorientierten Ereignisgesteuerten Prozeßkette (oEPK)'. In: *Informationsmodellierung*. Wiesbaden: Deutscher Universitätsverlag.

Object Management Group (OMG) (2008). *Business Process Definition Metamodel Specification*. URL: `https://www.omg.org/spec/BPDM/`.

Object Management Group (OMG) (2015). *XML Metadata Interchange (XMI) Specification*. URL: `https://www.omg.org/spec/XMI/`.

Object Management Group (OMG) (2016). *Meta Object Facility (MOF) Specification*. URL: `https://www.omg.org/spec/MOF/`.

Object Management Group (OMG) (2017). *Unified Modeling Language (UML) Specification*. URL: `https://www.omg.org/spec/UML/`.

Object Management Group (OMG) (2021). *BPMN Model Interchange Working Group*. URL: `https://www.omgwiki.org/bpmn-miwg/`.

Olivé, A. (2017). 'The Universal Ontology: A Vision for Conceptual Modeling and the Semantic Web (Invited Paper)'. In: *Conceptual Modeling*. Springer, Cham.

Ouksel, A. M. and Sheth, A. (1999). 'Semantic Interoperability in Global Information Systems: A Brief Introduction to the Research Area and the Special Section'. In: *ACM SIGMOD Record* 28.1.

Ouyang, C., Verbeek, E., Van der Aalst, W. M. P., Breutel, S., Dumas, M. and ter Hofstede, A. H. M. (July 2007). 'Formal semantics and analysis of control flow in WS-BPEL'. In: *Science of Computer Programming* 67.2-3.

Pastor, O., Giachetti, G., Marín, B. and Valverde, F. (2013). 'Automating the Interoperability of Conceptual Models in Specific Development Domains'. In: *Domain Engineering*. Springer, Berlin, Heidelberg.

Petri, C. (1962). 'Kommunikation mit Automaten'. PhD thesis. Bonn: Institut für Instrumentelle Mathematik.

Recker, J., Lukyanenko, R., Jabbari, M., Samuel, B. and Castellanos, A. (Mar. 2021). 'From Representation to Mediation: A New Agenda for Conceptual Modeling Research in A Digital World'. In: *MIS Quarterly* 45.

Reijers, H. A. (Jan. 2021). 'Business Process Management: The evolution of a discipline'. In: *Computers in Industry*.

Reinheimer, S. (2011). 'Modeling Needs in the BPM Consulting Process'. In: *S-BPM ONE - Learning by Doing - Doing by Learning. S-BPM ONE 2011.* Springer, Berlin, Heidelberg.

Riggio, R., Ursino, D., Kühn, H. and Karagiannis, D. (2005). 'Interoperability in meta-environments: An XMI-based approach'. In: *Advanced Information Systems Engineering. CAiSE 2005.* Springer, Berlin, Heidelberg.

Rinderle-Ma, S., Reichert, M. and Weber, B. (2008). 'On the Formal Semantics of Change Patterns in Process-Aware Information Systems'. In: *Conceptual Modeling - ER 2008*. Springer, Berlin, Heidelberg.

Rosenkranz, F. (2006). *Geschäftsprozesse - Modell- und computergestützte Planung*. 2nd. Berlin/Heidelberg: Springer-Verlag.

Rudnitckaia, J., Intayoad, W., Becker, T. and Hruska, T. (2019). 'Applying Process Mining to the Ship Handling Process at Oil Terminal'. In: *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*. Taipei, Taiwan: IEEE.

Scheer, A.-W. (1998). *ARIS – Modellierungsmethoden, Metamodelle, Anwendungen*. 3rd ed. Berlin, Heidelberg: Springer.

Scheer, A.-W. (2002). *ARIS — Vom Geschäftsprozess zum Anwendungssystem.* 4th. Springer-Verlag Berlin Heidelberg.

Scheer, A.-W. and Nüttgens, M. (2000). 'ARIS Architecture and Reference Models for Business Process Management'. In: *Business Process Management: Models, Techniques, and Empirical Studies.* Berlin, Heidelberg: Springer Berlin Heidelberg.

Schichl, H. (2004). 'Models and the History of Modeling'. In: *Modeling Languages in Mathematical Optimization.* Boston, MA: Springer.

Schützenmeier, N., Jablonski, S. and Schönig, S. (2021). 'Towards a Hybrid Process Modeling Language'. In: *Research Challenges in Information Science.* Cham: Springer International Publishing.

Signavio (2021a). *Product Information Signavio Process Manager.* URL: `https://www.signavio.com/downloads/product-information/signavio-product-brochure/`.

Signavio (2021b). *Signavio Process Manager User Guide.* URL: `https://documentation.signavio.com/pdfs/en/Classic-Process-Manager-User-Guide-en.pdf`.

Software AG (2016). *BPM with ARIS - The ARIS method, Lecture Slides - Part II.* URL: `https://s3-eu-west-1.amazonaws.com/arisexpress/community2/documents/urelation/BPM-ARIS_Part2.pdf`.

Software AG (2018a). *ARIS – Technical Product Matrix, ARIS 10.0 SR6, October 2018.* URL: `https://documentation.softwareag.com/aris/Designer/10-0sr6/yad10-0sr6e/10-0sr6_Technical_Product_Matrix.pdf`.

Software AG (2018b). *ARIS Function Product Matrix.* URL: `https://www.ariscommunity.com/system/files/10.0-SR7-Functional-Product-Matrix.pdf`.

Stachowiak, H. (1992). 'Modell'. In: *Handlexikon zur Wissenschaftstheorie.* München: DTV.

Strecker, S., Frank, U., Heise, D. and Kattenstroth, H. (2012). 'MetricM: a modeling method in support of the reflective design and use of performance measurement systems'. In: *Information Systems and e-Business Management* 10 (2).

Tarkkanen, K. (2009). 'Business Process Modeling for Non-uniform Work'. In: *Enterprise Information Systems. ICEIS 2008.* Springer, Berlin, Heidelberg.

Thalheim, B. (2010). 'Towards a Theory of Conceptual Modelling'. In: *Advances in Conceptual Modeling - Challenging Perspectives.* Springer, Berlin, Heidelberg.

Thalheim, B. (2012). 'Syntax, Semantics and Pragmatics of Conceptual Modelling'. In: *Natural Language Processing and Information Systems. NLDB 2012.* Springer, Berlin, Heidelberg.

Thalheim, B. (2013). 'The Conception of the Model'. In: *Business Information Systems.* Springer, Berlin, Heidelberg.

Thomas, M. (2002). 'Modelle in der Fachsprache der Informatik'. In: *Forschungsbeiträge zur „Didaktik der Informatik" - Theorie, Praxis, Evaluation.* Bonn: Gesellschaft für Informatik e.V.

Thomas, O. (2005). 'Das Modellverständnis in der Wirtschaftsinformatik: Historie, Literaturanalyse und Begriffsexplikation'. In: *Veröffentlichungen des Instituts für Wirtschaftsinformatik* Heft 184.

Thomas, O. and Fellmann, M. (2007). 'Semantic EPC: Enhancing Process Modeling Using Ontology Languages'. In: *Semantic Business Process and Product Lifecycle Management. Proceedings of the Workshop SBPM 2007, CEUR Workshop Proceedings.* Vol. 251. Innsbruck, Austria.

Van der Aalst, W. M. P. (2015). 'Business process management as the "Killer App" for Petri nets'. In: *Software & Systems Modeling* 14.2.

Van der Aalst, W. M. P. and ter Hofstede, A. H. M. (2005). ‘YAWL: yet another workflow language’. In: *Information Systems* 30.4.

Van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, A. and Barros, A. P. (2003). ‘Workflow Patterns’. In: *Distributed and Parallel Databases* 14.

Van der Aalst, W. M. P., ter Hofstede, A. H. M. and Dumas, M. (2005). ‘Patterns of Process Modeling’. In: *Process-Aware Information Systems*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Van Gils, B., Hoppenbrouwers, S. and Proper, H. A. (2022). ‘Conceptual Modeling in Digital Transformations – Enabling enterprise design dialogues’. In: *PoEM 2022 Forum, 15th IFIP Working Conference on the Practice of Enterprise Modeling 2022 (PoEM-Forum 2022), CEUR Workshop Proceedings*. London.

Vanderhaeghen, D., Zang, S., Hofer, A. and Adam, O. (2005a). ‘XML-based Transformation of Business Process Models - Enabler for Collaborative Business Process Management’. In: *Proceedings of the Second GI-Workshop XML4BPM - XML for Business Process Management held at the 11th Conference Business*. CEUR Workshop Proceedings Vol. 145.

Vanderhaeghen, D., Zang, S. and Scheer, A.-W. (2005b). *Interorganisationales Geschäftsprozessmanagement durch Modelltransformation*. Tech. rep. Veröffentlichungen des Instituts für Wirtschaftsinformatik - Saarbrücken.

Weidlich, M., Barros, A., Mendling, J. and Weske, M. (2009a). ‘Vertical Alignment of Process Models – How Can We Get There?’ In: *Enterprise, Business-Process and Information Systems Modeling. BPMDS 2009, EMMSAD 2009*. Springer, Berlin, Heidelberg.

Weidlich, M., Weske, M. and Mendling, J. (2009b). ‘Change propagation in process models using behavioural profiles’. In: *2009 IEEE International Conference on Services Computing*. Bangalore, India: IEEE.

Wieringa, R. (2011). ‘Real-World Semantics of Conceptual Models’. In: *The Evolution of Conceptual Modeling*. Springer, Berlin, Heidelberg.

Winter, R. and Blaschke, M. (2018). ‘Same Same But Different – Federating Enterprise Model-ling for the Digitalized and Data-driven Enterprise’. In: *EMISA Forum* 38.1.

Yousfi, A., Bauer, C., Saidi, R. and Dey, A. K. (2016). ‘UBPMN: A BPMN extension for modeling ubiquitous business processes’. In: *Information and Software Technology* 74.

Zaniolo, C. and Melkanoff, M. A. (1982). ‘A Formal Approach to the Definition and the Design of Conceptual Schemata for Database Systems’. In: *ACM Transactions on Database Systems* 7.1.

Zivkovic, S., Kühn, H. and Karagiannis, D. (2007). ‘Facilitate Modelling Using Method Integration: An Approach Using Mappings and Integration Rules’. In: *ECIS 2007 Proceedings*.

Chapter 21

The Process of Detecting Suspicious Communication with Psycholinguistics and Text Mining Techniques

Ewelina Księżniak and Witold Abramowicz

This chapter presents the process of detecting suspicious communication with text mining and psycholinguistics techniques. The chapter describes the key finding of research in the field of detecting deception in text with psycholinguistics and presents the algorithm designed to detect communication aims to deceive or harm the receivers of the messages.

21.1 Introduction

We discuss the process of detecting suspicious communication with text mining and psycholinguistics analysis techniques. According to many researchers lying is a cognitively complex task that influences the language used by liars. Therefore, it is possible to detect deception in a text by linguistics analysis. The proposed method combines the discoveries from the psycholinguistics and computer science fields and is applicable in many areas. The term 'suspicious communication' may be understood as any communication between individuals or business parties that aims to deceive or harm the receiver of the message.

In the chapter, we focus on presenting the application of the proposed algorithm to detect financial scams with an example of the Enron case study. However, the discussed method may also be applied to detect fake news or phishing. The chapter is divided into three main sections:

- Psycholinguistics in deception detection – this section discusses key concepts related to deception detection with psycholinguistics (Section 21.2).
- The process of detecting suspicious communication with Enron emails case study – in this section, we describe the proposed method of detecting suspicious communication with psycholinguistics with the Enron emails case study. We also briefly discuss Enron's history and present the data cleaning process (Section 21.3).
- Other applications and future work – this section examines future prospects and summarizes the potential use of the algorithm in various fields, including fake news detection (Section 21.4).

21.2 Psycholinguistics in Deception Detection

Psycholinguistics is a field of cognitive science combining the sciences of biology, psychology, linguistics, and computer science which studies the impact of psychological and neurobiological factors on how people acquire and use language. Numerous studies in the field of psycholinguistics prove that the way we communicate using language (whether in writing or speaking) results from psychological and neurobiological conditions. One of the areas of psycholinguistics is its use in detecting deception. Since lying is a task of high cognitive complexity, and lying is accompanied by some emotions (stress or excitement), we can assume that deceptive and honest communication will differ in style. The section explains how lying vary – from a cognitive perspective – from exchanging open communication and presents the essential characteristics related to the stylometry of deceptive messages.

21.2.1 Lying as a Cognitively Complex Process

According to the literature on cognitive science, people are more likely to tell the truth. Lying is more difficult from a cognitive science point of view and therefore affects the emotional and psychosomatic state of the lying person. This chapter describes the process of creating lies and its effects on the functioning of a healthy person. According to Barbara (2014), long-term memory stores only information relevant to the person. Since actual events evoke stronger emotions than made-up stories, it can be assumed that memories corresponding to the actual events are stored in long-term memory. In a situation of sincere communication about specific circumstances, the information about them is automatically transferred to the working memory and is thus available in the consciousness of the interlocutor. By sending a truthful message, this kind of process takes place in the brain. The process of lying is more complex – after the activation of the true information (transfer from long-term memory to working memory), the interlocutor must (Ganis and Keenan 2009):

- decide to lie to the other person (if he has not already done so), taking into account and evaluating the situational context in which he finds himself,
- extracts information from memory that might be useful for constructing false responses,
- retain and manipulate the information extracted from the long-term memory at the working memory level until the interlocutor responds,
- remembers the false statement so that when the interlocutor refers to it, the liar can maintain consistency.

Thus, the process triggered by lying is much longer than the process associated with telling the truth, and it reflects the higher cognitive complexity. According to Clark et al. (2008), cognitively complex tasks are defined as activities that require conscious, conceptual knowledge of the individual – and unconscious, procedurally acquired information (automatic memory).

Moreover, many researchers confirm that lying is associated with control processes. The liar must not only watch over the coherence and credibility of the story told but also – in the situation of face-to-face communication – control the behavior of the receiver of the false message (check whether the interlocutor believes him and, in case of doubt, change the lie or his posture – to hide symptoms of stress) (Van Bockstaele et al. 2012). The studies of brain activity using EEG or fMRI prove a connection between the intensity of the control processes and lying. Results of those studies show that during lying, neurons in the region of the prefrontal cortex – the brain area responsible for the control process – are activated. The following brain areas are also activated during lying: anterior insula, middle prefrontal, posterior parietal cortex, bilateral inferior parietal lobe (Adams-Quackenbush 2015). The activity of the different brain areas indicates that lying is a cognitively complex task. In the deception detection area, cognitive complexity is an essential factor that influences the way the lier formulates their speech. Because lying requires more effort, deceptive speech is often less complex in a stylometric way.

21.2.2 Concreteness of Text

One of the features that distinguish honest and deceptive communication is the degree of specificity. In psycholinguistics, it is assumed that words can be given a score that reflects their degree of concreteness. According to the generally accepted definition, the relationship between concreteness and abstractness (the opposite of concreteness) refers to human sensory experiences related to a particular concept. For example, the noun ‘chair’ describes an object that can be physically felt by touch – so it has a high degree of concreteness. The term ‘independence’, on the other hand, which is classified as an ‘abstract’ concept, can be understood in many ways, and most importantly, ‘sense of independence’ is not related to any sensory experience (Pollock 2018). The sciences use the concrete and abstract relationship to detect lies. According to the studies analyzed, deceptive messages are characterized by a higher degree of abstractness. The higher degree of abstractness of texts containing lies could be explained by two theories – Reality Monitoring and Verifiability Approach. According to the Reality Monitoring (EM) approach, all differences between honest and deceptive communication result from different ways of creating two types of statements. True statements are based on the individual’s experience, while lying is a ‘cognitive operation’ that results in the presence of more details, e. g., about the place and time of the event, in sincere communication (Kleinberg et al. 2019). In the Verifiability Approach (VA), the primary factor distinguishing sincere and false messages is the verifiability of the details provided in the statements. Different from RM, this theory assumes that a liar (just like a truthful person) can also give much detailed information in the message to authenticate it, however, these details will be difficult to verify (Kleinberg et al. 2019).

Both the Reality Monitoring and Verifiability Approaches have an impact on text abstractness. The phenomenon can be explained by analyzing a simple example. Two statements are presented below; the first contains specific details, and the second describes the situation with more general characteristics. The possible answer of the witness during police interrogation to the question: ‘*What were you doing yesterday at 3.00 p.m.?*’ is: ‘*At 3:00 p.m. I was at Starbucks on Polna Street with Anna.*’ (Theorem a). The answer contains detailed information; the statement indicated that:

- the meeting took place at 3:00 p.m.,
- the meeting took place in Starbucks on Polna Street,
- subject met Anna.

A more general answer might be: ‘*I was having coffee with a friend this afternoon.*’ (Theorem b). Even though both answers coincide and may describe the same event:

- ‘afternoon’ is less specific than 3 p.m.

- ‘coffee’ is less specific than ‘Starbucks on Polna’,
- ‘Anna’ is less abstract than ‘friend’.

According to the Reality Monitoring theory, it can be hypothesized that when asked about their activities at a specific time, a liar would construct their statement similarly to Theorem b. From the Verifiability Approach theory perspective, it is easier to discredit a specific claim such as the assertion that the subject met a person named Anna at 3:00 p.m. at a Starbucks on Polna Street, compared to more general information, such as meeting a friend for coffee (Kleinberg et al. 2019). Adopting the assumptions of the RM and VA theories affects the lexical vocabulary used by the liar, which reflects the level of abstraction of his utterance.

21.2.3 Psychological Distance

From a psychological point of view, liars try to avoid responsibility for telling untruths and somehow distance themselves from a false statement (Tovmasyan 2020). Psychological distance affects the linguistic structure of deceptive messages. One of the essential characteristics of misleading communication is the use of fewer first-person personal pronouns (I, we). By using the personal pronoun ‘I’, people unconsciously identify with the meaning of the statement – since, in lying, this relationship is reversed, it can be observed that liars avoid using this form (Keila and Skillicorn 2005).

Moreover, psycholinguistic research shows that based on the analysis of syntactical structure and the choice of word order, it is possible to determine the emotional attachment of a person to the discussed object. For example, in the sentence ‘Anna and I went to the movies’, the words, ‘Anna’ and ‘I’ are close to each other. In the statement ‘I went to the movies with Anna’, the word ‘Anna’ comes last. From a psycholinguistic point of view, the placement of the noun ‘Anna’ at the end of the second sentence may indicate a lower emotional attachment of the interlocutor to the person named Anna than in the first case (Mhundwa 2005).

Another consequence of the phenomenon of psychological distance is that liars tend to use fewer words associated with certainty. Liars may also try to avoid responsibility by weaving in words like ‘plan’, ‘try’, etc., in the context of plans for the future (Hauch et al. 2015).

21.2.4 Descriptive Features

As discussed in Section 21.2.1, most of the linguistic features of deceptive communication are based on the assumption that deception is cognitively more complex than sending genuine messages, resulting in a lower complexity of text that contains a lie. In general, the complexity of a text may be described by a lexical variety (the number of unique words in the text) or by ‘descriptive’ measures such as the length of sentences (Fuller et al. 2011). The list presented below contains examples of features related to the text complexity:

- total text complexity, measured as a:
  - total number of words used in the text,
  - total number of characters used in the text,
  - total number of sentences used in the text,
- sentence complexity, measured as a

  - average number of words per sentence in the text,
  - average number of conjunctions per sentence in the text,
  - average number of commas per sentence in the text,
- words complexity, measured as a;
  - average number of letters in words used,
  - average number of syllables in the word used,
- lexical diversity, measured as a:
  - number of unique words in the text (with normalized text length) measured with type-token ratio,
  - number of unique words in the text (with normalized text length) measured with hapax legomenon ratio.

Most researchers agree that lying is a cognitively complex task. Therefore, it should be assumed that messages containing lies are less complex:

- will consist of a smaller number of sentences/characters/words (colloquially, 'they will be shorter'),
- sentences used would be less context – would contain fewer conjunctions, commas, words,
- will use simpler words – deceptive text would contain words consists less number of letters/syllables,
- their vocabulary will also be more homogeneous (Fuller et al. 2011).

21.2.5 Sentiment Analysis

According to the literature review, one of the characteristics that distinguish fake news from genuine messages is sentiment intensity. According to Newman et al. (2003), deceptive messages usually contain more words referring to negative emotions. This hypothesis can be explained by the theory, according to which lying is often associated with an immoral act and, therefore, may cause feelings of guilt or fear of being 'caught'. Because of that, some deception detection techniques are based on counting words associated with anger, sadness, or fear. On the other hand, according to Goel's research findings Gandía and Huguet (2021), the sentiment of financial reports containing falsified data is three times more positive or four times more negative than the sentiment of correct reports. Wang et al. (2010) indicate a correlation between a highly positive sentiment of text and the occurrence of embezzlement (Dyck et al. 2013). A similar correlation was found by Larcker and Zakolyukina (2012), who observed that CEOs of companies that committed fraud used more extremely positive words, such as 'fantastic' or 'great' (Bel et al. 2021).

In psycholinguistics literature, there is no clear consensus on whether deceptive messages contain more positive or more negative vocabulary. However, many researchers indicate that sentiment level is essential in distinguishing between honest and misleading communication. In messages containing lies, sentiment polarity is often more potent. Two concepts could explain that phenomenon. On the one hand, lying is often perceived as an immoral act; deceiving can evoke emotional arousal and, depending on the nature of the interlocutor, can take the form of fear or, on the other hand, excitement (Zloteanu 2020). A person's emotional state affects the language used, which can be helpful from the perspective of developing deception detection methods. On the other hand, the purpose of the authors of false information is often to convince the recipient of the message to its truth, which results in using more persuasive and emotional language.

21.2.6 Pennebaker Model

Another psycholinguistics approach to detect deceptive messages is to use Pennebaker's model. Pennebaker's model focuses on analyzing four linguistic areas:

- number of first-person personal pronouns in the text – as described in Section 21.2.3, liers try not to identify themselves with false statements by avoiding to use of first-person personal pronouns.
- number of 'negative words' in the text – according to Pennbeker's theory, deceptive texts contain more words associated with negative feelings and emotions and words used in stressful situations. The approach could be explained by the fact that lying is often considered stressful.
- number of 'motion words' – theory assumes that deceptive messages contain more 'motion words' (words that describe action like walking) which could be explained by the fact that when lying, people focus on making up the flow of the story rather then describes details of the story (for example it is more accessible to made-up the false statement: 'on Saturday I walk through rail-way station', rather than, 'on Saturdays, I walk fastly through the railway station').
- number of exclusive words – it is assumed that deceptive texts contain fewer exclusive words (but, or), which is associated with the complexity of a text (Fissette and Vries 2017).

Pennebaker is an author of the Linguistic Inquiry and Word Count (LIWC) project. LIWC tool allows one to calculate the frequency of words in the text associated with a specific group of words used by people to express particular concerns or emotions (Pennebaker et al. 2015).

21.2.7 Deception Detection with Psycholinguistics – Summary

In this section, we have described the state-of-the-art approaches to detecting lies in the text. Table 21.1 contains a summary of described features.

Area	Examples of measures	Intensity of feature in deceptive language
Text concreteness	Reality Monitoring score calculated with LIWC software	Deceptive language tends to be more abstract
Psychological distance	Number of first-person personal pronouns	Fewer firs-person personal pronouns in deceptive text
Text complexity	Total number of words	Less complex
	Total number of characters	
	Total number of sentences	
Sentence complexity	Number of words per sentence in the text	
	Average number of conjunctions in sentences	
	Average number of commas in sentences	
Words complexity	Average number of letters in the word used	
	Average number of syllables in the word used	
Lexical diversity	Hapax legonemoin ratio	More homogenous lexicon in deceptive texts
	Type token ratio	
Sentiment	Synthetic sentiment score	Deceptive texts tend to use more emotional language
Pennbaker model	Number of first-person pronouns	Fewer firs-person personal pronouns in deceptive texts
	Number of motion words	More motion words in deceptive text
	Number of exclusive words	Fewer exclusive words in deceptive text
	Number of negative words	More negative words in deceptive text

Table 21.1: Deception detection: Linguistics features

21.3 Process of Detecting Suspicious Communication with Psycholinguistics: Enron Case Study

The study presents the process of detecting suspicious communication with psycholinguistics and text-mining techniques. The study has been conducted with the Enron emails dataset. The Enron scandal was one of the biggest frauds in history. Due to the scale of crimes related to falsifying Enron's financial results, it is assumed that there were emails exchanged between Enron's employees that contained a lie. The dataset version used in the study initially consisted of 517,407 messages. The proposed method of detecting suspicious communication in an organization with psycholinguistics and text-mining techniques is based on four steps:

1. Psycholinguistics features extraction – extraction of psycholinguistics features useful in deception detection area from text data in the analyzed corpus.
2. Identifying unusual text samples – in step 1, certain features were extracted from the text. Some samples of text may have extremely high or low levels of these specific features, which should be marked as anomalies. Depending on the specific feature, these anomalies may indicate a high or low probability that the analyzed text contains a lie.
3. Relevant topics extraction – extraction of topics from text marked as suspicious in the second step with text mining techniques. Determining the relevance of extracted topics.
4. Manual inspection – conducting a manual inspection of text marked as suspicious in step 2 and classified as relevant in step 3.

The present study did not consider the potential impact of dialect usage on the results. As the analyzed dataset pertains to business communication, it is reasonable to assume that the majority of the emails were written in a formal style. However, incorporating an analysis of dialect usage or language differences stemming from the personality, gender, or age of the message's author could potentially enhance the effectiveness of the deception detection algorithm. However, it is worth noting that incorporating such methods may involve additional costs. In the following sections of this chapter, we will provide an example of the abovementioned steps using the Enron email dataset. To provide context for the case study, we will also briefly discuss the history of the Enron corporation. Additionally, we will outline the data cleaning and experiment validation steps performed in this analysis.

21.3.1 The Brief History of Enron

Enron Corporation was founded in 1986 through the merger of Houston Gas Company and InterNorth Inc. Its CEO, Kenneth Lay, oversaw the company's operations in the gas industry, which initially positioned Enron as a 'typical' corporation. However, the hiring of Jeffrey Skilling, previously the head of the energy division at McKinsey Corporation, marked a turning point for Enron. Skilling saw the potential for growth in the newly deregulated energy market and proposed an innovative business model for energy sales (Cabello et al. 2009). This model was seen as revolutionary by management, who believed it would make Enron a leader in the global energy market. In 2000, Fortune Magazine named Enron 'The Most Innovative Corporation', with a stock value of 100 trillion USD, making it one of the largest companies in the United States (Pha 2010). However, not all of Enron's ventures were profitable, and the company faced financial struggles despite its impressive stock value. Enron employed two accounting techniques to mask these issues: market-to-market valuation and liability concealment. The former involved booking future revenues that had yet to be realized, while the latter involved transferring Enron's liabilities to subsidiaries to make the company's debts appear proportional to its capital (Pha 2010). In addition to these practices, Enron was also involved in causing artificial blackouts, which contributed to the energy crisis in California in 2000-2001. The deregulation of the energy market allowed for the sale of energy as derivatives, and favorable weather conditions in California at the time resulted in an imbalance of supply and demand (Weare 2003). Enron took advantage of this situation by creating artificial blackouts, which further increased demand and drove up electricity prices and profits (Wikipedia 2022). Enron's fraudulent activities were eventually uncovered through journalistic investigations. In 2001, media outlets began raising questions about the company's practices and suspected its value had been significantly inflated. As investors withdrew their support, Enron's stock price plummeted. On August 16, 2001, the company reported a loss of 618 million USD, prompting the Securities and Exchange Commission (SEC) to launch an official investigation on August 22. In December of that year, Enron filed for bankruptcy (Obringer 2022).

21.3.2 Data Preparation

To standardize the form of words, we perform lemmatization using the WordNetLemmatizer package. All terms were also reduced to a form beginning with 'lowercase'. Moreover, since one of the features designed to detect deception relates to the number of sentences, we replaced URL addresses with the phrase 'URL address'. The standard URL construction includes dots after the www abbreviation and before the domain endings (for example '.com'). In addition, some emails contained the body of messages that the emails respond to. To avoid statistical distortions, only the 'answer' has been analyzed. We have also removed stopwords using nlkt corpus.

21.3.3 Features Engineering

In the feature engineering phase, we utilized both Python programming and the LIWC software to extract five features related to psycholinguistic techniques for detecting deception, as well as additional features indicating the presence of linguistic anomalies. These measures were designed to provide insight into deceptive language patterns in the text data under analysis. As an effect of the feature engineering stage, five psycholinguistics features were invented:

- Number of sentences in a text,
- Lexical diversity ratio calculated as hapax legomenon score,
- Number of references to digits in text,
- Concreteness level – the level of abstraction of text. Text concreteness was calculated using the Reality Monitoring approach. To calculate the Reality Monitoring score, we use the following measures from the LIWC dictionary: percept, time, space, and cogproc.
- Sentiment level – sentiment level calculated with TextBlob() library.

The study presents an approach to detecting suspicious communication with anomaly detection techniques. We assume that emails characterized by a very low or very high level of psycholinguistics features should be treated as ‘anomalies’. With the given approach, we extract twelve additional features:

- red/orange/green flag – concreteness level,
- red/orange/green flag – lexical diversity,
- red/orange/green flag – number of sentences,
- red/orange/green flag – number of references to digits

The values of certain measures may suggest the presence of a lie in the text. Therefore, we assigned a ‘red flag’ to emails with low levels of specific psycholinguistic features (first percentile). Emails with scores higher than average for these features were labeled with a ‘green flag’, while those with scores within the average range were labeled with an ‘orange flag’. This allowed us to quickly identify texts that were potentially deceptive based on the values of these measures.

21.3.4 Subject Analysis

In the first step of the analysis, we used the LDA algorithm to identify email subjects marked with a red flag. Then, it was validated whether the subject of an email is relevant. In the initial stage of the study, we analyzed all of the emails in a dataset; however, a deeper analysis of low-sentence emails showed that these messages are usually not informative. We noticed that emails that were ‘interesting’ were typically long, so the research is focused on them.

Subject Analysis of Red-Flagged Emails due to the Lexical Diversity Metric

In the first step, we analyzed the subject of emails marked with red flags due to the lexical diversity metrics. Figure 21.1 shows a visualization of the 40 most popular email topics extracted with the LDA algorithm in the set of messages marked as ‘suspicious’ due to the lexical diversity metric.

The central cluster is located on the right side of the graph for emails, characterized by a shallow lexical diversity. This cluster was composed of messages discussing energy sales, the energy market, and Enron’s operations in California. These topics are relevant to the company’s core business and may be useful in detecting suspicious communication. The middle and left sides of the chart also contain business-specific topics, which appear less often in the conversations of Enron employees.

On the other hand, in a given group, we also identified advertising messages that contain phrases such as ‘book now’, or ‚‘click here’, as well as error notifications. We can also see that the cluster in the upper right part of the graph consists of emails related to matches played by a corporate sports league.

Subject Analysis of Red-Flagged Emails due to the Number of References to Digits

Figure 21.2 shows the 40 most popular topics among red-flagged emails due to the number of references to digits. The most frequent topics pertain to personal subjects such as holidays and leisure activities. A cluster containing these types of issues can be found in the lower-left

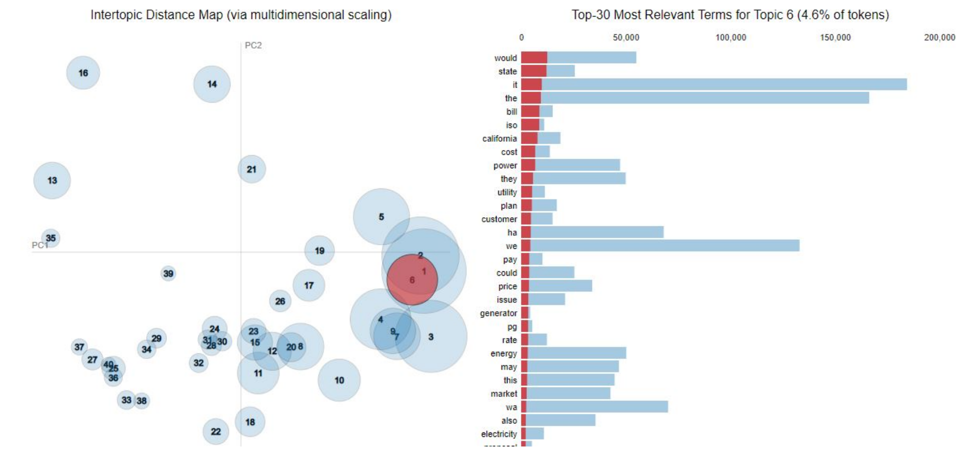


Figure 21.1: LDA visualization – Red flags – Lexical diversity metric

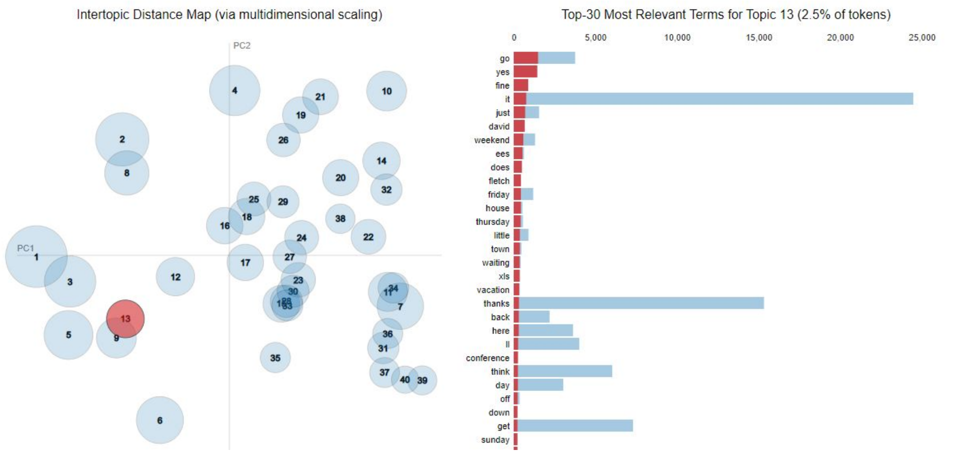


Figure 21.2: LDA – Red flags – Number of references to digits

corner of the chart. In contrast, clusters in the central and right parts of the chart primarily consist of business-related subjects. However, the business-related topics in this group appear to be less specific than those marked with a red flag due to their low lexical diversity.

Subject Analysis of Red-flagged Emails due to the Text Concreteness

The LDA algorithm was used to extract the topics of the selected news sample of red-flagged emails due to the text concreteness metric. Figure 21.3 shows a visualization of the 40 most popular topics.

The largest group of emails with a high level of text abstraction are messages containing salutations, such as 'thanks', 'sure', 'let know', 'could'. The specificity of this type of email implies a low level of text concreteness – messages among these groups are usually irrelevant in fraud detection. More interesting from the perspective of the researched issues would be

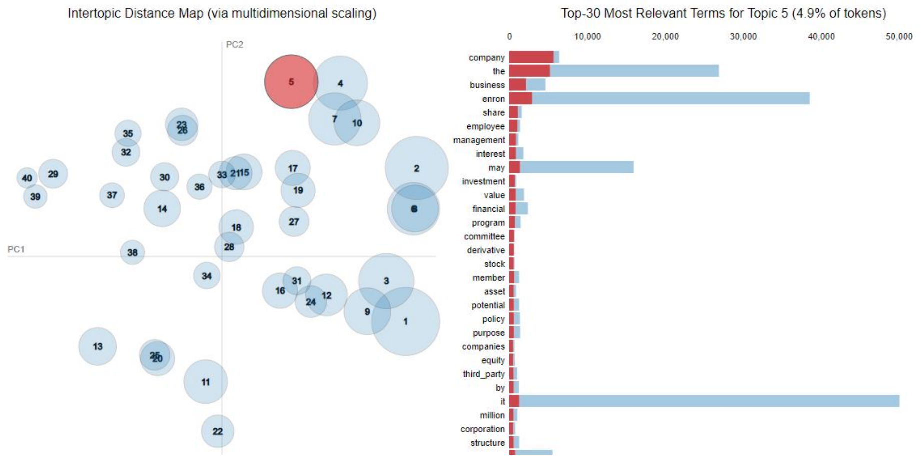


Figure 21.3: LDA – Red flags – text concreteness

emails concentrated in the upper right corner of the chart. This group of emails concerning subjects, e. g., derivatives and financial estimates (related keywords: 'investment', 'value', 'stock', 'derivatives', 'potential'), energy market regulations (related keywords: 'market', 'ferc', 'energy', 'regulators'), or about specific business problems (related keywords: 'issue', 'risk', 'process'). Additionally, this group includes error notifications and marketing messages that may be deceptive in nature. To sum up:

- In the group of emails characterized by an above-average level of abstraction, there are messages containing many polite phrases. Because of the nature of that messages, emails among these groups should be treated as irrelevant in detecting employees' fraudulent activities.
- Given the nature of Enron's business, the majority of messages exchanged by its employees relate to subjects such as securities trading or finance, which are likely to include a large number of references to digits. As a result, filtering the set of emails down to those flagged as suspicious due to their number of digit references results in a significant number of personal messages. This can be an obstacle in identifying messages relevant to fraud detection issues.
- We also identified clusters containing advertisement messages that are deceptive in nature and error notifications irrelevant to the research subject.
- On the other hand, there were identified clusters of business-related emails which should be further analyzed.

Applying the LDA algorithm allows for identifying relevant and irrelevant topics of red-flagged emails. Red-flagged emails belonging to business-related clusters should be considered necessary for further inspection.

21.3.5 Manual Inspection

In the next step, we manually inspected red-flagged and orange-flagged emails related to business topics. In as given section, we present the results of the manual inspection of a newly created subset of emails marked as suspicious due to the descriptive characteristics. The newly created subset poses about 7 per cent of the whole dataset. In the group of red-flagged emails, we identify messages sent by innocent persons and people convicted due to fraud. One of the examples of a suspicious email identified is a message written by Ken Lay in March 2001 in which Ken Lay responded to allegations against Enron related to the company's contribution to the energy crisis in California. CEO of Enron Company denied that Enron played a role in the energy shortage. He described that the frequent occurrence of blackouts resulted from an imbalance between demand and supply in the energy market in California, concealing the truth about the offensive strategy undertaken by the corporation. He also assured that the good interests of Enron's customers were one of the company's priorities. The mail is a blatant lie and an attempt to convince Enron employees of the company's integrity.

Moreover, we identified suspicious emails addressed by Ken Lay, sent after the Enron scam investigation began. In a given group of emails, he tried to convince Enron's employees that there was still a chance for the growth of the company. He also assured that Enron would fully cooperate with the SEC, which could suggest that companies action had been taken with integrity. He operated with phrases such as Cohen (2004) *'As I've said before, we welcome this [SEC Investigation] and look forward to working with the SEC to put this matter behind us. To cooperate so that we can leave this matter behind us'*, which may suggest that the company has nothing to hide and that the ongoing SEC investigation would not reveal significant irregularities or *'Today, we announced another positive development in our efforts to regain shareholder and market confidence, strengthen our balance sheet, and help maintain our credit rating. Know it is discouraging to read the negative media coverage about our company. On a positive note, there was an editorial in last Sunday's Houston Chronicle that I want you to read I would like you to read'* that somehow asserts Enron's strength and integrity. Ken Lay uses deceptive language to convince Enron's investors and employees that – despite allegations that Enron faced – the company's situation was stable.

Except, for the suspicious communication sent by Enron's CEO, we identified emails written with marketing jargon. One example of such email is a message advertising the new Enrons program: *'I am pleased to announce the launch of iBuyit eProcurement within Enron Corporate. Many of you are already using eProcurement. Effective today, employees throughout Corporate will begin to use eProcurement to purchase IT hardware and office supplies. This is a significant milestone in the Enron-wide cost-saving effort. This new tool enables u to watch our expenses and take advantage of preferred suppliers. The use of eProcurement ensures that all of u are spending our company dollars wisely. This is only the beginning, and eProcurement has the potential for unlimited growth. Be sure to provide the iBuyit team with feedback to ensure that eProcurement is a valuable and integral tool to your purchase activities. I encourage you to visit the iBuyit Portal today to logon to eProcurement, to access training information and materials, and to learn more about Enron's business-wide cost saving opportunities:* `http://ibuyit.enron.com`*'.(Cohen 2004)* The analysis of a given group of emails shows that there are messages that, although formally do not contain untruth, are written in a language that meets the assumptions of linguistic deception models. Marketing language has standard features with deception: a) it is aimed to convince the receiver of the messages about the high quality of the product, b) it is sent intentionally, and c) sometimes the authors of the messages try to hide the truth about the product's imperfections. The manual inspection of emails facilitated identifying those related to Enron's fraudulent activity and contained deceptive language. This method proved effective in detecting

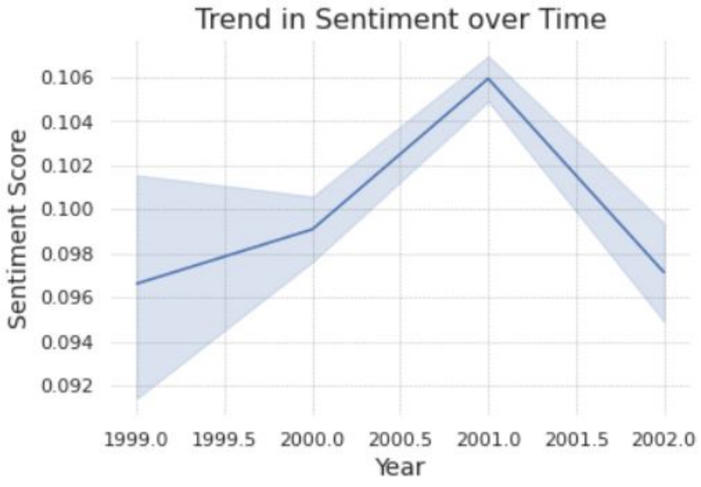


Figure 21.4: Sentiment analysis

suspicious communication and has the potential to mitigate the risk of fraudulent actions within business organizations. However, during this process phase, we also identified emails irrelevant to our research question, such as those written using marketing jargon. This highlights the need for improvement in step 3 by implementing more advanced topic extraction algorithms.

21.3.6 Sentiment Analysis

To validate the efficacy of the proposed method, we conducted two additional experiments related to sentiment analysis. In the first step, we analyze the sentiment level of all emails available in the dataset over the years. The experiment's goal was to check whether the events related to the disclosure of Enron's fraud reflected the sentiment of Enron employees' messages. In the second step, we examine the sentiment polarity of emails marked by green, orange, and red flags, separately to validate if there were differences between particular groups. The sentiment was calculated using the textblob function. Messages with positive sentiment reached a score close to 1, and 'negative' emails approached the lower end of the scale (-1). Emails with a neutral sentiment achieved scores fluctuating around zero.

Figure 21.4 illustrates the sentiment polarity of all messages sent by Enron employees over the years. The sentiment had been slowly increasing in the years 1999–2000, and from 2001 there was a sharp decrease. This decline reflects the behavior of Enron's share prices, shown in Figure 21.5. The period in which the downward trend is observed corresponds to the events related to the disclosure of Enron's unfair practices. Therefore, we can assume that as a result of the scandal, the mood of corporate employees was lower, which reflects email sentiment.

In the second stage of the study, we analyzed email sentiment levels marked with a green, orange, and red flags separately (Figure 21.6). It can be seen that the sentiment level of emails marked with a green flag due to the following metrics: lexical diversity, text concreteness, and the number of references to digits, has been decreasing since 2001. This

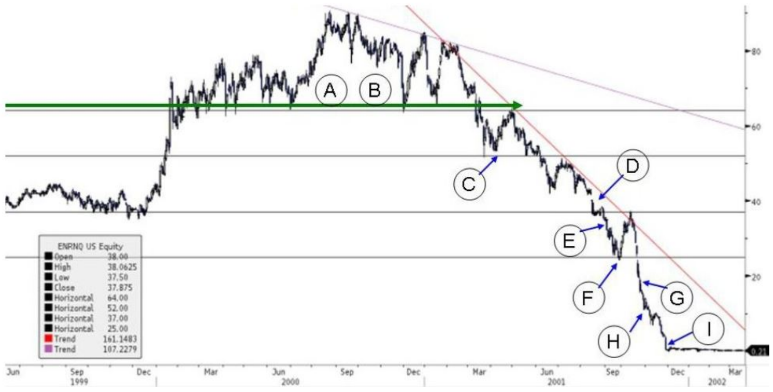


Figure 21.5: Enron stock prices
https://www.famous-trials.com/enron/1791-stockchart

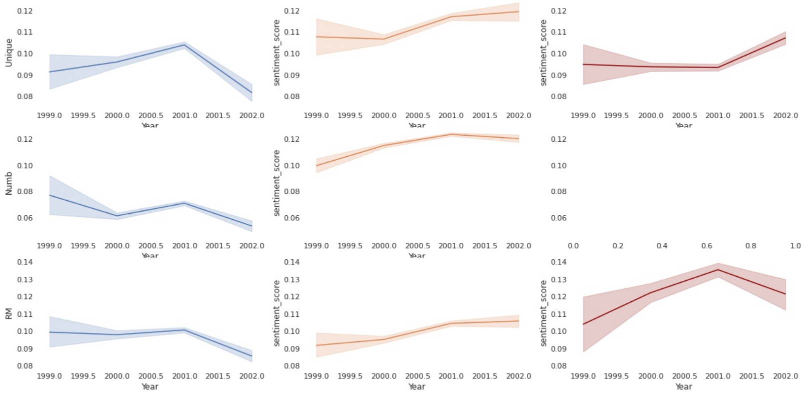


Figure 21.6: Sentiment analysis

behavior corresponds with the general trend observed for a sample of all emails available in the dataset. The decline reflects the events of 2001, which may support the assumption that green-flagged emails are likely, to be honest. An inverse relationship is observed for emails flagged as orange or red – the sentiment level among this group of emails has been growing since 2001.

Results from sentiment analysis of suspicious emails correspond with an observation drawn from the manual inspection step. Enron's management had been trying to convince the company's employees about the stable position of the corporation and underestimated the seriousness of the situation when the SEC investigation occurred. Moreover, management tried to devalue the importance of the SEC investigation by sending emails like: *'I know it is discouraging to read the negative media coverage about our company. On a positive note, there was an editorial in last Sunday's Houston Chronicle that I want you to read'* (Cohen

2004). Convincing employees of Enron's honesty could boost the team's morale; on the other hand - maintaining a positive mood was essential to the strategy adopted by the corporation. As described in Section 21.3.1, Enron's value depended mainly on a 'future revenue' policy and the public's trust and belief in corporate power. Maintaining positive moods after the events in 2001 could be an attempt to protect what was most important for Enron - the illusory conviction of investors about its profitability and innovation. The higher sentiment level of emails marked with an orange and red flag can also be explained – to some extent – by the specificity of some messages among the analyzed group. As mentioned in Section 21.3.5, among the group of suspicious emails, we identified marketing messages that have specificity, which forces a positive connotation and is deceptive in nature.

21.3.7 The Process of Detecting Suspicious Communication – Conclusions

In the study, we have analyzed three groups of linguistic features, that may indicate on presents of lie in text:

- descriptive measures (text length, lexical diversity, number of references to digits),
- text concreteness,
- text sentiment.

We have also enriched the proposed method of using topic extraction alghoritms to identify relevant subjects. The study led to the following conclusions:

- There are significant differences between the group of emails marked as suspicious and messages that do not meet the linguistic criteria of lying. The sentiment analysis confirmed the validity of email classification. There are significant differences in the behavior of both these groups – among emails marked with a green flag, there is a clear downward trend, which is consistent with the behavior of Enron's share prices. In the group of emails that meet the linguistic criteria of lying, an inverse relationship is observed.
- In the group of emails marked with an orange and red flag, we have identified messages containing lies and relating to the company's dishonest activities, which proves the potential of using a given method in systems alerting about anomalies in the organization.

21.4 Future Works and Practical Application

In this experiment, we demonstrated that linguistic methods could be effective in detecting criminal activities within organizations, thereby improving the security of their operations. By analyzing a dataset using psycholinguistic science-based measures, we were able to identify emails that differed from the average messages in the set in terms of the intensity of certain linguistic features. Upon manual inspection, we found that some of these emails likely contained lies and were intended to cover up the company's poor financial condition, potentially indicating fraud.

Despite the promising results of our study, we identified areas that require further investigation in order to enhance the reliability and validity of our findings. These issues suggest the need for future research in order to address these limitations and advance our understanding of the topic. The proposed measures for detecting criminal activity within organizations may be sensitive to the specific context in which they are applied. For instance, in the case of communication from financial companies, the prevalence of numerical references may be higher compared to messages produced by non-business entities. This observation highlights the need to develop diverse linguistic measures that are tailored to the nature of the examined organization's activities.

Additionally, we identified the need to refine step 3 of the process in order to minimize the number of irrelevant texts that require manual inspection. In future research, we recommend exploring the use of more advanced topic modeling algorithms, such as Latent Semantic Analysis or Gibbs Sampling Dirichlet Multinomial Mixture (GSDMM), to assess their potential to improve the efficiency of this step (Qiang et al. 2020). This could potentially reduce the burden on analysts and enhance the overall effectiveness of the process.

Additionally, as previously noted in Section 21.3, the present study did not consider the potential impact of dialects or language variations based on the personality, gender, or age of the email's author. While the incorporation of such an analysis, could improve the effectiveness of the proposed deception detection algorithm, this approach may be more costly to implement.

The proposed method has the potential to be utilized across a variety of fields, including forensic analysis and the optimization of a company's website and social media management strategies. By alerting for linguistic anomalies, this algorithm could potentially minimize the presence of hate speech on the company's professional social media accounts. In the forensic domain, the algorithm may be useful in detecting spam and phishing emails, as well as aiding in the auditing process for fraud detection. Additionally, the algorithm's ability to identify linguistic anomalies makes it particularly useful for detecting fake news. The potential of the proposed algorithm to identify fake news is especially noteworthy due to the current proliferation of fake news and its rapid spread.

References

Adams-Quackenbush, N. (Aug. 2015). 'The Effects of Cognitive Load and Lying Type on Deception Cues'. PhD thesis. Saint Mary's University, Halifax, Nova Scotia, Canada, Department of Psychology.

Barbara, D. (2014). 'A Cognitive Approach to Deception Detection: Multimodal Recognition of Prepared Lies'. PhD thesis. Università degli Studi di Milano-Bicocca, Department of Human Science for Education.

Bel, N., Bracons, G. and Anderberg, S. (2021). 'Finding evidence of fraudster companies in the CEO's letter to shareholders with sentiment analysis'. In: *Information* 12.8, p. 307. DOI: 10.3390/info12080307.

Cabello, A., Moncarz, E., Moncarz, R. and Moncarz, B. (2009). 'The rise and collapse of Enron: Financial innovation, errors and lessons'. In: *Contaduría y Administración* 218. DOI: 10.22201/fca.24488410e.2006.579.

Clark, R. E., Feldon, D. F., Merriënboer, J. J. G. van, Yates, K. A. and Early, S. (2008). 'Cognitive task analysis'. In: *Handbook of Research on Educational Communications and Technology*. Routledge, pp. 577–593. DOI: 10.4324/9780203880869.

Cohen, W. W. (2004). *Enron Email Dataset*. http://www.cs.cmu.edu/~enron/. (Visited on 22/06/2022).

Dyck, A., Morse, A. and Zingales, L. (2013). *How Pervasive is Corporate Fraud?* Rotman School of Management Working Paper No. 2222608, Social Science Research Network. Accessed: June 24, 2022.

Fissette, M. and Vries, T. de (2017). 'Text mining to detect indications of fraud in annual reports worldwide'. In: *Benelearn 2017: Proceedings of the Twenty-Sixth Benelux Conference on Machine Learning, Technische Universiteit Eindhoven, 9-10 June 2017*, p. 69.

Fuller, C. M., Biros, D. P. and Delen, D. (2011). 'An investigation of data and text mining methods for real world deception detection'. In: *Expert Systems with Applications* 38.7, pp. 8392–8398. doi: 10.1016/j.eswa.2011.01.032.

Gandía, J. L. and Huguet, D. (2021). 'Textual analysis and sentiment analysis in accounting'. In: *Revista de Contabilidad* 24.2, pp. 168–183. doi: 10.6018/rcsar.386541.

Ganis, G. and Keenan, J. P. (2009). 'The Cognitive Neuroscience of Deception'. In: *Social Neuroscience* 4.6, pp. 465–472. doi: 10.1080/17470910802507660.

Hauch, V., Blandón-Gitlin, I., Masip, J. and Sporer, S. L. (2015). 'Are computers effective lie detectors? A meta-analysis of linguistic cues to deception'. In: *Personality and social psychology Review* 19.4, pp. 307–342. doi: 10.1177/1088868314556539.

Keila, P. and Skillicorn, D. (Jan. 2005). 'Detecting unusual email communication.' In: IBM, pp. 117–125. doi: 10.1145/1105634.1105643.

Kleinberg, B., Vegt, I. van der, Arntz, A. et al. (2019). *Detecting deceptive communication through linguistic concreteness.* doi: 10.31234/osf.io/p3qjh.

Larcker, D. F. and Zakolyukina, A. A. (2012). 'Detecting Deceptive Discussions in Conference Calls'. In: *Journal of Accounting Research* 50, pp. 495–540. doi: 10.1111/j.1475-679X.2012.00450.x.

Mhundwa, P. H. (2005). 'The relationship between grammar and the psychological processing of language'. In: *Journal for Language Teaching= Ijenali Yekufundzisa Lulwimi= Tydskrif vir Taalonderrig* 39.1, pp. 149–161.

Newman, M., Pennebaker, J., Berry, D. and Richards, J. (June 2003). 'Lying Words: Predicting Deception from Linguistic Styles'. In: *Personality & social psychology bulletin* 29, pp. 665–75. doi: 10.1177/0146167203029005010.

Obringer, L. A. (2022). *How Cooking the Books Works.* Accessed: June 24, 2022. url: https://money.howstuffworks.com/cooking-books.

Pennebaker, J. W., Boyd, R. L., Jordan, K. and Blackburn, K. (2015). *The development and psychometric properties of LIWC2015.* Tech. rep.

Pha, A. (2010). *Enron: Capitalism in a Nutshell.* New Age Publishers, Australia. isbn: 1-876919-13-2.

Pollock, L. (2018). 'Concepts and concreteness in psycholinguistics, Department of Psychology and Language Sciences'. PhD thesis. University College London.

Qiang, J., Qian, Z., Li, Y., Yuan, Y. and Wu, X. (2020). 'Short Text Topic Modeling Techniques, Applications, and Performance: A Survey'. In: *IEEE Transactions on Knowledge and Data Engineering* 14.8, p. 1. doi: 10.1109/tkde.2020.2992485.

Tovmasyan, L. (May 2020). 'Linguistic Features of Lying (L.T.)' PhD thesis. Yerevan State University, Department of European Languages and Communication. doi: 10.13140/RG.2.2.11608.93440.

Van Bockstaele, B., Verschuere, B., Moens, T., Suchotzki, K., Debey, E. and Spruyt, A. (2012). 'Learning to lie: Effects of practice on the cognitive cost of lying'. In: *Frontiers in Psychology* 3.526. doi: 10.3389/fpsyg.2012.00526.

Wang, T. Y., Winton, A. and Yu, X. (2010). ‘Corporate Fraud and Business Conditions: Evidence from IPOs’. In: *Journal of Finance* 65.6, pp. 2255–2292. DOI: 10.1111/j.1540-6261.2010.01615.x.

Weare, C. (2003). *The California electricity crisis: causes and policy options*. Public Policy Instit. of CA.

Wikipedia (2022). *2000–01 California electricity crisis - Wikipedia, The Free Encyclopedia*. dostęp: 28-May-2022. URL: https://en.wikipedia.org/wiki/2000%5C%E2%5C%80%5C%9301_California_electricity_crisis.

Zloteanu, M. (Apr. 2020). ‘Reconsidering Facial Expressions and Deception Detection’. In: *Handbook of facial expression of emotion*. FEELab Science Books & Leya, pp. 238–284. ISBN: 978-84-10-74697-8.

Chapter 22

The Next Generation of ERP Systems: Problems of Traditional ERP-Systems and the Next Wave of Really Standardized ERP Systems

Reinhard Schütte

The research work and publications on ERP systems in the IS community are multi-layered, whereby here the banal attempt of a keyword-based literature analysis shan't be undertaken, being as insightful as a forest rich in sunshine. It is the peculiarity of a Festschrift that scientific freedoms present itself to the authors in dealing with the topic, which are closed in the other scientific publication practice. These freedoms enable the author to work on a topic where access to the subject matter is more difficult, in this case enterprise-wide standard software in general and ERP systems in particular.

22.1 Status Quo of ERP Systems

22.1.1 Introduction

There is a wide range of contributions dealing with teaching and research aspects of ERP systems (for an overview, see Bahssas et al. 2015 and Klaus et al. 2000), however, the question arises whether the nexus to 'ERP systems in enterprises' is sufficiently addressed. Based on more than twenty years of experience in the development and implementation of standard software systems, the author asserts that this is indeed not the case. There is a lack of work dealing with the technical basis of standard software and the interests of standard software producers on the one hand and the introduction and use of standard systems in companies depending on this technical basis on the other hand. SAP software in its different generations from R/2, R/3 up to S/4 HANA can be understood as the prototype of enterprise-wide standard software. Even in its current version, the software comprises up to 400 million lines of code; the reengineering of the old R/3 coding announced by Plattner and Leukert (2015) has failed to materialize, according to the author's assessment.

Accommodating individualizations is a considerable challenge in terms of software technology. Thus, not only the standard, provided upon delivery of the system, is relevant for decision-making from a business point of view, but also the technical capabilities for enhancing and adapting it. This technical environment then becomes even more important when implementing updates from the software manufacturer while maintaining formerly adopted individual developments. In an EDEKA project known to the author, a standard system now has about 4 million lines of code of additional developments that would not have been safely manageable for so long without the features of SAP software logistics using SAP Transport Management. The combination of high-level customization with a business and software standard, as defined in the SAP system, represents an interesting phenomenon that has been realized in many companies. It contradicts the expectation of a standard, by which many still understand a standard such as a norm (e. g., a power socket) or an economic standard such as Microsoft Office 365 or Adobe Acrobat. However, in the case of enterprise-wide standard software, which is generally assessed in the literature as integrated and also parameterizable or 'customizable', there is not one standard, but a variety space. As a result, firstly the parameterization space and secondly the individual developments in the standard system are the factual reality which makes the use of such standard systems possible in the first place and that represent the background against which the following statements are to be understood.

In addition to raising the questions of what a standard can be in the case of company-wide standard software, how standard conformity can be measured, which economic advantage its usage offers assuming it is even technically feasible, it must be asked when and in which corporate cultural context such projects can be successful at all. Why did the SAP implementation at Lidl lead to a write-off requirement of 500 million euros? Often, problem causes are searched for in the software product rather than in the implementation project. While there is no space to discuss such a potentially unreflective assessment comprehensively, it might hypothesized that, in the case of Lidl, it was not the deficiency of the software, but perhaps rather the judgment of a very successful company (and its decision makers) that the standard had to be aligned with the actual world. This would be the assessment of the author, who in this contribution makes use of the scientific freedom of this commemorative publication to adopt a research position, as was also advocated by Malik (2005) in business administration: that of a knowledgeable participant who, due to his experience, can dispense with questionnaire actions that need to be critically evaluated, because he has a far deeper insight due to his experience and contacts (Malik 2005, p. 7).

In view of the importance of ERP systems in companies, it seems to be far more important to address this topic than to deal with sometimes relatively insignificant topics that are accessible by means of simpler scientific methods. Figure 22.1 exemplifies ERP topics that are discussed in information systems research and education. In this paper, the fundamental question is what the claim of enterprise-wide standard software can be, where enterprise-wide is understood here as the use of integrated software across departments and disciplines. The demarcation between ERP, Supply Chain Management or also Customer Relationship Management systems is problematized at a later point.

The questions of the business possibilities and limits of standard application systems in enterprises are illuminated and the corresponding technical basis of ERP systems from the past and the requirements for the future are considered. Thus, the contribution deals with the core question of the organization of ERP systems also from the view of software manufacturers, which is neglected in the literature. However, this is ultimately the basis for all considerations regarding ERP systems.

22.1.2 Historical Perspective: Growing Scope of Tasks in Standard Application Systems

Application software is characterized by the purpose it serves its user. In this context, the size of an application varies greatly; it can support only a single use case – e. g., an app on a smartphone – or it can serve to cover the requirements of almost the entire company. The

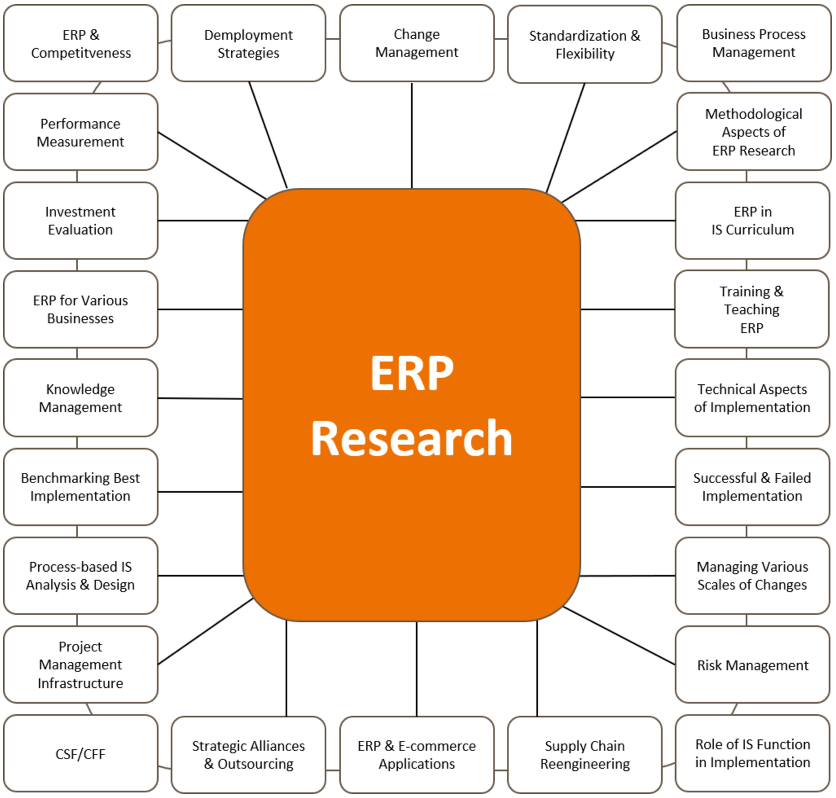

ERP & Competitveness
Deployment Strategies
Change Management
Standardization & Flexibility
Business Process Management
Performance Measurement
Methodological Aspects of ERP Research
Investment Evaluation
ERP in IS Curriculum
ERP for Various Businesses
Training & Teaching ERP
Knowledge Management
Technical Aspects of Implementation
Benchmarking Best Implementation
Successful & Failed Implementation
Process-based IS Analysis & Design
Managing Various Scales of Changes
Project Management Infrastructure
Risk Management
CSF/CFF
Strategic Alliances & Outsourcing
ERP & E-commerce Applications
Supply Chain Reengineering
Role of IS Function in Implementation
ERP Research

Figure 22.1: Research Topics ERP – example according to Al-Mashari 2002

latter is implied in ERP systems and their definitions, which represent an application system type that is in use in almost all companies. *'ERP systems are described as [...] integrated business standard application software packages that support almost all task areas and processes within the company, such as procurement, production, sales, accounting and human resources'* (Abts and Mülder 2017, p. 193). A similar definition is also given by Olsen and Kernashawi:

> *'An enterprise resource planning (ERP) system is a business management system that comprises integrated sets of comprehensive software, which can be used, when successfully implemented, to manage and integrate all the business functions within an organization. These sets usually include a set of mature business applications and tools for financial and cost accounting, sales and distribution, materials management, human resource, production planning and computer integrated manufacturing, supply chain, and customer information'* (Olson and Kesharwani 2010, p. 51).

In contrast to the term 'requirement' commonly used in computer science, which is the subject of requirements engineering as one of the most important phases of software engineering, the term 'task' was dominant in the context of organizational theory in business administration. In organizational theory, a task is the requirement for a task owner to perform an operation on an object in order to achieve (corporate) goals (Kosiol 1976). In the context of enterprise software, it seems important to speak not only of requirements, but also of tasks. Ultimately, tasks are the requirements that must be performed by an application software system. The software is understood as the technical task carrier in contrast to a human task carrier. From the perspective of the production factor theory, this also clarifies that software is always an instance of the production factor capital as opposed to that of labor.

The use of application systems in companies has a long tradition. With the advent of execution of tasks by computers in companies, the trade-off between technology investments and alternative investments solidified. Early on, investment in application systems was made with the expectation that this investment would be preferable. Elsewhere, there has been extensive research on the impact and cost-effectiveness of IT systems and why there are also so many paradoxes in IT systems (Schütte 2019; Schütte et al. 2022, pp. 117–191). This discussion is not to be repeated here, but instead it is to be motivated that a business management view is necessary. Therefore, tasks also form the nexus of application systems that makes business sense, because application systems are used to support them. Requirements can also be those of users that are not based on a business task.

From a business perspective, in the sixties, mainly those tasks were supported with the help of application systems that existed in the area of manufacturing companies (Sarferaz 2022, p. 4). Under the caption of Material Requirements Planning such systems employed different algorithms to determine the future need of parts and components necessary for the production of final products. Further task support then arose in the 1980s under the label of manufacturing resource planning systems (MRP II, Jacobs and Weston 2007, pp. 359–361). MRP II systems have a much more comprehensive planning approach, ranging from sales and program planning to production planning and production control. In particular, the consideration of capacities and the temporal sequence planning of the production form extensions of the software compared to the former MRP systems. MRP II systems have found their expression in operational practice and in business administration as production planning and control systems (PPC), even if MRP II and PPC are not to be understood as synonyms. In parallel, since the beginning of the 1970s, tasks in finance and accounting have been supported primarily by SAP with its R/2 system in the RF (real-time financial accounting) and RK (real-time cost accounting) modules (Wenzel 2001, p. 7).

In a further stage of development at the end of the 1980s, ERP systems emerged from the MRP II systems, in which finance and accounting, human resources management and other tasks were also integrated. Enterprise Resource Planning is already a clear indication of the constantly advancing demand for software systems to support business tasks more and more comprehensively with the help of software systems. The immense demand of such systems is also evident from the many applications that have belonged to the world's most successful ERP software since the mid-1990s, the SAP R/3 system (Figure 22.2).

A 'special role' in IT support in companies has always been played by those systems that support the dispositive factor in the company and for whose task support there have been separate systems with changing names such as Management Information Systems, Decision Support Systems or Business Intelligence. The purpose of each of these systems is similar: to support the dispositive factor in the sense of Gutenberg, i. e., to support employees in decision-making and coordination tasks in the company. These systems were not part of ERP systems. Since the beginning of the 2000s, the focus of ERP systems has changed, because the large (ERP) systems could no longer fully map all new requirements. ERP systems had primarily an internal company focus. Systems with dedicated purposes have evolved since then, for a variety of reasons:

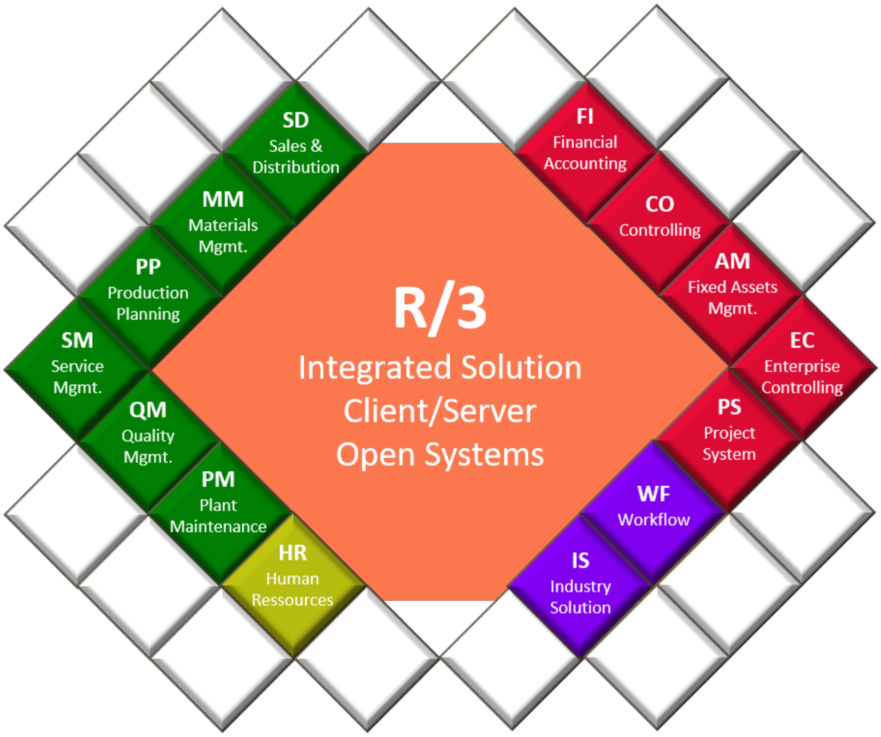


Figure 22.2: Components of the SAP R/3 system as 'ERP prototype' (According to SAP 2023)

Firstly, for reasons of a differentiated approach, support and analysis, so-called *customer relationship management systems* have established as its own product branch, which are not understood here as a component of an ERP system. The classic ERP systems had a considerable deficit in sales and logistics, as exemplified by the changes in the systems in the areas of sales and logistics. CRM systems serve thereby the administrative and the analytical support of tasks in connection with the management of the customer relations (Hippner and Wilde 2006). For the development of CRM systems in business practice, Salesforce deserves to be mentioned: founded at the end of the 1990s, it initially focused exclusively on the development of cloud-based CRM systems.

Secondly, comprehensive, cross-company logistics information systems evolved and found their way into corporate application architectures under the heading of *supply chain management systems* (Kuhn and Hellingrath 2002; Arndt 2008).

Thirdly, with the advent of the internet, transaction phases or complete transactions were implemented via the network using dedicated store systems. Special systems have thus become established which perform analogous functions to classic ERP systems: Quotations for customers, order entry and processing, customer analytics, etc. This means that similar functions have been implemented in the classic ERP system, in a CRM system or in a store system on the basis of the 'same objects' such as materials, customers, prices, etc. At the same time, this development has reduced the requirements of ERP systems, since the same requirements can also be implemented in other applications.

Beyond applications that grew out of CRM, SCM and store systems, there are now other applications for services, marketing, strategic and operational transport management, etc. that represent a renunciation from the one monolithic application system. From what was once just one central system, the ERP system, a multitude of application systems has emerged over the course of thirty years, for which the manufacturers receive license fees (may it be a subscription fee, a one-time payment with subsequent maintenance fees) or SaaS-integrated pay-per-use prices. The software manufacturers offer several applications and these are not subject to the corresponding technical basis. This also has its cause in the corporate acquisitions and the associated solutions of the manufacturers. Products for analytical tasks now extend the Salesforce portfolio, for travel expense accounting the SAP portfolio, or for classic ERP solutions the Microsoft portfolio. However, the development environments, the type of data storage, or even the possibility for further development of the software within the product portfolio are different. In this context, platforms and ecosystems have been established which are relevant for the technical and economic understanding of the software. The different applications (products) which a company uses and which can be assigned to the ecosystem of a software manufacturer nevertheless exhibit technical differences, since they cannot be assigned to a uniform technical platform of a manufacturer.

22.1.3 Historical Perspective: Software Architecture Development of the Standard Application Systems

From a technical perspective, it will be traced which different architectural styles have underpinned enterprise-wide software packages. The development up to the end of the 1980s (Figure 22.3) was characterized by monolithic software developed for mainframe computers with little interoperability and extensibility. At the beginning of the 1960s, when both the material requirements calculations under the acronym MRP – from the point of view of the literature (Section 22.1.2) – as well as the first finance and accounting applications – from the point of view of practice (among others, the point of view of SAP in Figure 22.3), mainframe computers formed the basis of data processing. The software was based on a monolithic architectural style, which was still the defining architectural style in the development of SAP R/2 as the first integrated package of standard application software until the 1980s. The mainframe world was characterized by a very low level of interoperability, because the applications were still tied to the underlying hardware and its operating systems, which included above all IBM's MVS (Multiple Virtual Storage) and its further developments or the BS1000 operating system from Siemens and its successor BS2000.

The problem with the mainframe world was that interoperability was very limited; communication and integration of systems was hardly possible, and the 'binding' of applications to the operating system was very pronounced. This also made it difficult to extend the different applications and also to use them on changed hardware. It was the situation at that time, where the hardware and software had a high connection to each other. It was only on the basis of rulings by American antitrust courts that the separation of software and hardware by manufacturers such as IBM became mandatory. From an economic point of view, hardware and software became independent assets and the idea of SAP's emergence is closely linked to this legal process (Mohrmann 2016, p. 27).

The next generation of ERP systems was based on the idea of client-server architectures (including SAP R/3, which became widespread in 1992 and enabled the worldwide success

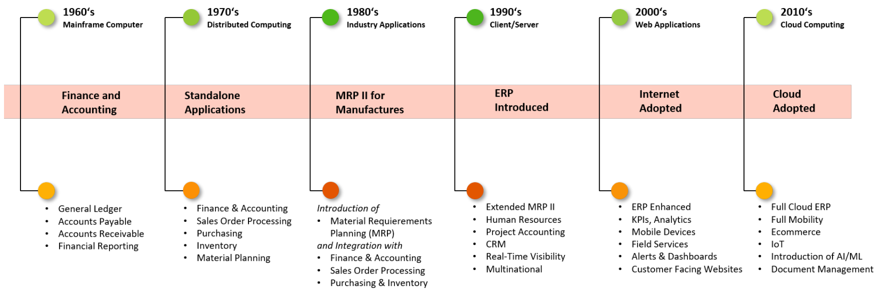

1960's
Mainframe Computer
1970's
Distributed Computing
1980's
Industry Applications
1990's
Client/Server
2000's
Web Applications
2010's
Cloud Computing
Finance and Accounting
Standalone Applications
MRP II for Manufactures
ERP Introduced
Internet Adopted
Cloud Adopted
• General Ledger • Accounts Payable • Accounts Receivable • Financial Reporting
• Finance & Accounting • Sales Order Processing • Purchasing • Inventory • Material Planning
Introduction of • Material Requierements Planning (MRP) and Integration with • Finance & Accounting • Sales Order Processing • Purchasing & Inventory
• Extended MRP II • Human Resources • Project Accounting • CRM • Real-Time Visibility • Multinational
• ERP Enhanced • KPIs, Analytics • Mobile Devices • Field Services • Alerts & Dashboards • Customer Facing Websites
• Full Cloud ERP • Full Mobility • Ecommerce • IoT • Introduction of AI/ML • Document Management

Figure 22.3: Development of enterprise-wide standard software from the perspective of SAP as the market leader for ERP systems (According to SAP 2022)

of SAP as the leading ERP provider. The client-server approach opened up a much better distribution of databases, applications and the graphical user interface to different types of computers. The basic understanding of two- or three-tier architectures is exemplified by the SAP understanding (Figure 22.4). The application servers communicate with the presentation components and the database, and communication between the application servers takes place via the message server. From the point of view of SAP logic, it is important to understand that all software code resides in the database, which is why a separation of database and application server logic is only possible to a limited extent when it comes to a release upgrade.

In a further development stage Application Integration Services were established, which always required a middleware. The system follows a logic as known from the literature with CORBA (Pope 1998; OMG 1999). In the SAP world, it was the SAP Exchange Infrastructure as a hub system (from 2002) that was to bring about a decoupling of systems and applications. Among other things, this made it possible for several SAP systems with different content characteristics (e. g., logistics and finance and HR) to be installed, so that these systems with dedicated tasks could have different SAP releases. In the evolution from modern to composable ERP architectures, as described by Gartner (2020) (Figure 22.5), it becomes clear that even such a separation of components through an integration platform does not yet mean that the architectures have the necessary extensibility and reusability; in addition, the requirements for the Web were not yet adequately mapped at first.

Service Oriented Computing (SOA) evolved, in which a more granular separation of applications was to become possible from an object perspective. Standardized objects were formed, defined by the Object Management Group, but an ‘order’ object is not in itself a solution for loosely coupling applications, as recommended in the service-oriented approach. The underlying technologies did not allow sufficient flexibility at that time, and the question of how to implement them for large systems remained unsolved. The business question of which functions belonged to which object was also not answered in the end, and there was no ERP company that was strived to consistently reimplement the software on an SOA basis. In the meantime, other approaches had already emerged which, as with Salesforce, offered software directly via the web, and the SOA approach was not yet favored for cross-application communication.

The SOA development has in any case initiated the development of so-called postmodern ERP systems, which are characterized by communication with the ERP core via APIs while

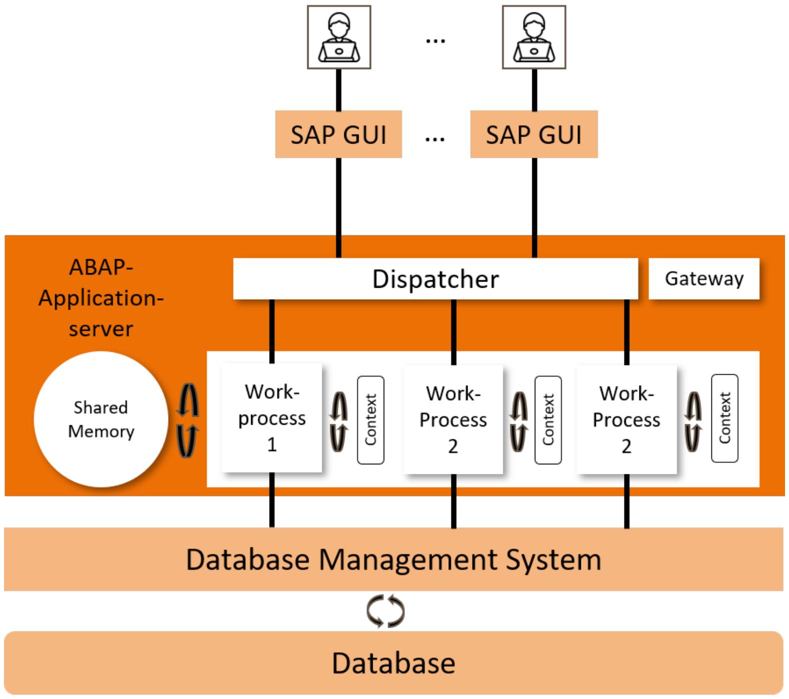


Figure 22.4: Three-tier client-server architecture with the central tasks of the SAP Application Server (SAP 2020)

operating on-premise or in the cloud as software as a service. This beginning of postmodern ERP systems, which Gartner dated at 2014, led to composable ERP systems, which have were set to dominate the following ERP system evolution since 2020, according to Gartner's idea at the time.

In recent decades, comprehensive enterprise systems evolved. Regarding the type of software, these systems share their intended suitability for a class of application situations, i. e., the software is to be used in many different enterprise contexts. This is the characteristic of standard software, which requires specific methods for software development, software introduction, and software use as well as ultimately entails the need to think about the technical artifact 'application software' substantially different. The traditional way of development, particularly the way it is taught in our discipline today as the dominant form of requirements and software engineering, is that of the individual software, that is individual software which is created for the specific (business) task of a single company (Kohncke 2015, p. 36).

The enterprise systems and their components are to be adapted as standard software systems first for the specific needs of the respective organization, in which it is supposed to be used for, as part of the on-boarding projects comprising parameterization and self-developments. In the case of the technical artifact, which is to be standard software, it is presupposed here that it is also a software, which was planned ahead in its design and its implementation for a class of use cases. This may not be the case for all systems that

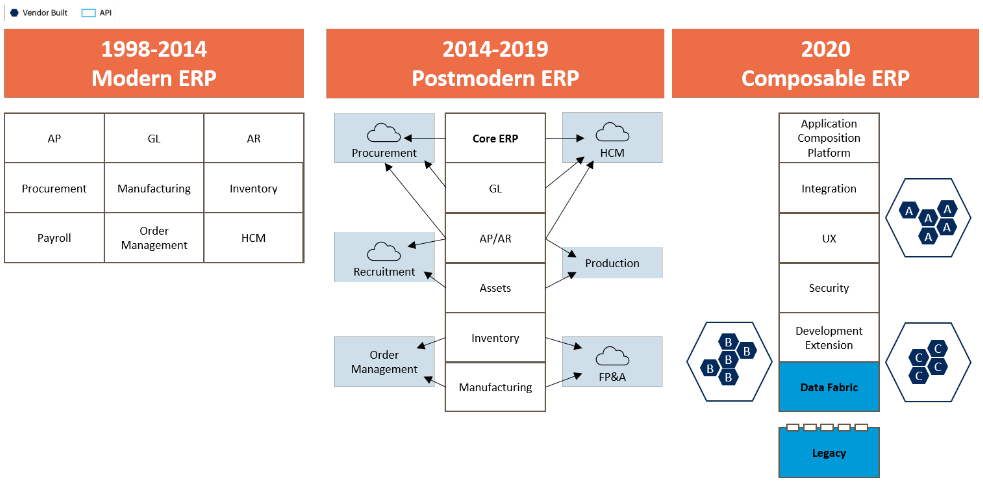


Figure 22.5: Evolution of the architecture of ERP systems according to Gartner 2020

claim to be standard software. In the case of 'genuine' standard software, care is taken right from the design stage of the software that, firstly, there are customer-specific customization options which do not jeopardize the core of the standard software itself (i. e., do not require any intervention in the coding of this software itself) and that there is a constant updating process for the software which is expressed in release and patch versions.

22.2 Enterprise Systems

22.2.1 Conceptual Unfolding of Enterprise Systems

In the literature, ERP systems and enterprise systems are sometimes used synonymously, or ERP systems are described as confusing because of the 'erroneous' reference to MRP and MRP II systems and the unrecognizable resource reference (Klaus et al. 2000). Other authors reject the term ERP directly and use that of enterprise system instead (among others, Davenport 2000), which has also established itself in its own conference series. Here, the assessment is shared that it is historically difficult to restrict the origin of ERP to the material or production sector and that this is also of little explanatory value for the systems that exist today. Therefore, a substantive derivation from a more general system-theoretical frame of reference as well as a classification of existing systems in the software world seems far more appropriate for an application-oriented science. Furthermore, for the subsequent discussion of a further development of existing systems (both from the point of view of the manufacturers and the domain companies), it is necessary to orient oneself to existing systems and not to choose terminology in which the classification of real objects is made difficult. Would Salesforce be an ERP system according to the traditional understanding? Should it be disregarded in its relevance for the entire software market just because it cannot be assigned to the diffuse terminology? The perspective of systems theory can be taken to analyse the essential components of a comprehensive enterprise system, which also takes into account the existing software product situation of vendors and domain companies. Assuming an application system understanding on the abstract level, the system

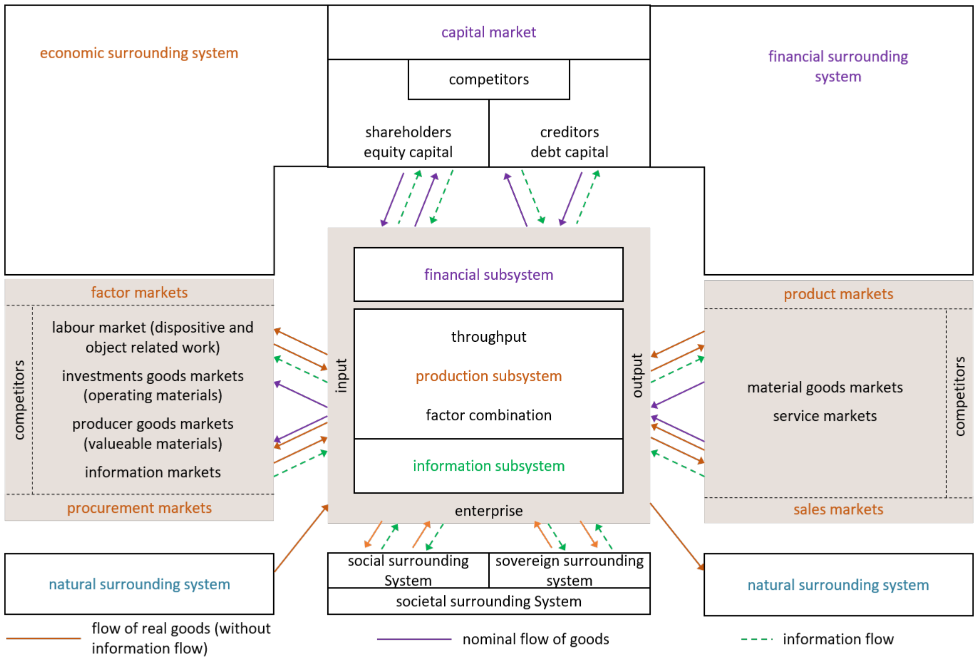

capital market
economic surrounding system
financial surrounding system
competitors
shareholders equity capital
creditors debt capital
financial subsystem
factor markets
labour market (dispositive and object related work)
investments goods markets (operating materials)
producer goods markets (valueable materials)
information markets
competitors
procurement markets
input
throughput
production subsystem
factor combination
information subsystem
output
enterprise
product markets
material goods markets
service markets
competitors
sales markets
natural surrounding system
social surrounding System
sovereign surrounding system
societal surrounding System
natural surrounding system
flow of real goods (without information flow)
nominal flow of goods
information flow

Figure 22.6: Enterprises as open systems with a performance-based and financial subsystem (adapted from Zelewski 2008, p. 63)

perspective in Figure 22.6 offers to decompose the enterprise systems of today's reality into six components (Figure 22.7):

- A classic *ERP system* that primarily supports the internal tasks in companies and usually supports the components human resources, accounting and controlling as well as purchasing, sales, production and logistics tasks. Depending on the company, such an ERP system already covers all areas of the organization. However, this is likely to be the case only in simpler application situations. Instead, in more complex organizations, additional applications have been established in the last twenty years to cover the entire range of requirements of companies in the sense of comprehensive enterprise systems. Therefore, from a historical perspective, an ERP system is also understood as a core application of today's enterprise systems.

- Many online business activities have been established since the advent of the internet, which are supported with the help of standalone *store systems* focused on these purposes. There are now standalone applications to support B2B or B2C business in many companies and also from many standard software manufacturers such as SAP, Microsoft, Oracle or Salesforce, without which the domain companies would have no software support for online business.

- In addition to these two systems, *customer relationship management (CRM) systems* have established themselves as a special system at the interface between the company and the sales markets. These are offered as cloud solutions by Salesforce or SAP with

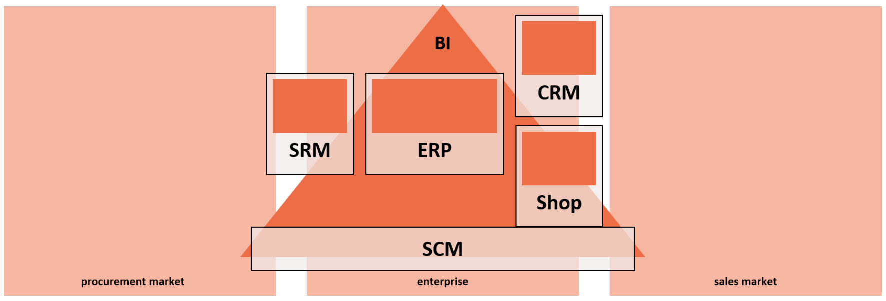


Figure 22.7: Components of a comprehensive enterprise system

the Sales Cloud, or as on-premise solutions for supporting various special customer relationship management measures such as campaign management.

- Dedicated *supplier relationship management (SRM) systems* exist at the interface to the procurement markets, which comprise *'the IT-supported design of strategic and operational procurement processes and supplier management emanating from an overall procurement strategy'* (Appelfeller 2020, p. 258).
- From a domain perspective, a special position is occupied by the *supply chain management (SCM) system* (Kuhn and Hellingrath 2002, pp. 142–156; Wannenwetsch 2014, pp. 489–518), which not only maps the narrow focus of a company but also serves the (strategic and operational) planning and control of the entire logistics chain; this means that suppliers (upstream producers) and transport service providers in particular are also included in the considerations. Sometimes, the SRM system is also understood as an integral part of an SCM system.
- Finally, BI systems designed for analysis purposes and thus to support the dispositive factor in companies form a sixth component, which is the subject of comprehensive enterprise.

22.2.2 Enterprise Systems Boundaries

There are various criticisms of enterprise systems or ERP systems in the literature, such as the poor support of functions and processes, the complexity of the processes, the complexity of the application systems themselves, the insufficient standard processes or the low support for inter-organizational activities (Frank and Strecker 2009, p. 1; Markus et al. 2000; Markus and Tanis 2000, Shang and Seddon 2007, Rettig 2007, Davenport 1998, Davenport 2000. The assessment from the information systems literature is conceptual in the sense that most of the work does not address the technical characteristics of software systems. The required brevity of the paper does not allow for a detailed discussion of enterprise systems assessment, because the literature always refers terminologically to ERP systems, which in this form, however, do not represent (enterprise) application architectures alone. Given the heterogeneity of what an ERP system can be in reality, generalizations on such an inhomogeneous basis hardly seem possible. Moreover, the distinction between a software product before it is deployed in a company and the software product changed by a project is rarely made. Thus, many enterprises started with the desire in projects to introduce the standard software in the sense of standard and in the course of the projects, which usually lasted several years, a change of the software took place. The use of ERP software does not reflect to what extent standard and to what extent changes have been made to the system. There are older empirical studies that document the enormous change in SAP R/3 systems, where there is a considerable deviation from the standard in the areas of sales and logistics. Thus, considerable adjustments have been made in the sales systems and in logistics and hardly any adjustments are made to the systems in the finance and human resources domains (Buxmann and König 1997).

The use of company-wide standard application systems faces considerable presuppositions that must first be unfolded. The fundamental suitability of a standard for a social system as a superordinate assumption, the institutional economic preferability of 'standard software' and the economic efficiency and strategy perspective must be problematized and placed as presuppositions in front of the following explanations.

From a sociological perspective, the success of off-the-shelf application systems such as SAP S/4 HANA, Microsoft Dynamics, or Oracle Cloud ERP is almost surprising to see, because each company is individual in its patterns of interaction and in the sense that companies should be. *'For many social scientists, especially those informed by sociology and anthropology, the large software suppliers like SAP should not be successful. Sociological/anthropological theory tells them that organizations are too diverse to deploy these highly generic systems. Many studies therefore end up suggesting – on the basis of difficulties and complications witnessed during fieldwork – that ERP systems and the like have no more than limited potential for extension'* (Pollock and Williams 2008, p. 8). It is also emphasized elsewhere: *'More generally, as concerns rise concerning the incommensurability of systems and contexts, there are demands for solutions that are already partially adapted to particular business settings (i. e., "semi-generic" packages) and for increased user-involvement in the shaping packages'* (Pollock et al. 2003, p. 318; see also Davenport 2000). Thus, it is to be assumed for the further discussion first that there is no incommensurability problem of social reality of an enterprise and the employment of a business application software, which prevents the employment of this software (contextual applicability presupposition). Here, context subsumes the linguistic, cultural and social dimensions that must be reconciled with the software and that, in turn, may prevent the use of the software.

From a new institutional economics perspective, which was brought forward by Ciborra with enterprise-wide databases (Ciborra 1985, p. 64; Ciborra 1987, p. 31, in the context of the reference modeling Schütte 1998, p. 7–9), the problem of the standardizability is to be discussed, with which it comes to a paradoxical situation. With the new institutional economics on basis of behavior assumptions (bounded rationality, opportunism and use maximization) among other things in the context of the transaction cost theory, assets are either exchanged over the market by the price coordination mechanism or in enterprises by the governance coordination mechanism. Thereby, specific, uncertain, and rare transactions are to be provided in an enterprise according to the theory itself. According to this, non-specific, i. e., standardizable services with a high degree of certainty and a frequent transaction are not produced in the company, but are procured from the market. Regarding the discussions on enterprise-wide standard application software, this distinction would mean that the enterprises themselves would not be entitled to exist any longer because such assets are to be obtained over the marked rather than produced in an enterprise. Or, the other way around, the existence of standard software would mean that the transaction cost theory would be empirically disproved. A similar illustration of tasks in standard software and above all their use in the enterprise means that the tasks are standardized and not specific and that the modalities of the task execution are subject to hardly any uncertainties (regardless of the general environmental uncertainty). Moreover, the use of these tasks, technically implemented as standard in software, in domain companies would mean that the tasks are performed frequently, otherwise automation would be less likely. Following traditional economic theory, the implementation in the standard software itself would also only be economical if multiple use is given. In summary, the tasks are thus not specific, not uncertain and frequent, so that, following new institutional economics, the execution would have to be done via the market. It comes to the paradox of the standard application software: The employment of 'a standard' in many enterprises leads – under assumption of the transaction cost theory – to the abandonment of the economic raison d'être of the enterprises, since the completion over the market would be more efficient. From the existence of the enterprises, which use the same standard application software, only two conceivable conclusions can be drawn. Either it can be concluded that the transaction cost theory does not apply and is to be rejected as theory, or there is not this one standard over the enterprise borders. It is assumed here that the transaction cost theory could not be falsified. Also, for further remarks, an *enterprise existence presupposition* is assumed, reaching beyond enterprise boundaries with the employment of standard application software.

From a strategic perspective, it must be discussed whether a cross-company standard is strategically possible and sensible. In the context of discussions of enterprise-wide application software, a standard is not referred to as a norm, as in the case of a power socket, but as a standard that has been or is being established in a domain. A distinction can be made between a descriptive and a prescriptive understanding of a standard. The one who de facto sets the standard is the manufacturer of the software. The business solution or solution space that can be used for a class of companies has not been identified by means of a recognized method, but is the factual, economically motivated determination of the standard software manufacturer. Therefore, the 'best practices' frequently propagated by standard software vendors also appear to be more of a marketing message that is unlikely to stand up to serious scientific scrutiny and discussion (Wagner et al. 2006). The core question is whether the processes existing in a company of a domain (e. g., an industry) can be so similar that they can be mapped in a standard application software. As an anchor point for this discussion a quotation of Porter shall serve, who understands the effects of the chosen generic competitive strategies of companies on the design of activities in a company as follows:

> *'Ultimately, all differences between companies in cost or price derive from the hundreds of activities required to create, produce, sell, and deliver their products or services, such as calling on customers, assembling final products, and training employees. Cost is generated by performing activities, and cost advantage arises from performing particular activities more efficiently than competitors. Similarly, differentiation arises from both the choice of activities and how they performed. Activities, then, are the basic units of competitive advantage. Overall advantage or disadvantage results from all a company's activities, not only a few'* (Porter 1996, p. 38).

If one follows Porter's statement that the strategies of companies are reflected in the activities of the company, analogous company-wide systems would mean that the companies also pursue the same strategy. The strategies of Aldi and Lidl as discount grocers are not comparable with those of EDEKA and Rewe as supermarket operators; nor are the strategies of VW and Mercedes or Porsche identical.

Thus, following Porter's thesis, entirely different tasks would have to be supported in the respective standard application software – in the value-creating, primary activities. Empirical studies prove the small standard use in the ranges such as logistics, production and selling (Mauterer 2002; Buxmann and König 1997 and/or the enormous meaning of the adaptability of an enterprise system for the competitive ability (Almutairi et al. 2022, p. 748). In business practice, the author has accompanied many medium-sized and large ERP implementations and gained knowledge of other projects in which the extent of customization of standard systems has been considerable. This tendency will intensify with the increasing digitization of the value chains of enterprises, because with it ever more activities differentiating opposite the competition (and thus other strategies) will be possible only under inclusion of software systems. Then almost every classical organizational problem, *'determination and coordination of tasks based on the division of labor (...)'* (Picot 1991, p. 144) will become a problem — always also an information-technical problem.

For the design of enterprise systems, then, the dream of the configurational world has also come to an end: against the background that competition is always a discovery process, it seems theoretically hardly possible for companies to obtain their solutions from standard software producers in time if these are to have a competitively differentiating effect. It will hardly be possible to identify the necessary differences ex ante from the software manufacturers, to map them in solution spaces so that companies can build on them. These fundamental considerations, which are not to be unfolded here theoretically from the view of the strategic management, the competition theory and politics further, indicate in each case that the requirement of standard application manufacturers is limited, which can be realized in a system as standard.

Above all, the important question for software manufacturers is what can be standardized across company boundaries and what cannot. As a rule, according to the author's experience, requirements are collected in co-innovation projects or other coordination circles with customers and tested for their generalizability as well as for feasibility and also 'saleability' in the standard system.

The author is aware of many examples in the SAP trading solution that were already demanded as elementary requirements in the mid/late nineties by the pilot customers at that time and that have only now been implemented in the new SAP S/4 HANA solution. For 20 years, the requirements had not been taken into account (e. g., multi-level articles, a purchase order and delivery schedule). In view of the required brevity, this problem resulting from the behavior of software manufacturers cannot be discussed further for standard application software, but it is permissible to point out that it involves various principal-agent problems in the ecosystems of software manufacturers as hubs, the consulting and implementation companies, and the domain companies. In summary, it is assumed that it is impossible in the case of standard business software in the value-creating differentiation areas to make the requirements of companies fully and immediately available (*strategic implementation presupposition*).

The existence of standard application systems in companies also assumes at the same time that their use is economically advantageous, because otherwise their use would not be rational. The sometimes minor economic success of the introduction of standard application systems, which has been discussed in the course of the productivity paradox for information technology, will not be traced here (Schütte et al. 2022). It is assumed that there is a form of business standard application software that can be used economically in a company (economic *efficiency presupposition*).

22.3 Enterprise Systems Development Trends

22.3.1 Basic Preliminary Considerations

It is assumed here that standard business software can only be interpreted as standard if it has a 'minimum degree of standardization' for the activities and processes of a company. Otherwise, it would be possible for standard software from Microsoft, for example, to be used on the surface, but this statement would only be correct with regard to the technology used, provided that perhaps only 50% of the intended processes and data objects are used. In this case, there is a technological basis for the individualization, but no process-related standard. Without addressing the problems of a degree of standardization in this article, it should be assumed that at least 90% of the requirements should be covered by the software so that one can speak of standard software. In domains such as human resources or finance and accounting, such a minimum standard is usually achieved. This also raises the question of whether the understanding of 'generic standard systems' presented in the literature (Klaus et al. 2000; Dorn 2000; Brehm et al. 2002; Ettlie and Perottie 2002; Gronau 2021) is a target-oriented model for ERP systems. The technical realization of these requirements in the software is referred to in the majority of the literature as customizing (Gronau 2021, p. 28; Kurbel 2013, p. 167; Dorn 2000, p. 201; Klaus et al. 2000), which at this point should be understood in the narrower sense as parameterization. The scope of parameters here is considerable (Dittrich and Vaucouleur 2008) and this parameterization assumes that the enlarged code base with an overlying parameterization layer brings advantages. Such a target picture of standard software assumes that it is possible to achieve high levels of standardization by having different requirements that are realized in the standard software by expanding the code base. The problem can be outlined by quoting the former CIO of ALDI Nord:

> *'[...] The introduction was largely driven by replacing the old solutions with a new solution. In other words, there was less of a strategic focus on how to optimize processes directly, but rather a need for a solution that would continue to function. And that's why SAP was taken as the standard. But only very few people know that this standard can be defined very broadly and that there are also many options for customization. In the end, we went and listened very closely to the existing needs of the respective departments or country-specific requirements and did not standardize, but rather customized to a high degree. If you compare this and look at the coding, about 70 percent of the code we use today is of an individual nature. That is, actually developed or modified ourselves, in SAP parlance, and only about 30 percent of the code sections are taken over from the software, from the manufacturer [...]'* (Hoepfner 2021).

From the outlined conceptual standardization problems of enterprise-wide application systems in the primary activities, which particularly shape the enterprise and become unique, it is presupposed here that there cannot be the standard or the standard solution space in business management terms. The former claim of standard software is not a conceptually desirable goal. Instead, it is assumed that there is a core functionality that can be represented as a standard by software in a domain. Business-administrative systems are excluded from this. This core functionality will, as the later technological possibilities in cloud native computing will indicate, lead to fundamentally differently designed and realized systems. The guard rails under which development are to take place are unfolded below from a business management and a software technology perspective.

22.3.2 Business Guard Rails of Future Development: The Perspective of the Domain

From a business perspective, four aspects are of particular importance: today's level of digitization in general, the changed customer approach and expectations, the division of labor between human and machine, the integration problem (as a facet particularly emphasized in the past) of company-wide systems, and the process perspective:

Digitized scope of tasks: Iin the course of digitization, value creation today is not to be achieved solely on the basis of the material product, but many (informational) services and the development of companies toward platforms must be taken into account (Schütte and Wulfert 2022). Not only are preliminary products purchased, products designed and manufactured, and sold to customers, but it is combinations of information, services, and material entities that shape the value creation of companies today. As a result, the volume of services that are company-specific will continue to increase. Due to their criticality for the success of the company, these requirements must be developed quickly in order to satisfy customer needs in a timely manner.

Customer approach and expectations: This is also expressed in the special importance of application systems at the interface to customers, so that one could also speak of 'customer-facing IS'. In the course of digitization, IT has increasingly reached customer interaction. Examples such as car-sharing services, in-store retail apps (inventory information in the app, intelligent in-store navigation) or the booking of parking spaces and their direct billing via the car manufacturer may be enough to enable a different kind of customer support across the four transaction phases (search, agreement, conclusion, post-contract phase). This reduces the ERP system requirement because the customer approaches are likely to be far more company-specific.

As a result of the increase in customer-facing IT, companies have also developed the requirement that there is no longer a single customer, but that the individual customer should ultimately be addressed in the sense of 1:1 marketing. It is a question of digital customer representation in the system, to which enormous added value is attributed. The customer is to be mapped in the system as a digital twin, as it were, for the purpose of customer analytic IT, in order to provide him at all times with the services and recommendations that are recommended from the point of view of the customer and the company addressing him. This means that analytical workloads are also particularly time-critical (in the range < 500ms), so that they can be used, for example, as part of recommendation engines. The classic BI architecture in the context of enterprise systems may thus become obsolete for various tasks. There will probably be a reduction in BI requirements and a shift to proprietary solutions for BI services.

In addition, the customer has high expectations for performance fulfillment, because during the transaction phases, companies are expected to have a level of information that never shows vulnerabilities. The classic backend functionalities concern the customer and these systems are expected to operate 'at planet scale', otherwise there would be an immediate impact on business performance from the company's point of view. Thus, zero downtime requirements and other technical deliverables are derived directly from business challenges.

Division of labor between human and machine: An 'instantiation of the world' is predicted, i. e., the classic way of seeing and working in companies is oriented towards objects. These objects were modeled and implemented as classes, while IoT scenarios require instance handling (e. g., running shoe of a single person) analogous to the previously required representation of the customer in an application system. Currently, this business requirement (defining services for the wearer of the one pair of running shoes) is implemented in IoT solutions, which sits alongside the traditional transaction logic of an ERP system. The business implication from instantiation is that there are much more detailed, instance-related processes to consider.

The significant increase in the degree of digitization of companies has also led to a different proximity of IT to the dispositive factor, to management. Far more decisions traditionally attributed to the dispositive factor can also be perceived by application systems. These decisions have a very high added value and are particularly context-dependent and highly company-specific. Decision automation therefore requires highly company-specific algorithms, which leads to an 'individualization' of business software for internal tasks.

Integration problem of company-wide systems: The integration problem of application systems forms a special challenge from a business perspective as well as the technical requirements derived from it. What is the value of integration today and in the future? Basically, decision makers assume that the applications offered by a manufacturer are integrated, which is not technically the case because the systems are sometimes bought in. The still existing desire for low data redundancy contrasts with the need to decouple development activities. From the perspective of existing application architectures, there is usually one dominant software vendor (e. g., SAP, Microsoft, Oracle, or Salesforce), but the number of applications from different software vendors has increased significantly compared to the golden age of ERP systems (ca. 2000–2010). This shifts the integration problem from the physical level of a database to a conceptual level in system design. The integration problem, seemingly solved by the manufacturers of software, will then also have to be focused on in view of the business management developments in companies outlined above.

Process perspective: A major driver of the success of ERP systems since the early 1990s has been the emergence of *process orientation.* The various management approaches have led to the corporate world being structured in terms of processes when ERP systems were introduced. Ultimately, system implementations have been the impetus for process-based reality in organizations, rather than isolated process management approaches (Davenport 2000). A primacy of processes has implicitly emerged on the user and business unit side, which is also expressed in the manifold modeling approaches and the extensive standardization of process description using BPMN. This development has run parallel to the change in software architecture, which structures the world more in terms of services (Section 22.3.2). This is in stark contrast to the way software is developed. The question arises, how the process orientation on the one hand and the object, service orientation on the other hand conception can be brought in agreement. The danger of an overly far-reaching, technically motivated service orientation lies in the neglect of cross-functional and cross-application processes.

22.3.3 Information Technology Guard Rails of Future Development: The Perspective of Software and its Development

The existing application landscapes of the companies are characterized by different systems, as already outlined. An SAP ERP is in use with Informatica for Product Information Management, Workday for Human Resources, Salesforce for CRM and a Magento web store, etc. Enterprise Architecture Integration Technologies are usually used to communicate and decouple the applications. In the best case, the further development of the applications follows a structured release process in the companies, which has to be aligned with the different release cycles of the software manufacturers. The releases in the company form the central synchronization points via which the different further developments in the applications involved are integratively tested and taken productive in the company. Hasso Plattner, the founder of SAP, reported at SAP's own SAPPHIRE conference that the average SAP customer uses an SAP release that is about six years out of date. There are significant conflicting goals between the software vendor and the domain company. For the standard software manufacturer, such a long period is enormously costly and, at the same time, further development of the software is hindered, while the domain companies do not always view the benefit from the productive use of a new release positively and therefore 'skip' some of the software's life cycles, which also has its cause in the outlined individualization problem of standard systems. This application reality is the result of a continuous historical process and is characterized by high static and dynamic complexity. Thereby the maintenance costs of the software systems are always higher than the creation costs: the accumulated maintenance costs increase with the number of releases in the time interval to be considered and the relative increase in value from the software with the number of releases is likely to decrease in view of traditional the marginal utility processes. With such a cost and benefit trajectory of the software deployed, companies have no need for ongoing release-induced change in their software.

The new basis of cloud native computing: From a technological point of view, there are currently some requirements that also concern the development state of the software systems. First, cloud computing as the central megatrend in the IT industry continues to be effective (*cloud presupposition*), so cloud native application systems are being developed for cloud computing. Cloud computing is understood here to be scalable, code-based on-demand provisioned standard computing resources that are provided from a shared resource pool and billed according to consumption (Figure 22.8). Cloud native applications are not just applications that are operated in the cloud. They are applications that are designed with the characteristics of cloud computing in mind (Figure 22.8) and whose characteristic features are: Container Virtualization and Orchestration, Published API Contracts, Private and Polyglot Persistence, Event Streaming, Infrastructure Code, Test and Delivery Automation

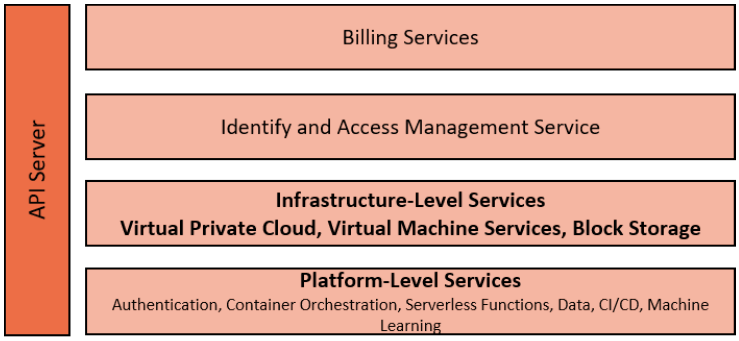

API Server
Billing Services
Identify and Access Management Service
Infrastructure-Level Services
Virtual Private Cloud, Virtual Machine Services, Block Storage
Platform-Level Services
Authentication, Container Orchestration, Serverless Functions, Data, CI/CD, Machine Learning

Figure 22.8: Characteristics of cloud computing

on Immutable Architecture. In this context, the main aim is to reduce the value chain of IT itself on the basis of standardized technologies. In a certain sense, the cloud is the industrialization of IT itself; just as the assembly line ushered in the mass production of automobiles in its day, the cloud has the same effect of accelerating the creation of software solutions and reducing the creation costs of the software.
Service-Oriented Architectures:

- Decoupling development through services: However, in order for the benefits of cloud computing to be realized for applications, there needs to be a greater decoupling of application systems as well as a reduced size of the applications themselves. For this reason, a strengthening of the microservice architecture is expected for the development of company-wide standard systems as an architectural basis, which is already being used for some components of complex enterprise systems (e. g., established in store systems). The thought of the encapsulation, which originated for decades from the object orientation, is realized by the definition of services. Each microservice (or microservice set) is implemented in its own container image. There is complete, logical resource isolation (isolated file system, isolated port range, isolated software dependencies, isolated environment variables). Reproducibility by packaging all dependencies including libraries, drivers, application files into the container becomes possible. A code-based (and thus testable) build of the container image can be completely realized. While resource isolation allows independent deployments, logical dependencies remain.

- Easy extension and communication through published APIs: According to the original idea of the microservice approach, the data store of each microservice is private and uses a service-specific data model and database technology, and the communication of each microservice is realized through a specified and published API offered via network. This provides a technical solution that seems to fulfill the promises of agility: independent deployability of each module (vs. monolithic deployment of all modules) to enable 'frequent deliveries' per dev team. This aspect is of paramount importance given the release issues discussed above. The partial independence (which can only be realized for minor changes and patches, while for a change declared as major in Semantic Versioning the integrative tests outlined at release change become necessary before going live). The special meaning of API exposure should be briefly explained. Application Programming Interfaces offer means of synchronous communication between service and client (client = external client or other service), where contracts regarding data formats exist and guarantees regarding backward compatibility and minimum deprecation periods based on Semantic Versioning are available. The interface also represents an atomic test object where the use of cross-service API gateways is also possible. There is a single point of entry for centralized security, identity enforcement and billing. Approaches toward logic and data composition as well as traffic management (proxying, routing, balancing, quoting) require careful consideration during the API design. Distribution across services promises advantages at high loads due to individual scalabability, yet independent scalability across services cannot create object independence. The advantage of the independent deployabilty does not exist, if breaking changes of the API are present, since that an adaptation of all clients is necessary. Thus, major version bumps entail the need for complicated release schedules, diminishing the promised advantages of the concept. Furthermore,

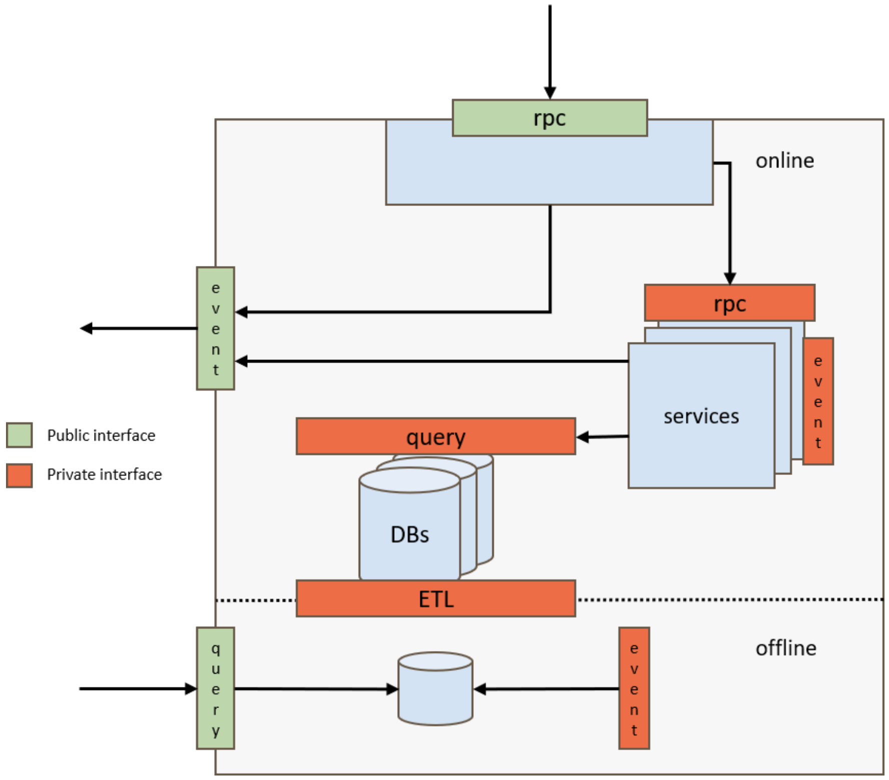


Figure 22.9: Grouping of services in a service domain with shared gateway service and shared data storage (Source: https://eng.ubez.com/microservice-architecture/, 10.09.2022)

private data storage is not always desirable, as redundancies might arise or demands for referential integrity between entities cannot be enforced, or cross-service joins become infeasible. The integration possibilities formerly particularly associated with ERP systems are thus considerably reduced, and the limited technical connectivity must then be solved by conceptual model considerations during system design. In addition, domain-neutral requirements, which are, however, of an overarching nature from the point of view of the individual services, must be solved through the use of cross-service API gateways (Figure 22.9).

- Agnosticity of the backend from the frontend: In addition to the previously outlined technical properties, which are primarily concerning the development of functionalities, it will be necessary for the fulfillment of the business developments from Section 22.3.2 to consider the usability of application systems and the user experience they evoke to satisfy special and also company-specific requirements (referred to as headless in the Mach approach, see Mach 2023).

In summary, this follows the approach of a ‘composable ERP approach’, as Gartner described it in a study (Gartner 2020). In such an approach, which has the technical characteristics outlined above, the MACH principles are to be adhered to (Microservice, API First, Cloud Native, Headless (Mach 2023). The guiding principle is that there are solutions from software

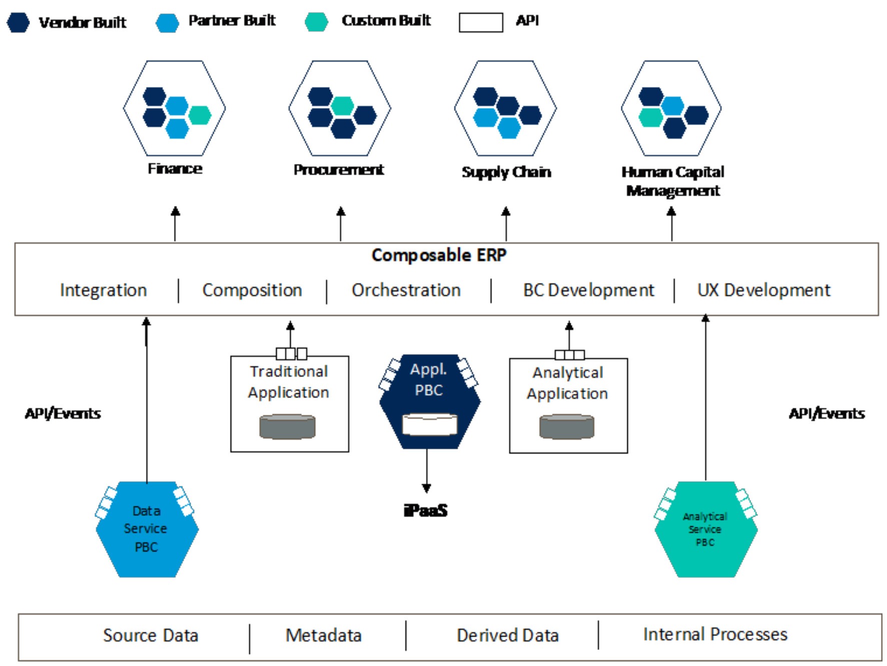


Figure 22.10: Gartner's Composable ERP Approach (Gartner 2020)

manufacturers in the domains (e. g., procurement) which are enriched by those provided as services by partners of the manufacturer (blue pentagons in Figure 22.10), usually participants in the corresponding ecosystem. Solutions from vendors and partners can also be enriched by those from customers (green pentagons in Figure 22.10). The solutions of the partners and the domain companies themselves are created with the help of a Platform as a Service and use the APIs of the respective solutions of manufacturers, partners or customers.

22.4 Outlook

The enterprise-wide standard application systems are facing a significant challenge because the software manufacturers must fundamentally change their existing systems against the background of the changing information technology world and also the application reality in companies, or completely new systems must be developed. The world of standard software of the past will no longer exist, there will be a redimensioning of the core to the 'Pudels Kern', in reference to Goethe's Faust. The options often offered when standard application systems were introduced will no longer exist in this form.

The notion of service orientation and loosely coupled systems faces various hurdles in reality. Conceptually, it is not always possible to clearly assign objects to a service. The customer or the material or the employee are required in many services, and the question arises as to what methods are anchored, and how and where, when it comes to defining services. This fundamental question has been open for a long time and was not satisfactorily solved by the OMG with its business objects in the mid-1990s, among others. Due to the widely accepted dominance of traditional ERP systems, the most important objects were shaped by these ERP systems; this is also why they form the nexus and backbone of the system landscape of many companies. In any case, the customization power of the enterprise will have to be provided by the vendor's ecosystem rather than being predisposed in the software product itself. The new solutions will be exposed via APIs and technical individualization solutions will be created with the help of the respective platform and its services (PaaS). This means that the partners in the ecosystem will become much more like development partners, and will take place beyond the standard software manufacturer ecosystem.

From the perspective of the software manufacturers, it must be stated that only the new developments will lead to the 'New ERP Systems' actually representing standard software in the value-added areas, as is the case with Microsoft Office. However, this will define a standard from each manufacturer that has a business core, the true essence of systems. This can lead to a new competition for the content of the solution and for its preparation for the user. However, the procedural depth of the solutions will be decisive in the end because it is easy to build a CRM system in a standardized way, and much more difficult to implement billing in the EU, Switzerland or Eastern Europe with all its processing modalities. The domain is likely to be of particular importance, and domain-oriented architectures will become more significant for integration. Integration does not only take place on the infrastructure level, but above all on the domain level (e. g., in data models or APIs). There are currently no dedicated explanations or recommendations on the domains from the cloud providers. It would be particularly important for developers who are anchored in such domains to know what the corporate API is that every developer works with on a daily basis. There will be significantly more in-house development in the domains than ever before, despite the growth of available technologies and because of custom development (or via ecosystem solutions) to meet domain-specific technology needs and compete in the market.

References

Abts, D. and Mülder, W. (2017). *Grundkurs Wirtschaftsinformatik. Eine kompakte und praxisorientierte Einführung*. 9. erweiterte und aktualisierte Auflage. Wiesbaden: Springer.

Almutairi, A., Naeem, M. and Weber, G. (2022). 'Understanding enterprise systems adaptability: an exploratory survey'. In: *Procedia Computer Science* 197, pp. 743–750.

Appelfeller, W. (2020). 'E-Supplier Relationship Management und die digitale Transformation der Beschaffung'. In: *Handbuch Digitale Wirtschaft. Band 1*. Ed. by T. Kollmann. Wiesbaden: Gabler, pp. 257–281.

Arndt, H. (2008). *Supply Chain Management: Optimierung logistischer Prozesse*. 4. Aufl. Wiesbaden: Springer Gabler.

Bahssas, D., AlBar, M. and Hoque, R. (2015). 'Enterprise resource planning (ERP) systems: design, trends and deployment'. In: *The International Technology Management Review* 5.2, pp. 72–81.

Brehm, L., Heinzl, A. and Markus, M. (2002). 'Tailoring ERP systems: a spectrum of choices and their implications'. In: *Proceedings if the 34th Annual Hawaii International Conference on Systems Sciences. Track: Organizational Systems and Technology, Minitrack: ERP System Issues and Answers, Island of Maui (Hawaii, USA), January 3-6, 2001*. URL: https://www.

researchgate.net/publication/2952201_Tailoring_ERP_Systems_A_Spectrum_of_Choices_and_their_Implications_Lars_Brehm (visited on 01/10/2023).

Buxmann, P. and König, W. (1997). 'Empirische Ergebnisse zum Einsatz der betrieblichen Standardsoftware SAP R/3'. In: *Wirtschaftsinformatik* 39.4, pp. 331–338.

Ciborra, C. U. (1985). 'Reframing the Role of Computers in Organizations. The Transaction Cost Approach'. In: *ICIS 1985, Proceedings 9*, pp. 57–69.

Ciborra, C. U. (1987). 'Reframing the Role of Computers in Organizations. The transaction cost approach'. In: *Office Technology and People* 3, pp. 17–38.

Davenport, T. H. (1998). 'Putting the Enterprise Into The Enterprise System'. In: *Harvard Business Review* 76.4, pp. 121–131.

Davenport, T. H. (2000). 'The future of enterprise system-enabled organizations'. In: *Information Systems Frontiers* 2.2, pp. 163–180.

Dittrich, Y. and Vaucouleur, S. (2008). 'Practices around customization of standard systems'. In: *Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2008, Leipzig, Germany, Tuesday, May 13, 2008*. Ed. by L. Cheng, J. Sillito, M. D. Storey, B. Tessem, G. Venolia, C. R. B. de Souza, Y. Dittrich, M. John, O. Hazzan, F. Maurer, H. Sharp, J. Singer and S. E. Sim. ACM, pp. 37–40. DOI: 10.1145/1370114.1370124. URL: https://doi.org/10.1145/1370114.1370124.

Dorn, J. (2000). 'Planung von betrieblichen Abläufen durch Standardsoftware – ein Widerspruch?' In: *Wirtschaftsinformatik* 42.3, pp. 201–209.

Ettlie, J. and Perottie, V. (2002). 'The adoption of enterprise resource planning (ERP) systems'. In: *15th Triennial World Congress, Barcelona, Spain*. URL: https://folk.ntnu.no/skoge/prost/proceedings/ifac2002/data/content/00199/199.pdf (visited on 01/10/2023).

Frank, U. and Strecker, S. (2009). *Beyond ERP Systems: An Outline of Self-Referential Enterprise Systems. Requirements, Conceptual Foundation and Design Options*. ICB-Research Report 31. Essen: University of Duisburg-Essen, Institute for Computer Science and Information Systems.

Gartner (2020). *The Future of ERP Is Composable*. Tech. rep. Gartner. URL: https://www.gartner.com/en/documents/3991664 (visited on 10/12/2022).

Gronau, N. (2021). *ERP-Systeme. Architektur, Management und Funktionen des Enterprise Resource Planning*. 4. Auflage. Berlin, Boston: Walter de Gruyter.

Hippner, H. and Wilde, K. (2006). *Grundlagen des CRM. Konzepte und Gestaltung*. 2. überarbeitete und erweiterte Auflage. Wiesbaden: Gabler.

Hoepfner, G. (2021). *Interview von Guido Hoepfner zur SAP-Einführung bei ALDI Nord durch Reinhard Schütte*.

Jacobs, F. and Weston, T. (2007). 'Enterprise resource planning (ERP) – A brief history'. In: *Journal of Operations Management* 25, pp. 357–363.

Klaus, H., Rosemann, M. and Gable, G. (2000). 'What is ERP?' In: *Information Systems Frontiers* 2.2, pp. 141–162.

Kohncke, O. (2015). *User acceptance of enterprise-wide standard software. Theory, influencing factors, and recommendations for action*. Wiesbaden: Springer.

Kosiol, E. (1976). *Organisation der Unternehmung*. Wiesbaden: Gabler.

Kuhn, A. and Hellingrath, B. (2002). *Supply Chain Management. Optimierte Zusammenarbeit in der Wertschöpfungskette*. Berlin, Heidelberg: Springer.

Kurbel, K. (2013). *Enterprise Resource Planning and Supply Chain Management. Functions, Business Processes and Software for Manufacturing Companies*. Berlin, Heidelberg: Springer.

Mach (2023). *The MACH Alliance*. URL: https://machalliance.org (visited on 10/01/2023).

Malik, F. (2005). *Management. Das A und O des Handwerks.* Frankfurt/M.: Frankfurter Allgemeine Buch.

Markus, M. L., Petrie, D. and Axline, S. (2000). 'Bucking the Trends: What the Future May Hold for ERP Packages'. In: *Information Systems Frontiers* 2.2, pp. 181–193.

Markus, M. L. and Tanis, C. (2000). 'The Enterprise Systems Experience: From Adoption to Success'. In: *Framing the Domains of IT Research: Glimpsing the Future Through the Past.* Ed. by R. W. Zmud. Cincinnati, OH: Pinnaflex Educational Resources, Inc, pp. 173–207.

Al-Mashari, M. (2002). 'Enterprise resource planning (ERP) systems: a research agenda'. In: *Industrial Management & Data Systems* 102.3, pp. 165–170.

Mauterer, H. (2002). 'Der Nutzen von ERP-Systemen. Eine Analyse am Beispiel von SAP R/3'. Dissertation. Wiesbaden: Technische Universität Berlin.

Mohrmann, H. (2016). *Das Projekt SAP. Zur Organisationssoziologie betriebswirtschaftlicher Standardsoftware.* Bielefeld: transcript.

Olson, D. and Kesharwani, S. (2010). *Enterprise information systems. Contemporary trends and issues.* New Jersey et al.: World Scientific.

OMG (Oct. 1999). *The Common Object Request Broker: Architecture and Specification. Rev. 2.3.1, OMG Doc. formal/99-10-07.* Tech. rep. Object Management Group, OMG.

Picot, A. (1991). 'Ökonomische Theorien der Organisation. Ein Überblick über neuere Ansätze und deren betriebswirtschaftliches Anwendungspotential'. In: *Betriebswirtschaftslehre und ökonomische Theorie.* Ed. by D. Ordelheide, B. Rudolph and E. Büsselmann. Stuttgart: Poeschl, pp. 143–170.

Plattner, H. and Leukert, B. (2015). *The In Memory Revolution. How SAP HANA Enables Business of the Future.* Berlin: Springer.

Pollock, N. and Williams, R. (2008). *Software and Organisations: The Biography of the Enterprise-Wide System or How SAP Conquered the World.* London, New York: Routledge.

Pollock, N., Williams, R. and Procter, R. (2003). 'Fitting Standard Software Packages to Non-standard Organizations: The "Biography" of an Enterprise-wide System'. In: *Technology Analysis and Strategic Management* 15.3, pp. 317–332.

Pope, A. (1998). *The CORBA Reference Guide.* Reading (Mass.): Addison-Wesley.

Porter, M. (1996). 'What is Strategy?' In: (November-December), pp. 37–54.

Rettig, C. (2007). 'The Trouble with Enterprise Software'. In: *Sloan Management Review* 49.1, pp. 21–27.

SAP (2020). *ABAP Applikationsserver.* URL: `https://help.sap.com/saphelp_snc700_ehp01/helpdata/de/fc/eb2e8a358411d1829f0000e829fbfe/content.htm?no_cache=true`.

SAP (2022). *Ein kurzer Abriss – die rasante Entwicklung von ERP.* URL: `https://www.sap.com/germany/insights/what-is-erp.html` (visited on 10/12/2022).

SAP (2023). *The Evolution of SAP in Three Pictures.* URL: `https://blogs.sap.com/2012/11/27/the-evolution-of-sap-in-three-pictures/` (visited on 08/02/2023).

Sarferaz, S. (2022). *Compendium on Enterprise Resource Planning Market, Functional and Conceptual View based on SAP S/4HANA.* Cham: Springer.

Schütte, R. (1998). *Grundsätze ordnungsmäßiger Referenzmodellierung. Konstruktion konfigurations- und anpassungsorientierte Modelle.* Wiesbaden: Gabler.

Schütte, R. (2019). 'Paradoxien der Nutzung von IT-Systemen'. In: *Dokumentation der Jahreskonferenz 2017 des Netzwerks Verbraucherforschung.* Ed. by B. Blättel-Mink and P. Kenning. Berlin, Heidelberg: Springer, pp. 59–84.

Schütte, R., Seufert, S. and Wulfert, T. (2022). *IT-Systeme wirtschaftlich verstehen und gestalten. Methoden – Paradoxien – Grundsätze*. Berlin, Heidelberg: Springer.

Schütte, R. and Wulfert, T. (2022). 'Digital platforms and trading companies: the evolution of traditional business ecosystems into integrated digital business ecosystems'. In: *Handbook on Digital Business Ecosystems. Strategies, Platforms, Technologies, Governance and Societal Challenges*. Ed. by S. Baumann. Cheltenham, Northampton: Edward Elgar, pp. 212–231.

Shang, S. and Seddon, P. B. (2007). 'Managing process deficiencies with enterprise systems'. In: *Business Process Management Journal* 13.3, pp. 405–416.

Wagner, E., Scott, S. V. and Galliers, R. (2006). 'The creation of 'best practices' software: myth, reality and ethics'. In: *Information and Organization* 16.2006, pp. 251–275.

Wannenwetsch, H. (2014). *Integrierte Materialwirtschaft, Logistik und Beschaffung*. 5. Auflage. Berlin, Heidelberg: Springer.

Wenzel, P., ed. (2001). *Business applications with SAP R/3. an introduction including customizing, ABAP/4, Accelerated SAP (ASAP), Project System (PS)*. Braunschweig, Wiesbaden: Vieweg.

Zelewski, S. (2008). 'Grundlagen'. In: *Handwörterbuch der Betriebswirtschaftslehre*. Ed. by H. Corsten. München, Wien: Oldenbourg, pp. 1–97.

Chapter 23

Process Management: A Reflection from a Practical Perspective

Stefan Jablonski and Stefan Schönig

Process management is an already established method and technology for enacting processes in organizations. Both, academia and professional practice, are subscribing to this statement for certain. Nevertheless, it is very revealing to recall the last years of research and practical usage of process management approaches and draw conclusions from that. This is why we like to share our fragmental and very personalized insights into research on the one hand side and professional practice on the other hand side. We intend to motivate both academia and professional practice to further develop and enact process management concepts since we are firmly believing in this approach and would like to see this domain emerging further. Especially, we like to foster a broader dialogue between these two disciplines.

23.1 Introduction

Process management is a widely accepted approach to organize collaboration between distributed process participants. Since about three decades there are enormous research activities in that domain. Also practical systems have been evolving. Nevertheless, we have experienced a gap between concept and method development in the research domain and enactment of those ideas in praxis. In this discourse we like to reveal and discuss some of those gaps. Thereby, we are pursuing two perspectives. On the one hand, we especially look into the modeling phase of a process application, whereby we do not neglect the other phases of a process lifecycle completely. On the other hand, we adopt the perspective of a software engineer that has the obligation to enact a process management application in an application-oriented setting. Finally, our first goal is to stimulate vendors of process management tools to more enrich their systems by integrating practical and useful research concepts. We do this since we know from our academic background that there are powerful and interesting concepts already developed that are still not available in practical tools but are extremely helpful when enacting practical process applications. Since not all aspects of process management are already researched perfectly, our second goal is to motivate researchers to continue research on these still open issues and not to neglect them because they are not most current.

We structure our discourse into two main sections. In Section 23.2, the fundamentals of process management are introduced in a nutshell, only just to sketch the main characteristics and challenges of process management. Those issues are necessary to know in order to be able to follow the discussion in Section 23.3. This argument critically presents major issues, mostly deficiencies, that we have experienced in practical projects due to the lack of functionality of available tools for process management.

23.2 Process Management in a Nutshell

A business process is a set of activities that are performed to achieve a business result (Dumas et al. 2013). *Business Process Management (BPM)* is a managing approach to goal-driven design and the optimization of diverse and cross-organizational business processes using methods and tools for the design, execution, management, and analysis of business processes.

23.2.1 Process Lifecycle Phases and Perspectives

BPM is not a one-off activity, but a continuous cycle. BPM life cycles from literature typically comprise the following phases: Starting with the *process identification*, where the goal is to create a new or updated process architecture or process map, and the *process discovery* with modeling the as-is process, the *process analysis* follows including the identification, documentation, and quantification of problems and opportunities. Within the *process redesign* phase inferred improvements are analyzed and combined into a to-be process model. After *process implementation* the last step of the cycle is computer-based execution of models supported by the use of a *Business Process Management System* (BPMS) leading to *process monitoring* to measure process performance against selected metrics to assess compliance with the implemented processes.

Without going into detailed discussions we attach major importance to process modeling, independently whether it is happening in the process discovery or the process redesign phase. The outcome of process modeling, the process models, embodies the basic of process execution and thus for a successful and adequate process based application. Due to the central role a process model is playing the question of its correctness and adequacy is arising. There are at least two perspectives for answering this question. A first perspective stems from theory and examines whether formal aspects of a process model are sustained, e. g., avoidance of deadlocks and guarantee of liveness. Besides, we consider the pragmatical perspective even as more important. This issue aims at the adequacy, applicability, and effectiveness of a process model. Its adherence cannot be formally proved but must be assessed by domain experts.

Independent from the lifecycle phases a business process can be divided into complementary perspectives (Curtis et al. 1992; Jablonski and Bussler 1996): *Functional perspective:* The functional perspective identifies a process step and defines its purpose. Also the composition of a process is determined by this perspective. *Data perspective:* The data (flow) perspective defines data used in a process and the flow of data between process steps. Business documents and other objects which are used within activities as well as local variables of the process may be reflected by pre- and post-conditions of activity execution. Typically, process data is passed into and out of applications through interfaces, allowing manipulation of the data. *Organisational perspective:* The organisational perspective defines agents, for instance users, roles, who are eligible and/or responsible to perform a process step. The personal information can be enriched by group and role memberships. *Behavioural* *perspective:* The behavioural perspective defines causal dependencies between process steps (e. g., step B may only be executed after step A). Often these dependencies are called control flow.

According to literature (Curtis et al. 1992) and to our experiences in many practical projects conducted the above introduced perspective constitute the backbone of a process model. Depending on the application domain certain perspectives are of fundamental importance, other perspective might even be neglectable. Besides, further perspective can be added to a process modeling language in case this perspective is fundamental for describing behavior and characteristics of a process (Jablonski and Bussler 1996). We'll discuss examples for such a new perspective in Section 23.3.

23.2.2 Types of Process Models

Current research in the field of business process management has pointed out that a useful process-aware information system must exhibit a trade-off between supporting people in achieving their goals and granting them as much freedom as possible at the same time (Ter Hofstede et al. 2009). In this context, a business process that restricts people to a lesser extent is characterized as a *flexible* process (Sadiq et al. 2001). Concerning process modeling, a flexible process must comprise more eligible execution paths than a rigid one.

Following the terminology of programming languages, there are two paradigms of describing business process models: the *imperative* and the *declarative* style. The imperative way corresponds to imperative or procedural programming where every possible path must be foreseen at design time and must be encoded explicitly. If a path is missing then it is considered not to be allowed. In declarative modeling, on the other hand, only the undesired paths and constellations are excluded so that all remaining paths are potentially feasible; they do not have to be foreseen individually a in advance. As the so-called flexible type of business processes incorporates many often unforeseen paths, the declarative approach is best suited for it (Fahland et al. 2009).

Traditional notations for business process modelling like Business Process Model and Notation (BPMN)[1] rely on an explicit encoding of sequential, alternative, and parallel paths. As a result, every possible flow must be known at design time. Such imperative models are well-suited for more rigid and strict processes. BPMN represents the industry state of the art of procedural modelling. As a result, rigid routine business processes can be covered by offering execution support for BPMN models.

The Case Management Model and Notation (CMMN)[2] represents recent efforts to standardize declarative business process modelling. Instead of relying on an explicit flow, event-condition-action (ECA) rules constrain the entry and/or exit of activities within the model.

23.3 Application Areas of Digital Process Management

This section is oriented towards the application and usage of process management technology in various application domains. We are referring concrete practical projects with partners from industry and from the public sector. In all projects we have been responsible to enact a process-based application. We summarize our experience with the deployment of

1 https://www.omg.org/spec/BPMN/2.0

2 https://www.omg.org/cmmn/

professional process management systems and discuss positive and negative insights. We focus our observations on the field of process modeling whereby also the other phases of the process lifecycle are considered partially. Each case of the presented case studies is addressed according to the following issues:

- characterization of the use case
- discussion of the applicability of available products and the adequacy of standards
- observations, recommendations, and conclusions

23.3.1 Personell-Driven Administration Processes

A first use case stems from the project PRIME[3] with three partners from industry and the public sector beyond academic partner institutions. Besides other goals, one major focus of this project is to model processes in a municipality. Since the examined processes are pretty rigid an imperative modeling style is suitable. Therefore, a BPMN-based process management system is chosen for process modeling and execution. A work group of this project is especially focusing on modeling the organizational perspective of the processes under consideration, since it is most important that the right contact of the public services are allocated to the citizens' inquiries.

Before we are going into the discussion of this modeling task, we recapitulate the modeling primitives of the BPMN standard for the organizational perspective. BPMN offers two modeling elements for the organizational perspective: pools and lanes. Pools represent global participants of a process. Lanes are partitioning pools and describe groups of people, sometimes called roles, that are eligible to perform some of the tasks. Lanes can be further structured into sub-lanes to finer determine potential process performers. We shortly denote the eligible persons identified by a lane as group. It has to be mentioned that these organizational elements are not considered by the execution engine of a BPMN tool; they are just illustrating organizational issues on a modeling level. When examining the requirements towards the organizational perspective of the processes of the municipality we reveal certain critical issues:

- Due to a whole bunch of legal regulations – what is typical for a public authority – many different groups are responsible for executing the steps of a process. Since each group requires an extra lane, the diagram not just grows horizontally – according to the length of a process – but also grows tremendous vertically according to the multitude of groups. Taken together, the process diagram obtains a size that diminishes readability and thus comprehensibility drastically. Both issues are a major requirement towards a model of a real world entity.
- Pools and (sub-)lanes somehow assume that organizational structures are predominantly hierarchies. Nevertheless, in that authority we experience a lot of so called *working teams* spanning various hierarchically ordered departments. Positioning those *transverse* organizational units equally with normal (sub-)departments as lanes under a pool (or superordinate lanes) somehow causes uncomfortableness since it is a totally different form of classification. Besides, the model structure hardly can be comprehended.

3 The project has been financed with funding provided by the Federal Ministry of Education and Research and the European Social Fund under the "Future of work" programme and managed by the Project Management Agency Karlsruhe (PTKA). The authors are responsible for the content of this publication.

- Many tasks in public authority requires the strict observance of dual control, i. e., the four-eyes-principle, or similar restrictive organizational rules. Actually, there is no comprehensive and catchy way of modeling, i. e., representing, issues like that through pools and swimlanes, respectively. As a result such situations are paraphrased by cumbersome pseudo groups and thus are not very illustrating.

What are the inevitable consequences from the above observations? We regard two conclusions as most relevant. First, in general a model is seen as a communication means between domain experts and engineers (computer specialists) (Bialy et al. 2017). Summarizing the above observations BPMN-like process models – being based on a swimlane notation for organizational issues – cannot meet this requirement at all. Taking into account that domain experts are the only involved participants that can deeply evaluate the correctness of a process model from a pragmatical view, a major drawback is identified. Second, the standard but also most of the available professional tools are missing functionality being able to express sophisticated organizational issues as described before (e. g., separation of duty).

It is interesting to look into practice and explore how practitioners work around these deficiencies. Since pools and swimlanes are not prescriptive for the implementation of BPMN processes but are just *decoration* they abuse them in a certain way. This especially happens by modeling organizational issue very coarse-grained. This in turn entails that process models cannot be validated correctly and completely by domain experts since they are not representing the subsequently enacted processes.

For enacting sophisticated organizational assignments process implementer code this logic completely disconnected from the organizational settings of a BPMN process model, i. e., from pools and swimlanes. In Russell et al. (2005) such an implementation is called *workaround* since it is not an integrated component of a process management system. Thus, it is not globally available for modeling and validating organizational issues.

What could be a solution for the ascertained deficiency? Due to our long-term experience both in academia and in industry we put the following forward for discussion. There are research approaches out there that have developed decent ideas how to model complex and sophisticated organizational issues (Jablonski and Bussler 1996). Other approaches show how to integrate the organizational perspective into a process model abstaining from a swimlane notation (Cabanillas et al. 2015). Besides, Russell et al. (2005) shows that several process management systems have been offering sophisticated modelling elements for organizational issues already. However, it is just a minority at the market.

One obvious consequence drawn from these observations is to adopt those exemplarily depicted research ideas and product features by the standard and/or by modern process management systems. For instance, the cited research approaches suggest notations for modeling the four-eyes-principle or arbitrary organizational structures beyond strict hierarchical ones. Although these approaches might still not be well-engineered, they do form a profound basis for practical deployment. Besides, past implementations of powerful organizational issues in former process management system show their feasibility. Nevertheless, in case these approaches from academia and industry are still not matured and extensive enough, this could be a motivation for research to re-consider this research domain and refine and extend these pioneering concepts and ideas.

23.3.2 Decision and Knowledge Intensive Processes

A second example is taken from the manufacturing area.[4] One of our project partners is producing special-purpose machines. This production process is characterized by two things. First, each machine is manufactured through piece production and widely does not resemble other machines. Second, the decision whether and – if positively decided – how such a machine is manufactured has to be planned more or less from scratch each time. Thus, the acceptance of an order is a multi-step decision process, being performed by various employees from different departments of the company.

The process steps stem from two complementary realms. A first set is reflecting the financial and budget planning side of production: It has to be found out whether such a machine can be produced profitably. A second set of steps is of technical nature. Depending on the machine to be produced a great number of technical checking and planning steps have to be performed. It is obvious that various tools both from the financial and from the technical area have to be deployed.

Both sorts of process steps, technical and financial, affect each other; also the process steps within the two realms have an effect on each other. In summary, this manufacturing process is a typical representative of a so-called descriptive process (Jablonski 1994) with multiple dependencies between process steps, whereby the sequence of their execution is mostly not predefined but is determined by intermediate step results and personal decision of domain experts. Thus, a declarative process modelling approach seems to be predestinated for this job.

In literature and practice, a couple of approaches for declarative process modeling have been proposed. Declare (Pesic et al. 2007) and MP-Declare (Schönig et al. 2016) have been researched for years by academia. CMMN has been defined by the Object Management Group as standard declarative process modeling languages. Several vendors have been offering practical tools for modeling declarative processes according to this standard, e. g., Camunda.[5] DMN[6], also a standard from the Object Management Group is naturally complementing CMMN when decision tasks have to be specified.

Due to our acquaintance we firstly tried to model our industrial scenario with (MP-) Declare. However, since from the beginning of this endeavour the involved domain people had enormous problems to grasp the semantics of these languages – especially due to the lack of a graphical modeling language – we have not used them further in the project. Also, the usage of DCR graphs (Hildebrandt et al. 2013) could not enhance the acceptance of this modeling approach. Especially, the many different relationships between process steps bearing different semantics has caused problems in understanding.

Finally, we suggested to apply a mixture of CMMN and DMN for modeling the complex application scenario. Our experience with the usage of these technologies has been ambivalent. We divide the following discussion into two parts: A first part tackles the functional and behavioral perspective of a process model while the second part deals with the data, the organizational, and the operational fragments of a process description.

4 KEBAP. Key figure-based Management of Agile Processes. This project was funded by Bayerisches Staatsministerium für Wissenschaft und Medien, Energie und Technologie in the context of the programme *Informations- und Kommunikationstechnik.*

5 `https://camunda.com`

6 `https://www.omg.org/dmn/`

The practitioners get along quite well with the representation of tasks within a CMMN case plan. Also, they comprehend how tasks within a case are interconnected and how decisions are modelled through the DMN language. Nevertheless, they feel kind of overwhelmed with the many different symbols and notions (e. g., stage, sentries, milestones, tags for tasks).

In contrast to the quite positive experience with modeling the functional and behavioral perspective, they have enormous problems with the organizational, operational, and data perspective. Very similar to the experience with BPMN process models (Section 23.3.1), the professional practice has a strong demand for powerful process modeling elements for these perspective. For example, it is of most importance to describe which organizational units and agents are responsible for each task. Beyond that, especially for the technical part of the overall process they need to be able to specify tools, services, and systems required for performing the various tasks.

Since the principle approach to declarative process modeling seems to be quite acceptable, it is a pity that it has not become marketable. We tie this to the congested notation of these languages and the lack of modeling capabilities for the organizational, operational, and data perspective. Camunda[7] conforms this observation where in this blogpost mostly the unusual way of modeling is identified as a reason for the rejection of CMMN. Very similar to the experience from Camunda also in our project we (had to) migrate to *normal* BPMN modeling for the scenarios from above, although those process models have had enormous problems to present the real world decision processes. Besides, according to the observation in the former use case (Section 23.3.1) the problems of missing modeling capabilities for the organizational,the operational, and the data perspective are still not solved.

23.3.3 Field Service and Location Aware Processes

Field service processes are an essential component of many industries, including telecommunications, manufacturing, handcraft, utilities, and healthcare. They comprise various activities such as installation, maintenance, repair, and inspection, which are carried out by field technicians at customer sites. Despite their critical role in ensuring high-quality service delivery and customer satisfaction, field service processes have traditionally been understudied in comparison to other aspects of business operations.

The increasing complexity of field service processes, driven by rapid technological advancements and growing customer expectations, has created an urgent need for a more systematic and comprehensive approach to model and support these processes. Field service operations are resource-intensive involving labor, transportation, and equipment expenses. The emergence of new technologies such as the Internet of Things (IoT) and augmented reality (AR) has transformed the field service landscape. Integrating these technologies into field service processes requires a deeper understanding of the operational perspectives. An aging workforce and a shortage of skilled technicians pose challenges to the field service industry as well. A comprehensive business process model of field service activities helps organizations better manage their workforce by identifying areas for training, improving resource allocation, and enhancing overall productivity. Given these factors, there is a pressing need for a systematic approach to modeling the operational perspective in business processes within field service operations.

7 https://camunda.com/blog/2020/08/how-cmmn-never-lived-up-to-its-potential/

An important use case in this area stems from our practice-oriented research project TRADEmark[8] with three partners from the crafts sector in addition to the academic project partners. The aim of this project is systematic digital support of field service processes in craft enterprises on the basis of process-oriented information systems. A systematic recording and modelling of the companies' operative processes is the basis for the implementation of the process execution software for the field service. For this project, we chose a BPMN-based modelling approach in order to create comprehensible and technically feasible process models. In discussions and workshops with the project partners, a strong focus on two perspectives of process modelling emerged, in particular, which have so far been considered rather little or not at all in existing approaches: (i) the *operational* perspective, i. e., (manual) tools to be used for a specific task, and (ii) the *locational* perspective, i. e., considering the location of entities by the control flow within a process and by task assignment. When examining the requirements towards the operational and locational perspective of processes of handcraft companies we reveal certain critical issues:

- Field service processes have a strong focus on tools to be used for certain tasks, e. g., drills, special tongs. For companies involved, the challenge is to link tasks to specific tools to be able to offer recommendations to handcraft workers who are selected to carry out the process steps. There is no possibility in BPMN to incorporate these requirements into a process model. In BPMN the operational perspective only distinguishes between the different types of tasks, e. g., Service Task, Human Task. Manual aids or concrete factual tools cannot be declared. The technical possibilities of the IoT offer the chance to digitally identify such tools, for instance by means of RFID sensors. A bridge between IoT middleware and process management system could therefore exchange data between the systems and thus automatically distinguish between inappropriate and fitting tools.
- Mobile and locationally distributed processes typically include several process participants working on subsequent tasks at different locations, e. g., a craft business employing multiple craft workers working on different construction sites. Let's consider a mobile distributed business process from handcraft businesses, where craft workers drive to customers to replace broken heaters. Here, location-based data should be used, for example, to allocate the repair order instance and the corresponding tasks to the person closest to the customer's location. However, in the area of BPM most research has touched only aspects of location-awareness such as process adaptation, process modeling and mining (Dörndorfer and Seel 2018; Behkamal et al. 2022) or managerial aspects of context-aware processes (Hallerbach et al. 2008; Rosemann et al. 2008). The description and implementation of an executable system based on standard notations such as BPMN to capture and process location data is either neglected or vaguely contained in one of the others. In particular, to the best of the authors' knowledge, the allocation of concrete users to tasks in BPMN based on location data is also still an open research issue (Poss and Schönig 2023). Again, real-time location data from IoT devices consumed through process management technology helps businesses to implement more efficient, effective, and more sustainable processes.

8 The project IoT-based data and process management in the craft sector (TRADEmark) has been financed with funding provided by the Bavarian Research Foundation (BFS). The authors are responsible for the content of this publication.

What conclusions can we draw from these experiences? It turns out that important aspects of spatially distributed processes in field service have so far been insufficiently considered in process management notations and systems. There is a lack of practical notational elements for location-based contexts such as location-based task assignment. The research community has sufficiently recognised the broad spectrum of possibilities of the IoT for BPM. Nevertheless, both modelling elements and system support for linking non-IT tools and process management are missing. From our point of view, it would be an enormous improvement for practice-oriented process modelling to integrate an operational perspective into existing modelling languages.

23.3.4 Production Processes and Industrial IoT Security Management

Production or manufacturing processes lie at the core of various industries, encompassing a wide range of activities such as material handling, assembly, machining, quality control, and packaging. These processes are essential for creating goods and services that meet customer requirements while maintaining operational efficiency and cost-effectiveness. In recent years, the manufacturing landscape has experienced significant changes due to globalization, technological advancements, and increasing customer demands, making efficient management of these processes more crucial than ever.

The use of business process management in manufacturing processes can provide numerous benefits to organizations, enhancing their competitiveness, overall performance and security awareness. BPM allows organizations to analyze and visualize their manufacturing processes, helping them identify inefficiencies, bottlenecks, areas for improvement, and security issues. By addressing these issues, companies can optimize their processes, leading to increased productivity, reduced cycle times, and lower production costs. In addition, manufacturing industries often operate under strict regulatory requirements and adhere to industry standards such as ISO 9001 or Six Sigma. BPM can help organizations ensure compliance by documenting processes, defining roles and responsibilities, and establishing standardized procedures. BPM can facilitate the integration of new technologies and methods into existing production systems, enabling companies to adapt more quickly and maintain their competitive edge (Erasmus et al. 2020).

Let’s take a closer look at the security aspect in manufacturing companies in particular. The following use case from the INSIST[9] project with two partners from the manufacturing industry in addition to the academic project partners comes from this area. The aim of the project is, among other things, the process-oriented recording of entities involved in the process, i. e., machines, components and people, as well as their security requirements. While the Industrial IoT (IIoT), i. e., the integration of IoT technology within production environments, offers new opportunities, it also has its downsides. The connectivity and IT integration of industrial components creates new opportunities for attackers to infiltrate, disrupt or maliciously alter processes. Due to the novel possibilities and the associated dangers, cyber security in IIoT environments requires special attention. As this problem has already been recognised, there are moves by the EU to establish the implementation of security measures, such as IEC62443, as a standard across the EU. In order to be able

9 The project Industrial IoT Security Operation Center (INSIST) has been financed with funding provided by the Bavarian Ministry of Economic Affairs, Regional Development and Energy. The authors are responsible for the content of this publication.

to consider security in industrial processes from the outset, a comprehensive modelling approach that also covers the security requirements in IIoT processes is necessary.

In this respect, it is crucial for meaningful and sustainable security management to know and define the company resources, their operational processes and their information needs. Based on this, risks can be identified, protective measures taken and security incidents monitored. Against this background, the discipline of BPM offers numerous established methods, concepts and technologies for systematically modelling operational IIoT processes, which can also be used to improve security. While there is already research on the integration of IoT and BPM technology in general, the BPM discipline represents a previously untapped source for improving cyber security in manufacturing companies. Based on these considerations and the requirements, the following problems arise in the modelling of security-relevant aspects in the production plants involved:

- A formally defined process modelling notation such as BPMN can provide a foundation for implementing a BPM-based security-by-design approach. However, since IIoT security requirements from the IEC-62443 standard are not yet supported, a method is missing and must be created to represent security requirements and possible protective measures accordingly. While some notations already exist for security aspects in the classical IT area, a specialised approach for the IIoT with its unique requirements of industrial operating technology is still missing (Hornsteiner et al. 2022).
- The importance of IIoT is already recognised, and by that, a lack of a modelling language to represent IIoT aware processes (Meyer et al. 2015). For that the extensive EU-funded research project *Internet of Things - Architecture (IoT-A)*[10] developed a comprehensive BPMN extension that covers sensors, IIoT-specific tasks, and cloud devices (Meyer et al. 2015; Meyer et al. 2013). For our project we want to exploit these research results to model the manufacturing processes of the practical partners involved. However, the developed research results and artefacts are neither published on the projects' websites nor on public websites of the EU. Also, we do not see any effort to adopt the results of that most comprehensive project in any standard or tool.

We can draw very similar conclusions from the experiences gained in that project as discussed formerly. Due to the increasing merging of IT and operational technology there is an urgent need for process management in IIoT. Here, we focus on the security perspective which is one of the most important aspects of software management these days. However, too little effort can be observed developing this aspect such that is comfortably usable in process management systems. Therefore, our latest research offers a first approach to using BPMN as a security management tool in the IIoT (Hornsteiner and Schönig 2023).

23.4 Conclusion

What is our conclusion from the experiences we have gained both from following and participating in academic research and from continuously conducting practical projects. It is a platitude to postulate that academic research must find its way into professional products and that this is not the job of academia but mostly of companies. Nevertheless, we regard several developments that we have observed as sub-optimal.

10 `https://www.iot-a.eu`

- From an industrial perspective, we miss that industry requires more elaborated research in disciplines of process management, e. g., the organizational and the operational perspective, that are most relevant for practical projects but lack powerful and implementable academic prototypes. Besides, we regret that already developed academic concepts are not adopted neither in standards nor in products.
- From the academic perspective, it should be recognized that certain fields in process management are still not matured enough and especially are not implemented satisfactorily in prototypes and products. That should stimulate further research in that area instead of neglecting such fields and unconditionally following recent research trends.
- Analyzing the practical projects presented in this paper and especially focusing on their selective need for specific perspectives of process modeling – and thus also of execution – we see that modularizing a process model according process perspectives is a desirable feature. Especially, individual extensions of those perspectives for these perspectives covering the requirements of specific projects would be very favourable.

Nevertheless, we regard process management as a powerful technique to systematically support many practical application domains. It has found its way from research into practice. Since we regard computer science, i. e., informatics, also as an engineering discipline, we see a major opportunity in bringing developments forward when academia and praxis are stronger interconnected. Especially, since engineering naturally aims at developing implementable concepts also scientists should place more effort into this issue for developing matured prototypes that afterwards can easily be adopted by (software) companies.

References

Behkamal, B., Pourmasoumi, A., Rastaghi, M. A., Kahani, M., Motahari-Nezhad, H. R., Allahbakhsh, M. and Najafi, I. (2022). ‘Geo-Enabled Business Process Modeling’. In: DOI: 10.48550/ARXIV.2204.08063.

Bialy, M., Pantelic, V., Jaskolka, J., Schaap, A., Patcas, L., Lawford, M. and Wassyng, A. (2017). ‘Software engineering for model-based development by domain experts’. In: *Handbook of System Safety and Security.* Elsevier, pp. 39–64.

Cabanillas, C., Knuplesch, D., Resinas, M., Reichert, M., Mendling, J. and Ruiz-Cortés, A. (2015). ‘RALph: a graphical notation for resource assignments in business processes’. In: *CAiSE.* Springer, pp. 53–68.

Curtis, B., Kellner, M. I. and Over, J. (1992). ‘Process Modeling’. In: *Commun. ACM* 35.9, pp. 75–90. ISSN: 0001-0782.

Dörndorfer, J. and Seel, C. (2018). ‘A Framework to Model and Implement Mobile Context-Aware Business Applications’. In: *Modellierung 2018.* Ed. by I. Schaefer, D. Karagiannis, A. Vogelsang, D. Méndez and C. Seidl. Bonn: Gesellschaft für Informatik e.V., pp. 23–38.

Dumas, M., La Rosa, M., Mendling, J. and A Reijers, H. (2013). *Fundamentals of Business Process Management.* Springer.

Erasmus, J., Vanderfeesten, I., Traganos, K. and Grefen, P. (2020). ‘Using business process models for the specification of manufacturing operations’. In: *Computers in Industry* 123, p. 103297.

Fahland, D., Lübke, D., Mendling, J., Reijers, H., Weber, B., Weidlich, M. and Zugal, S. (2009). 'Declarative versus imperative process modeling languages: The issue of understandability'. In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, pp. 353–366.

Hallerbach, A., Bauer, T. and Reichert, M. (June 2008). 'Context-based configuration of process variants'. In: *3rd International Workshop on Technologies for Context-Aware Business Process Management (TCoB 2008)*, pp. 31–40.

Hildebrandt, T., Marquard, M., Mukkamala, R. R. and Slaats, T. (2013). 'Dynamic condition response graphs for trustworthy adaptive case management'. In: *On the Move to Meaningful Internet Systems*. Springer, pp. 166–171.

Hornsteiner, M. and Schönig, S. (2023). 'Designing Business Processes for Comprehensive Industrial IoT Security Management'. In: *18th International Conference on Design Science Research in Information Systems and Technology (DESRIST)*.

Hornsteiner, M., Stoiber, C. and Schönig, S. (2022). 'Towards Security- and IIoT-Aware BPMN: A Systematic Literature Review'. In: *Proceedings of the 19th International Conference on Smart Business Technologies, ICSBT*, pp. 45–56.

Jablonski, S. (1994). *MOBILE: A modular workflow model and architecture.* Fourth International Working Conference on Dynamic Modelling and Information Systems.

Jablonski, S. and Bussler, C. (1996). *Workflow Management: Modeling Concepts, Architecture and Implementation.* International Thomson Computer Press.

Meyer, S., Ruppen, A. and Hilty, L. (2015). 'The things of the internet of things in BPMN'. In: *Lecture Notes in Business Information Processing*. Vol. 215. Springer.

Meyer, S., Ruppen, A. and Magerkurth, C. (2013). 'Internet of things-aware process modeling: Integrating IoT devices as business process resources'. In: *Lecture Notes in Computer Science*. Springer.

Pesic, M., Schonenberg, H. and Van der Aalst, W. M. (2007). 'Declare: Full support for loosely-structured processes'. In: *EDOC*. IEEE, pp. 287–287.

Poss, L. and Schönig, S. (2023). 'A Generic Approach towards Location-aware Business Process Execution'. In: *Enterprise, Business-Process and Information Systems Modeling (BPMDS)*. Lecture Notes in Business Information Processing. Springer.

Rosemann, M., Recker, J. and Flender, C. (2008). 'Contextualisation of business processes'. In: *International Journal of Business Process Integration and Management* 3.1, p. 47. DOI: 10.1504/ijbpim.2008.019347.

Russell, N., Van Der Aalst, W. M., Ter Hofstede, A. H. and Edmond, D. (2005). 'Workflow resource patterns: Identification, representation and tool support.' In: *CAiSE*. Vol. 5. Springer, pp. 216–232.

Sadiq, S., Sadiq, W. and Orlowska, M. (2001). 'Pockets of flexibility in workflow specification'. In: *Conceptual Modeling—ER 2001: 20th International Conference on Conceptual Modeling*. Springer, pp. 513–526.

Schönig, S., Di Ciccio, C., Maggi, F. M. and Mendling, J. (2016). 'Discovery of multi-perspective declarative process models'. In: *Service-Oriented Computing (ICSOC)*. Springer, pp. 87–103.

Ter Hofstede, A. H., Van der Aalst, W. M., Adams, M. and Russell, N. (2009). *Modern Business Process Automation: YAWL and its support environment.* Springer Science & Business Media.

List of Contributors

Alan Kay	Viewpoints Research Institute (VPRI), USA Palo Alto Research Center, USA and many others
Alexander C. Bock	University of Duisburg-Essen, Germany alexander.bock@uni-due.de
Andreas Hermann	University of Münster, Germany andreas.hermann@ercis.uni-muenster.de
Arne Lange	University of Mannheim, Germany lange@uni-mannheim.de
Bernhard Rumpe	RWTH Aachen University, Germany rumpe@se-rwth.de
Bernhard Thalheim	University of Kiel, Germany thalheim@is.informatik.uni-kiel.de
Christian Tunjic	University of Mannheim, Germany tunjic@uni-mannheim.de
Colin Atkinson	University of Mannheim, Germany colin.atkinson@uni-mannheim.de
Dimitris Karagiannis	University of Vienna, Austria dimitris.karagiannis@univie.ac.at
Elmar J. Sinz	University of Bamberg, Germany elmar.sinz@uni-bamberg.de
Ewelina Księżniak	Poznań University of Business and Economics, Poland ewelina.ksiezniak@ue.poznan.pl
Fabian Muff	University of Fribourg, Switzerland fabian.muff@unifr.ch
Florian Matthes	Technical University of Munich, Germany matthes@tum.de
Friedrich Steimann	FernUniversität in Hagen, Germany steimann@fernuni-hagen.de
Giancarlo Guizzardi	University of Twente, Netherlands g.guizzardi@utwente.nl
Hans-Georg Fill	University of Fribourg, Switzerland hans-georg.fill@unifr.ch
Heinrich C. Mayr	University of Klagenfurt, Austria Heinrich.Mayr@aau.at
Hend*erik* A. Proper	Technische Universität Wien, Austria henderik.proper@tuwien.ac.at

Jens Gulden	Utrecht University, Netherlands
	j.gulden@uu.nl
Jörg Becker	University of Münster, Germany
	joerg.becker@ercis.uni-muenster.de
José Ignacio Panach	Universitat de València, Spain
	joigpana@uv.es
Judith Michael	RWTH Aachen University, Germany
	michael@se-rwth.de
Juraj Vladika	Technical University of Munich, Germany
	juraj.vladika@tum.de
Kurt Sandkuhl	University of Rostock, Germany
	kurt.sandkuhl@uni-rostock.de
Malte L. Peters	Federal University of Applied Administrative Sciences, Germany
	post@malte-peters.de
Marc Frerichs	Universität Hamburg, Germany
	marc.frerichs@uni-hamburg.de
Marco Konersmann	RWTH Aachen University, Germany
	konersmann@se-rwth.de
Markus Nüttgens	Universität Hamburg, Germany
	markus.nuettgens@uni-hamburg.de
Naciye Akca	Federal University of Applied Administrative Sciences, Germany
	Naciye.Akca@hsbund.de
Oscar Pastor	Universitat Politècnica de València, Spain
	opastor@dsic.upv.es
Paul Kruse	University of Münster, Germany
	paul.kruse@ercis.uni-muenster.de
Peter Fettke	Saarland University, Germany
	peter.fettke@iwi.uni-sb.de
Phillip Schneider	Technical University of Munich, Germany
	phillip.schneider@tum.de
Reinhard Schütte	University of Duisburg-Essen, Germany
	Reinhard.Schuette@icb.uni-due.de
Rene Noel	Universidad de Valparaíso, Chile
	rnoel@pros.upv.es
Robert Winter	University of St.Gallen
	robert.winter@unisg.ch
Rosa Velasquez	Universitat Politècnica de València, Spain
	rvelasquez@vrain.upv.es
Stefan Jablonski	University of Bayreuth, Germany
	stefan.jablonski@uni-bayreuth.de
Stefan Klein	University of Münster, Germany
	stefan.klein@uni-muenster.de
Stefan Schönig	University of Regensburg, Germany
	stefan.schoenig@ur.de
Stephan Zelewski	University of Duisburg-Essen, Germany
	stephan.zelewski@pim.uni-due.de

Steven Alter	University of San Francisco, USA alter@usfca.edu
Tim Schopf	Technical University of Munich, Germany tim.schopf@tum.de
Tobias Kautz	University of St.Gallen tobias.kautz@unisg.ch
Tony Clark	Aston University, United Kingdom tony.clark@aston.ac.uk
Witold Abramowicz	Poznań University of Business and Economics, Poland witold.abramowicz@ue.poznan.pl
Wolfgang Reisig	Humboldt-Universität Berlin, Germany reisig@informatik.hu-berlin.de

Colophon

True to Ulrich's preference for open science, open access to scientific knowledge, and free/open source software, his Festschrift is typeset with LaTeX (LuaLaTeX 1.17.0, TeX Live 2023) using the Libertinus and `Inconsolata` typefaces (both of which come with open font licences). Libertinus is a typeface derived from the Linux Libertine typeface (`https://en.wikipedia.org/wiki/Linux_Libertine`) initially by Khaled Hosny (`https://github.com/khaledhosny`) and later by Caleb Maclennan (`https://github.com/alerque`) and The Libertinus Project Authors (`https://github.com/alerque/libertinus`). Inconsolata is a typeface drawn by Raph Levien (`https://levien.com`) partly sponsored by the TeX User Group (TUG) and further developed by Kirill Tkachev and the Cyreal foundry (`https://levien.com/type/myfonts/inconsolata.html`). Both typefaces are used in the Open Font Format (based on ISO/IEC14496-22) in conjunction with the Memoir document class by Peter Wilson and Lars Madsen (version 3.8.1 of 21 August 2023). As Ulrich prefers British English, punctuation and section title capitalisation has been typeset following style guidelines common to British English. However, spelling has not been changed from the authors' spelling. Initial chapter submissions in the Lectures Notes in Informatics format were converted by Stefan Strecker and Jürgen Jung. Jürgen took on the laborious task of converting the submissions in non-TeX formats to LaTeX. The Festschrift was typeset by Stefan Strecker.

Informing Possible Future Worlds is the Festschrift in honour of Ulrich Frank on the occasion of his 65th birthday. The Festschrift includes twenty-three essays written by friends, colleagues, and fellow researchers in recognition of Ulrich Frank's contributions to Wirtschaftsinformatik research and the scientific community. Each essay is a personal and unique *birthday present* to Ulrich Frank written exclusively for the Festschrift. From original research contributions to more personal reflections, the essays cover a wide range of topics, themes, and fields.

The Festschrift is edited by Stefan Strecker, FernUniversität in Hagen and Jürgen Jung, Frankfurt University of Applied Sciences.

Logos Verlag Berlin

ISBN 978-3-8325-5768-3