An Assessment Model to Foster the Adoption of Agile Software Product Lines in the Automotive Domain

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades

Doktor der Ingenieurwissenschaften

Dr.-Ing.

genehmigte Dissertation von

M.Eng.

Philipp Nikolaus Hohl

geboren am 28.03.1988

in Ravensburg

2019

Chapter 1

Introduction

This chapter describes the main challenges addressed by this thesis. It defines the research questions, as well as the goals and hypotheses. Furthermore, it presents the research contribution. Finally, the thesis structure is outlined.

1.1 Background and Motivation

Within the last decades, the amount of software and systems that rely on software has increased significantly. This rise is still an ongoing trend. Software surrounds us and is present in all areas of our daily life. According to Thomas [246], about 86% of the world's population own a smartphone. He mentions that there are more than 6 billion cellphone subscriptions. Nowadays, every smartphone can remotely control things. We use it to control our smart homes from all around the world. Furthermore, we are driving cars which are getting smarter and are connected to the environment. All over the world, people are craving for new technological enhancements and electronic devices. This desire can be observed in all industry sectors.

The automotive domain is recently in a disruptive change. E-Mobility, self-driving, and community owned cars are new technologies and upcoming market trends in the automotive domain [185]. Vehicles are no longer isolated but connected to the environment [241]. Upcoming challenges and market trends need to be addressed in the development of new car generations in order to cope with new ways of mobility. A lot of innovation for automobiles' functionalities is nowadays addressed in software.

According to Oliveira [196], 85% of the automobiles' functionalities are realized with software running on Electronic Control Units (ECUs). The amount of software has evolved from zero to tens of millions of lines of code [210], distributed on up to 70 ECUs [39, 184]. Broy [25] identified the increasing amount of software already in 2006. He stated that the amount of software in the car has increased exponentially in the last decades [25]. McCaffery et al. [176] expect that the quantity of software grows continuously. They further emphasize that software must be developed faster and in a more cost effective way, but still in high quality [176].

1.1.1 Quality Aspects and the Need for Software Reuse

The reuse of software parts is necessary to develop the large amount of different software variants, while simultaneously maintaining the quality of the software. Software product lines are a software paradigm for systematic software reuse [157]. This paradigm is particularly important for the automotive domain to meet different requirements across multiple markets [245]. Software product lines help to manage changes, coordinate the worldwide software development, and increase the software quality by the reuse strategy. Furthermore, the software variants enable a customization and individualization to meet customers' requirements and generate more value for the core business [109].

According to Wozniak et al. [267], the automotive domain is the most challenging environment for systems and software product line engineering. They identify a "very large number of individually complex products with incomparably rich feature variation" [267, p.336]. The variability in the software is made explicit through variation points which introduce the possibility to delay the design decision during the development [24]. Automotive software platforms can easily incorporate thousands of variation points and configuration parameters [245, 73]. The final decision, whether a feature is part of a specific product or not, is known as the binding time [259]. One can distinguish between four different binding times: during programming, at the integration, in the assembly, and during runtime [157]. All of them are used in the automotive domain. The consistency and the reliability of systems has to be ensured at all times [157]. Therefore, it is necessary to link and manage the variation points throughout all phases of the full product life cycle, such as requirements elicitation, implementation, test processes, and at the client's premises.

1.1.2 The Demand for Faster Software Development

Traditional working structures make it difficult to deliver the large amount of software fast enough [25], because the development processes are too slow to keep pace with the fast changing market [23]. Future challenges will likely be addressed in software and need flexible development processes to react on changing market demands. Furthermore, it is important to keep the development time short [221], in order to achieve business goals and release the software faster to the market to generate profit. However, currently used development processes have shown to be insufficient for handling such an exponential growth of software [63].

The development processes need to be redesigned in a way that allows to learn and adapt continuously at a fast pace. Agile methods are promising approaches to address these new challenges. Since 2001, agile software development methods promise an improved software development with a faster time to market and an increasing speed of learning. In addition, agile development practice offer the possibility to react on changing requirements and to refine the final software during the development process. The adaption of agile practices within the automotive domain is therefore a possible way to keep pace with fast changing market demands [140].

1.1.3 Development Process and Agile Transformation

The combination of agile software development and software product lines in the automotive domain is assumed to be difficult [140]. Agile methods are not tailored to the specific characteristics of the automotive domain. Although there have been efforts to apply agile methods in the automotive domain, widespread adoptions have not yet taken place.

Agile methods and practices are primarily designed for short development cycles in small development teams. One benefit of agile software development is that the used methods emphasize collaboration and communication between development teams as well as between the customer and developers [10]. Product Owners (POs) define requirements in terms of business value and collaborate with the development teams. For larger teams, agile methods are scaled up to approaches for large organizations, such as the Scaled Agile Framework (SAFe) or Scrum of Scrum (SoS). The scaled frameworks aim to maintain an open communication culture within the development team and between development teams. This is challenging whenever the software development is distributed all over the world [213]. In the automotive domain, software product lines are used to manage the global software development [245]. This leads to communication overhead for worldwide coordination, with several included POs. The POs are responsible for different markets and different model ranges. It is important for a global software development that Product Owners are consistent regarding the requirements to maintain the software product line. However, the worldwide scoping process and the need to find common solutions hinders short iterations [108].

Therefore, it is necessary to combine agile software development with software product line engineering to shorten the time to market and to meet customer and market demands even better. Furthermore, this combination promises an improved long-term productivity, efficiency, and profit [173].

1.2 Research Scope

The research scope of this thesis can be described along the three areas *Automotive Domain, Agile Software Development* and *Software Product Lines* as follows (cf. Figure 1.1):

1.2.1 Automotive Domain

In 2007, Pretschner et al. [209] characterized the automotive domain and presented five salient features of the automotive software domain. They mention the (1) "heterogeneous nature of the software, [(2)] the organization of labor, [(3)] the distributed nature of automotive software, [(4)] the huge number of variants and configurations, [(5)] and the predominance of unit-based cost models" [209, p. 4]. The research presented in this thesis is limited to automotive embedded software development with its special peculiarities, such as rigid quality and safety requirements [245]. Furthermore, the software development must consider deep integration between hardware and software, strong focus on development processes and strong supplier involvement [108].

Chapter 2

Theoretical Foundation

This chapter presents the main concepts and principles of using assessment models, software product lines and agile software development. It provides information on concepts that form the basis for the proposed assessment model. First, the peculiarities of the software development in regulated domains and the use of assessment models is described. Second, the paradigm of software product lines is explained. In the third section, agile development practices and methodologies are presented. Finally, the combination of agile software development and software product lines in practice is discussed.

2.1 Automotive Assessment/Reference Models

2.1.1 The Need for High Quality and Faultless Software

The automotive domain is confronted with an increasing complexity of embedded software. Upcoming market demands and desired customer features are addressed in software. Therefore, it is important to ensure that the software is working as expected and is of high quality. All software development processes aim to ensure a high quality and faultless software. In the automotive domain, ISO 26262 [124] specifies software development tasks to realize and verify requirements introduced by safety-critical functions [221]. For this purpose, the software development process is organized in phases, comprising activities and tasks for a specific part of software development [237]. The phases are aligned to a V-shaped curve, the so-called V-Model, which is prevalent in the automotive domain and prescribed by ISO 26262 [122]. In 2005, the V-Model was replaced by the "V-Modell XT" [66] which is published by the German government and defines a standard software development process for IT-projects [66]. The V-Model combines design phases on the left descending path and testing phases on the ascending side of the V. Testing processes demonstrate that the software units fulfill the software unit design specifications. Furthermore, testing processes verify that the software does not contain undesired functionality and is of high quality [122].

2.1.2 Automotive Software Process Improvement

Improvements can be made in many areas within the development process. For example, one possible improvement is the development process itself. Different standards for software process improvement have evolved over the past. Process maturity models provide a technique to assess existing processes, determine process compliance levels and improve the development process. The possibility of using a process assessment model and the improvement resulting from an assessment always depends on the development context. Relevant standards for different contexts are considered. Process assessments are used to evaluate the processes of an organizational unit against a predefined process assessment model. The most popular standards in the automotive domain are CMMI [243] and ASPICE [260]. Both define methods to evaluate complete process models and organizations [98].

These process models define how software shall be developed and tested. Furthermore, they represent best industry practice. Following standardized processes have the advantage that outcomes are reproducible and comparable. This is important for the automotive domain, especially in collaboration with suppliers in order to make outcomes comparable.

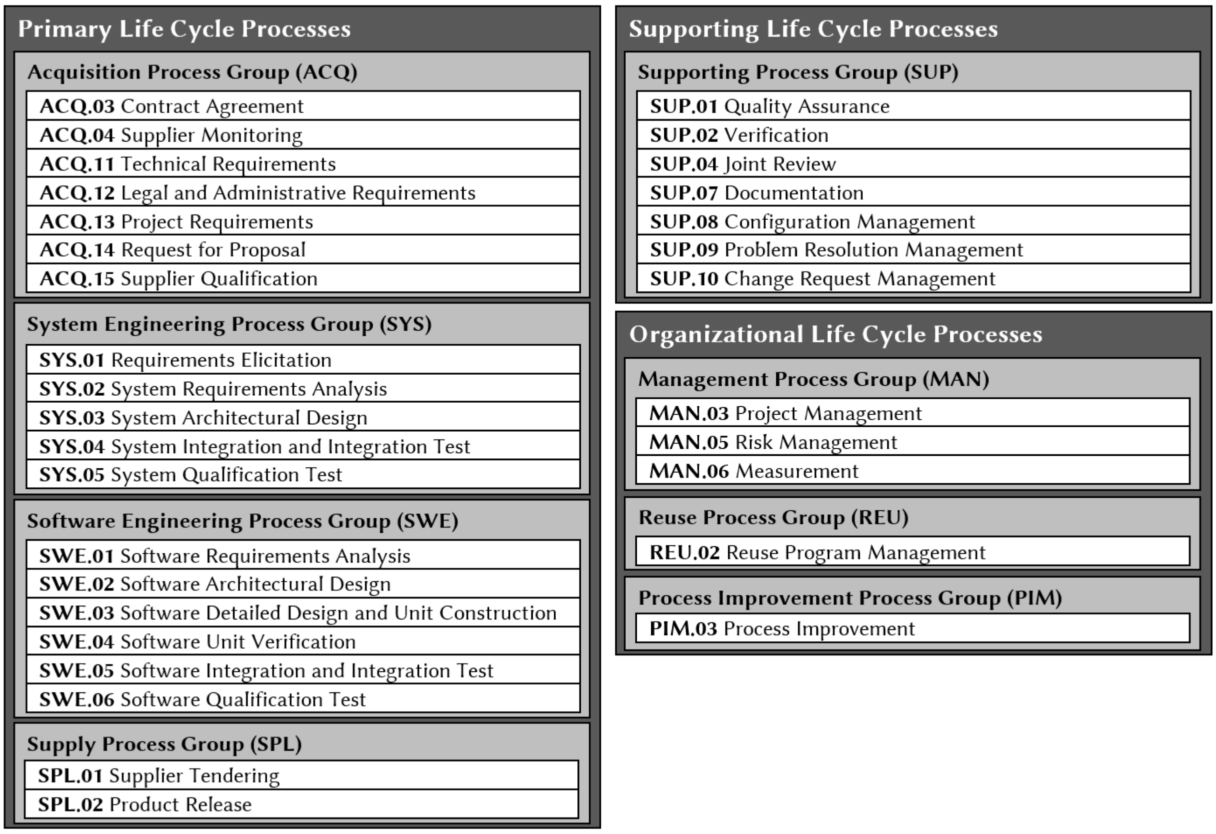
2.1.3 ASPICE Process/Reference Model

The ASPICE Process/Reference Model [260] is a model to evaluate the process capability of the development of embedded automotive systems. Automotive Original Equipment Manufacturers (OEMs) use this assessment model to assess the software capability of their suppliers. The ASPICE Process/Reference Model defines the processes and introduces two process-specific performance indicators: Base Practices and Work Products. The former is defined as an activity or task to achieve the process outcome. The latter focuses on the outcome of the process and is result oriented [260].

ASPICE is maintained by the German Association of the Automotive Industry, called Verband der Automobilindustrie (VDA). The VDA has more than 600 members[1]. Member companies are large OEMs, but also many renowned Small and Medium-sized Enterprises (SMEs). In addition, the membership is nowadays not limited to German companies only. The ASPICE standard itself was not newly developed by the VDA. It has been drafted in 2005 "under the Automotive SPICE initiative by consensus of the car manufacturers within the Automotive Special Interest Group (SIG), a joint special interest group of Automotive OEM, the Procurement Forum and the SPICE User Group" [260, p. 2], and evolved since that time to the latest release 3.1 in 2017.

The ASPICE Process/Reference Model groups the processes into process categories and process groups according to the type of activity they address [260]. Three categories are defined: (1) Primary life cycle processes, (2) Organizational life cycle processes, and (3) Supporting life cycle processes [260]. Figure 2.1 shows the three categories of life cycle processes and the processes which are assigned to them.

[1]https://www.vda.de/en/association/members



Primary Life Cycle Processes

Acquisition Process Group (ACQ)
- **ACQ.03** Contract Agreement
- **ACQ.04** Supplier Monitoring
- **ACQ.11** Technical Requirements
- **ACQ.12** Legal and Administrative Requirements
- **ACQ.13** Project Requirements
- **ACQ.14** Request for Proposal
- **ACQ.15** Supplier Qualification

System Engineering Process Group (SYS)
- **SYS.01** Requirements Elicitation
- **SYS.02** System Requirements Analysis
- **SYS.03** System Architectural Design
- **SYS.04** System Integration and Integration Test
- **SYS.05** System Qualification Test

Software Engineering Process Group (SWE)
- **SWE.01** Software Requirements Analysis
- **SWE.02** Software Architectural Design
- **SWE.03** Software Detailed Design and Unit Construction
- **SWE.04** Software Unit Verification
- **SWE.05** Software Integration and Integration Test
- **SWE.06** Software Qualification Test

Supply Process Group (SPL)
- **SPL.01** Supplier Tendering
- **SPL.02** Product Release

Supporting Life Cycle Processes

Supporting Process Group (SUP)
- **SUP.01** Quality Assurance
- **SUP.02** Verification
- **SUP.04** Joint Review
- **SUP.07** Documentation
- **SUP.08** Configuration Management
- **SUP.09** Problem Resolution Management
- **SUP.10** Change Request Management

Organizational Life Cycle Processes

Management Process Group (MAN)
- **MAN.03** Project Management
- **MAN.05** Risk Management
- **MAN.06** Measurement

Reuse Process Group (REU)
- **REU.02** Reuse Program Management

Process Improvement Process Group (PIM)
- **PIM.03** Process Improvement

Figure 2.1: The Automotive SPICE® (ASPICE) process reference model. Processes are assigned to the three categories of life cycle processes [260].

The (1) primary life cycle processes category comprises processes that are used in the collaboration with customers and suppliers. The category provides processes that help to acquire and support a product or service [260]. Furthermore, the life cycle processes category includes engineering processes for elicitation and management of customer requirements, specifications, development of the corresponding software architecture and design, integration and testing of the software [260].

The (2) organizational life cycle processes category provides processes that support the organization in achieving its business goals. Specifically, processes are provided that help to manage the development process and improve the processes performed in an organizational unit. Furthermore, reuse opportunities are exploited to manage a systematic reuse [260].

The (3) supporting life cycle processes category may be applied by any of the other processes. The use of these processes is not restricted to a specific point in the life cycle. In fact, supporting life cycle processes, such as documentation and quality assurance, need to be considered continuously [260].

The process reference model contains the unique functional objectives of each process and provides a list of specific and expected results of the process [260]. An overview of