Mälardalen University Press Dissertations
No. 95

LEAN THINKING APPLIED TO SYSTEM ARCHITECTING

Håkan Gustavsson

2011

MÄLARDALEN UNIVERSITY
SWEDEN

School of Innovation, Design and Engineering

Copyright © Håkan Gustavsson, 2011
ISBN 978-91-7485-003-1
ISSN 1651-4238
Printed by Mälardalen University, Västerås, Sweden

Abstract

Software-intensive systems are increasingly part of new products, which leads to significant business impact. This is especially true for the automotive industry where a majority of new innovations are realized through the use of software. The architecture of the software-intensive system will enable value creation when working properly or, in the worst case, prevent value creation.

Lean Thinking is about focusing on the increase of customer value and on the people who add value. This thesis investigates how system architecting is performed in industry and how it can be improved through the use of Lean Thinking. The architecting process does not create immediate value to the end customer, but instead creates the architecture on which value, in terms of product features and functionality, can be developed. A Lean tool used to improve the value creation within a process is Value Stream Mapping (VSM). We present a method based on VSM which is adapted to enable analysis of the architecting process in order to identify improvements.

A study of architecting at two companies shows what effect differences such as a strong line organization or a strong project organization have on the architecting process. It also shows the consequences technical choices and business strategy have on the architecting process. In order to improve the understanding of how architecting is performed, a study was carried out, including interviewing architects at six different well-known international companies. The study presents the practices that were found to be most successful. The context of the different companies as well as the architecting practices are compared and analyzed.

The early design decisions made when developing software-intensive systems are crucial to the outcome of development projects. In order to improve the decision-making process a method based on Real Options was developed. The method improves the customer focus of critical design decisions by taking the value of flexibility into account.

This thesis provides a toolbox of knowledge on how Lean Thinking can be applied to system architecting and also presents how architecting is performed in industry today.

Acknowledgements

Going through the process of getting a PhD is a long and winding road, similar to raising a baby. At first, even before the baby is born, the parents think a lot about what becoming a family will be like. This is very similar to before starting work for a PhD. You think you know exactly what it will be like and what you will do, but in the end it is like nothing you could imagine.

As an industrial PhD student I sometimes miss the academic atmosphere found at the university. When visiting Mälardalen University I have always found support and inspiration from the members of the BESS research group. It has been wonderful to make this PhD journey together with Peter Wallin and Stefan Cedergren.

This work has been supported by the Knowledge Foundation, the Swedish Agency for Innovation Systems (VINNOVA) and Scania CV AB, for which I am grateful. I also owe my thanks to many people at Scania who have given me support and interesting discussions. I am grateful to my former manager, Nils-Gunnar Vågstedt, and current manager, Tony Sandberg, for always giving me time to discuss my research and taking care of the project's administration.

My steering committee has done a great job in keeping me on track and on time. My former industrial supervisor and co-author Jan Sterner was great support at the beginning of the journey. During the latter part I was lucky to have a great partnership with my co-author, Ulrik Eklund. My assistant supervisors, Joakim Fröberg and Christer Norström, have provided great support throughout my journey. Discussions with Christer Nordström early in the project were particularly good inspiration and motivation. This journey would not have started if it were not for my supervisor Jakob Axelsson. He was one of the main reasons I started this journey and has guided me to the end, always challenging, supporting and continuously improving my skills. You have been amazing!

I would like to thank my parents for giving me the courage to accept the challenge of pursuing this journey, but recently the biggest fan of my research has been Ebba. When I explain my recent findings to her she sometimes screams with happiness. Her enthusiasm is always there, and even when I discuss research methodology she waves her arms in joy.

Needless to say, Ebba is my newborn daughter who has been the best motivation to finish on time. Lastly, this work would not have been possible without the love and support of my fiancée Cecilia, and I am looking forward to continuing our journey together. You are the light of my life.

Håkan Gustavsson

Huddinge, February 2011.

List of Included Papers

Paper A

Improving the system architecting process through the use of Lean tools. Håkan Gustavsson and Jakob Axelsson, In *Proceedings of Portland International Conference on Management of Engineering and Technology*, ISBN: 978-1-4244-8203-0, Thailand, 2010.

The case study at Scania and Volvo Cars was conducted in co-operation with Ulrik Eklund. The paper was written by the author with support from Jakob Axelsson.

Paper B

Architecting Automotive Product Lines: Industrial Practice. Håkan Gustavsson and Ulrik Eklund, In *Proceedings of the 14th International Software Product Line Conference*, ISBN: 978-3-642-15578-9, Lecture Notes in Computer Science, Vol. 6287, South Korea, 2010, pp. 92-105.

The design of the study was made by the author. The remaining work was done in close co-operation with co-author Ulrik Eklund.

Paper C

A Comparative Case Study of Architecting Practices in the Embedded Software Industry. Håkan Gustavsson and Jakob Axelsson, To be published in *Proceedings of 18th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*, IEEE, Las Vegas, April, 2011.

The case study was conducted by the author and the paper was written with support from Jakob Axelsson.

Paper D

Evaluation of Design Options in Embedded Automotive Product Lines. Håkan Gustavsson and Jakob Axelsson, in *Applied Software Product Line Engineering*, Editors K. C. Kang, V. Sugumaran, and S. Park, ISBN: 9781420068412, Auerbach Publication, 2009, pp. 478-495.

The development of the method was conducted by the author and the book chapter was written with support from Jakob Axelsson.

Additional publications

Theses

- Economical valuation of architectural decisions within automotive electronics. Håkan Gustavsson, Licentiate thesis, Mälardalen University Press, October, 2008.

Journal

- Architecting Automotive Product Lines: Industrial Practice. Håkan Gustavsson and Ulrik Eklund, Invited and submitted to *Journal of Science of Computer Programming*, Elsevier, 2011.

Conference papers

- Architecting Complex Embedded Systems: An Industrial Case Study. Håkan Gustavsson and Jakob Axelsson, To be published in *Proceedings of IEEE International Systems Conference,* Montreal, Canada, April, 2011.
- Implementing Value Stream Mapping: VSM in a R&D organization. Johan Tingström, Håkan Gustavsson and Peter Palmér, In *Proceedings of the 8th Biannual Conference Norddesign 2010*, ISBN 978-91-633-7064-9, Gothenburg, August, 2010
- Analyzing the System Architecting Value Stream. Håkan Gustavsson, Jakob Axelsson and Stefan Cedergren, In *Proceedings of the 7th European Conference on Systems Engineering*, Stockholm, May, 2010.
- A Framework for the Evaluation of Resource Efficiency in Automotive Embedded Systems. Håkan Gustavsson and Erik Persson, In *Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, ASME, ISBN 0-7918-3831-5, New York, August, 2008.
- Evaluating Flexibility in Embedded Automotive Product Lines Using Real Options. Håkan Gustavsson and Jakob Axelsson, In *Proceedings of the 12th International Software Product Line Conference*, IEEE, ISBN 978-0-7695-3303-2, Limerick, September, 2008.

- An Industrial Case Study of Design Methodology and Decision Making for Automotive Electronics. Håkan Gustavsson and Jan Sterner, In *Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, ASME, ISBN 0-7918-3831-5, New York, August, 2008.
- Using Real Options In Embedded Automotive System Design. Håkan Gustavsson and Jakob Axelsson, In *Proceedings of the Conference on Systems Engineering Research*, INCOSE, Redondo Beach, April, 2008.

Workshops

- Coping with Variability in Automotive Product line Architectures Using Real Options. Håkan Gustavsson and Jakob Axelsson, In *Proceedings of the 11th International Conference of Software Product Line Conference, workshop on Managing Variability for Software Product Lines*, Kyoto, Japan, September, 2007.

Preface

I have taken this opportunity to present myself and the background to this work. I have worked with vehicle electronic systems integration and architecture at Scania in Södertälje since 2002. At Scania, each truck and bus produced is customer ordered and unique, but is based on the same architecture. In order to reduce complexity, the interfaces need to be simple and flexible and this often requires a trade-off. When I started working as an architect, I thought decisions were made solely on the basis of technical and financial aspects. It soon became clear that the technical issues are often the easy part of the job. The organizational issues, such as where competence is allocated or how responsibilities are shared, are often much more complex.

As my only experience of architecting comes from working at Scania, I thought I needed to learn more in order to improve our way of working. This idea led to the start of my research journey. Working at Scania, it is hard not to be affected by the company's core values, which are very influenced by Lean. Scania has applied Lean to its production for 20 years, and for more than 10 years in research and development. This background resulted in the question of how we can improve our work further by applying Lean Thinking to system architecting.

I hope this work will provide academia with knowledge of how architecting is performed in industry and how Lean can be applied to architecting. I believe that the methods found can be used in industry for comparison and inspiration regarding process improvements.

Table of Contents

Chapter 1. Introduction

Product development involving software-intensive systems is becoming more and more complex, both organizationally and technically. Most large companies are offering their products to a global market and development is often conducted in different countries. Global presence leads to an increased number of variants and more competitive market. A global product development organization is challenged by geographical distance, cultural differences and, more practically, different time zones. To stay competitive models are launched more frequently, leading to a demand for shorter development cycles and a shorter time-to-market. The development cycle can be shortened by improving the process – making more with fewer resources. The development cycle can also be shortened by increased reuse of technology and components– making more variants with fewer parts. Or as stated by a Japanese development manager:

"No-change development is the best development".

Software-intensive systems are often cross functional, which leads to more and closer cooperation with suppliers and between different organizational units. The increased complexity of the products through a larger number of variants and models places high demands on the interfaces between the different parts of the system. The architecture of those systems is therefore important in order to handle the complexity and to cope with market demand. In 1951, the architect of the DC3 aircraft series highlighted some of the essentials of successful aircraft development. One of the fundamental elements is argued to be the adaptiveness of the development process:

"The ability to cope with the unexpected. Since no amount of planning or technique can bring all factors under control, there must be an ability to capitalize on good luck or minimize the effects of hard luck."

Today, it is even more important to reduce risk in the early phases of development in order to prevent projects running over time and budget. This thesis aims at improving how system architecting is performed by the analysis of industrial practice and through the development of new methods.

1.1 Background

Architectural changes to distributed embedded systems are either evolutionary or revolutionary [40]. The main purpose of this research project is to understand the evolutionary architecting process and its contribution to lean values. The lean philosophy is basically common sense that is packaged so it can be applied to different domains. The project will thereby provide an understanding of the process and the value of the deliverables coming out of the architecting process.

1.1.1 System architecting

The context of architecting includes many disciplines, ranging from human science to computer science [64]. Even the architecture of software-intensive systems is a very large concept. The level of architecture studied in this work is focused on a complete system level. The architects at this level are responsible for the overall system, rather than just one sub-system. The main industrial users of the results from this project are systems architects and their managers, but the results will also be highly relevant to developers of software-intensive systems. The academic reader is probably conducting research in the field of software engineering or technology management.

In order to approach the field of system architecture scientifically, it is necessary to define the terminology as precisely as possible, and we will therefore now introduce definitions of some key concepts.

System is defined by Rechtin and Maier [61] as a set of different elements so connected as to perform a unique function not performable by the elements alone.

Figure 1 Examples of software-intensive systems.

Software-intensive system [1]: Any system where software contributes essential influences on the design, construction, deployment, and evolution of the system as a whole. Examples of such software-intensive systems are industrial robots and vehicles (Figure 1).

Architecture [1]: The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.

Architecting is the process (Figure 2) of shaping the architecture to meet customer demand by balancing requirements, guiding principles and product vision.

One architect interviewed in this project replied as follows to the question: How do you know if the architecting process is working well?

"When new functionality can be absorbed by the architecture without the need for large changes."

A similar view is given by Coplien and Bjørnvig [23] when explaining how architecture adds value to the end-user. Good architecture shortens the time between understanding user needs and delivering a solution by eliminating rework [23].

According to Maier and Rechtin [61] one of the most widely applicable heuristics in systems architecting is: "Simplify. Simplify. Simplify." Another is "Build in and maintain options as long as possible in the design and implementation of complex systems. You will need them."

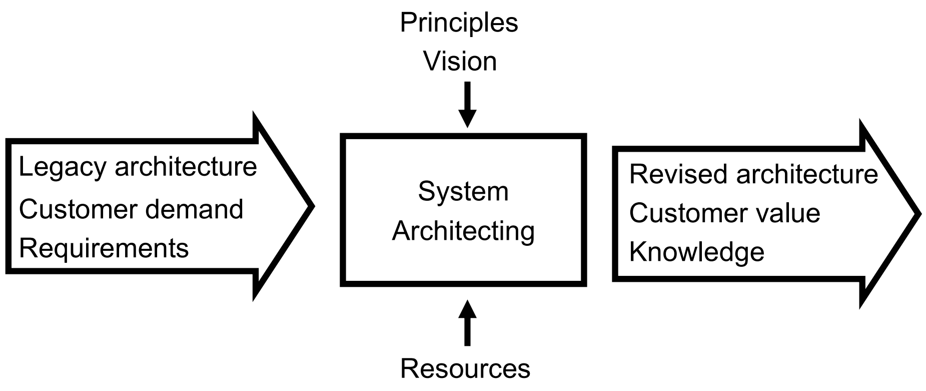

Figure 2 Attributes that affect the architecting process.

Similar heuristics was found in a case study at Scania [39]. A development team illustrated how alternatives are evaluated on the basis of three architectural principles:

- Simple is best.
- Smallest number of variants.
- Minimal interface between modules.

1.1.2 Automotive embedded systems

Today, most innovations made within the automotive domain are driven by electronics. In 2006, Volvo Cars [28] estimated the value of electronics in a high-end car to be 30%. The increased use of electronics also means that the organization involved in the development of embedded systems is growing. Figure 3 shows the increase in employees within embedded systems at Scania as well as the stagnating effect of the automotive crisis in 2008-2009. According to a study by Hoch et al. [44] the total value of electronics in automobiles was expected to rise from 25% in 2006 to 40% in 2010.

Automotive customers demand new functionality with every new product release and the time-to-market is constantly shortened. One example of new functions is Advanced Driver Assistance Systems that help the customer to drive the vehicle safety. Those systems typically use information about the surroundings to increase road safety.

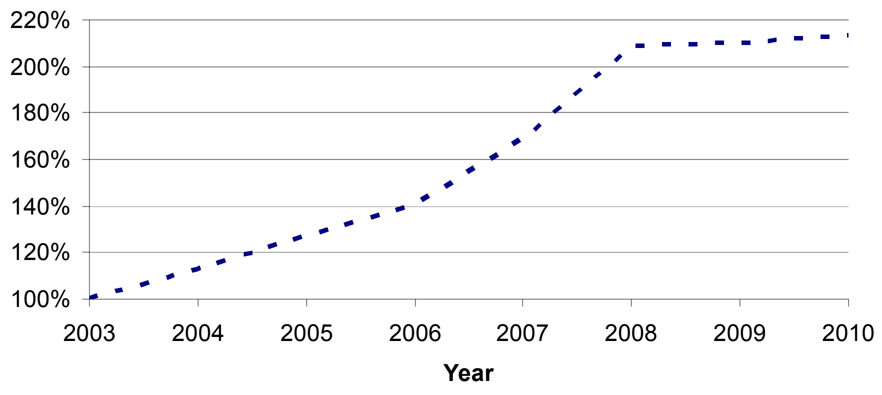

Figure 3 The accumulated increase in employees within embedded systems at Scania relative to 2003.

This is done by using sensors to identify nearby objects or communication with other vehicles or infrastructure to obtain more information. The increased interaction between various components and the wider boundaries of the system increases its complexity and demands flexibility for easy integration.

The building blocks of an automotive electrical and electronic (E/E) system consist of electronic control units (ECUs) executing the software modules that implement the functionality. ECUs are connected to communication networks. As shown in Figure 4, the communication networks are usually divided into sub networks and the communication between those is done through gateway ECUs connected to a backbone. Different sensors and actuators are connected to the ECUs, depending on how functions are allocated to the ECU.

When designing an automotive E/E system, there are many different attributes to consider, such as functional requirements, energy management and the wiring harness [37].

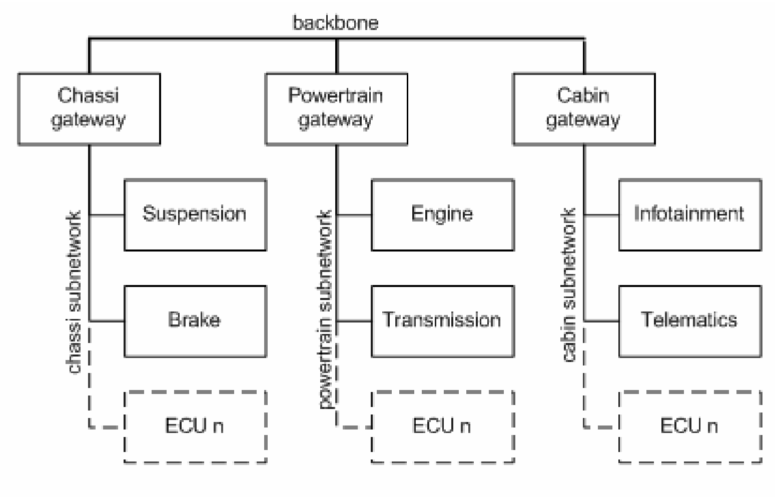

Figure 4 A typical vehicle communication network.

1.1.3 Lean

The concept of Lean derives from the production methods developed by Toyota in the 1950s. Since then, Lean philosophy has been applied to diverse areas of operation. The ideas originating from Toyota are also used in the Six Sigma quality system [41]. Lean development is a way of thinking and a system of management used to create customer value [91]. The value creation starts at the suppliers and goes through factories into product features and out to customers. The concept of Lean production has today moved from manufacturing into various sectors, such as maintenance, purchasing, logistics, and on to product development, which is the topic of this thesis. In a comparison of the product development process made by Morgan [62] it was found that Toyota outperformed its US competitors in both quality and time-to-market.

Software development has also been inspired by Lean, resulting in Scrum, Agile [82] and Lean Software Development [70]. In this thesis, Agile is used to refer to the software development practices of the Agile Manifesto[1].

Mapping the practices to the different functions (Figure 5) of a company that produces software-intensive systems shows how Lean is a much larger concept than Agile, for example. Agile is applied only to the development of software and Scrum is used mainly within R&D. Six Sigma can be used to improve processes in most parts of the company, but does not discuss cultural issues [41]. The Lean philosophy is not just a set of tools; instead it affects all parts of the company, from human resources to marketing. This is also the reason why Lean applies so well to architecting, because architecting is a cross-functional activity.

It is important to note that none of the seminal work in Lean Product Development was carried out by people working inside Toyota or by native Japanese people. This means that the available knowledge should be considered more as a western interpretation of the Toyota Way, though the success of the Lean philosophy is undeniable.

Lean development focuses on creating re-useable knowledge - knowledge that contributes to the profitability of future operational value cycles and that can ideally be used for many projects [91].

[1] http://www.agilemanifesto.org/principles.html

Baines et al. [7] present the result of a systematic literature review of what is meant by the term Lean in product development. One finding is that the definition of Lean is drifting and moving from waste reduction towards value creation. Another result is that value is added in product development when useful information is produced, but value needs to be defined precisely.

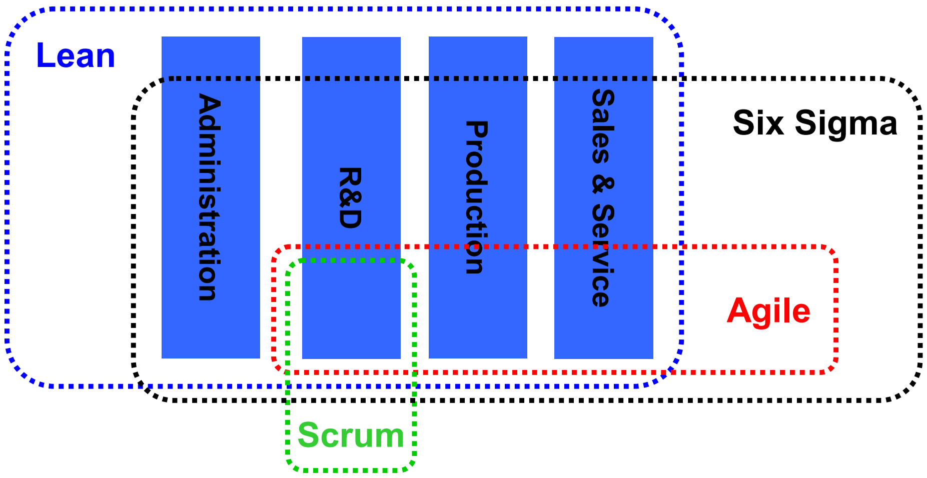

Figure 5 Implementation of development practices in the organizational parts of a company.

1.1.4 Personal experience - Lean in Japan

During a study trip to Japan I visited a number of companies and personally experienced the culture that gave birth to the Lean philosophy.

Within Japanese culture there is great dedication to following rules and avoiding errors. Lean literature often mentions how work should be standardized in order to ensure quality. My experience is that it is easy to make a standard, but very hard to get everybody to follow it. Visiting a Toyota factory, I noticed how everybody at the plant indicated with their hands that they are looking right and left before crossing a street. Finally they pointed straight ahead before crossing. During the entire two hour stay at the factory I did not see anyone breaking this standardized way of crossing. Independent of the number of people crossing, everybody did it according to the standard. This practice is called pointing-checking (yubisashi-kakunin) and is a common Japanese practice for dealing with safety checks.

In development, other methods are used to ensure quality. The main method I experienced during different company visits was the extensive use of checklists. Checklists are used during all different steps of the development process. The checklists reflect the engineering knowledge accumulated over time. When a failure is found during test activities the relevant checklist is updated. In this way, checklists are used to ensure that errors never will be repeated, as well as to transfer knowledge. When visiting one company they explained that they not only test to the specification, but also to twice the limit of the component.

Japanese companies invest a great deal in training new employees. New students are therefore educated during their first years at the company [63]. One company mentioned that 70-90% of the time was spent on education during the first year and 50% during the second year, another company mentioned that even math was taught. One possible explanation as to why companies invest so much in their employees is the Japanese concept of lifetime employment. If engineers move, they usually move from original equipment manufacturer (OEM) to supplier; it is rare to move from supplier to OEM. Unlike in the west, a salary decreases on changing employment [36].

Lean literature argues that companies should establish long term relationships with a small number of suppliers, so you can really know them, and they rely on you for business [91]. Studies made by Fujimoto and Clark [18] shows that Japanese OEMs involve suppliers to a much higher degree than in the US and Europe.

The companies we visited also seemed to be working closer to the supplier than western companies. This is supported by a study of patent applications [57] made by Japanese automotive OEMs and their suppliers. It shows that the ratio of shared patents applications made by Toyota is twice as high as its Japanese competitors. This study indicates that Toyota works very closely with its suppliers in the early phases of development. It also shows that Toyota, in this regard, is different from its Japanese competitors. Research and Development in Japanese companies are often geographically separated (Figure 6). Corporate research seems to include what is commonly defined as research and advanced engineering. The manufacturing we visited includes development and production. Development is located by the production site in order to support production.

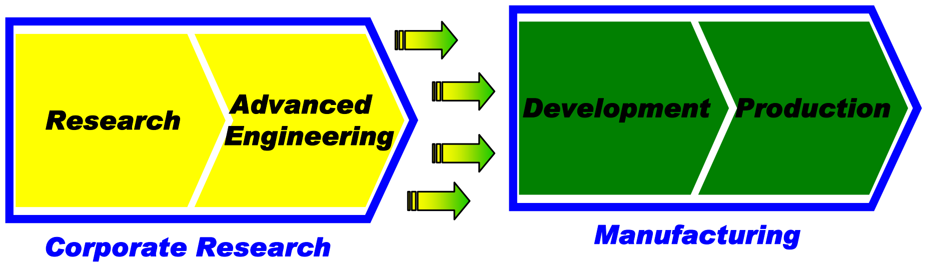

Figure 6 Organizationally and geographically separated functions in the R&D organization.

During my visits, two explanations were given as to why research is separated:

- To attract top students they need to be close to the top universities.
- To keep research disconnected from production and development.

Having a separated organization could entail complications for the architecture of the system. During our study trip the companies we visited showed a great deal of interest in how to achieve reuse and product modularization, although there was no hard evidence found to show that the level of reuse is low.

1.2 Research scope

The purpose of this research is to improve how systems architecting is performed within software-intensive systems. A more specific purpose is to find the success factors for different methods used within the industry.

The overall goal of the project is to investigate how system architecting is performed in the automotive industry and how it can be improved by the use of Lean Thinking. To achieve this goal, architecting will be studied in various industrial settings in order to find a successful use of methods and areas for improvement. Lean Thinking will be studied to find how it can be applied within the architecting of software-intensive systems. New methods that can improve decision making when developing software-intensive systems will be developed and evaluated. The results of this research are increased knowledge in this field. Case studies of the system architecting process at the different companies will result in an increased understanding of the system architecting process. The analysis of the processes will provide the companies with inspiration for improvements and ease future academic studies within the field. Four research questions (RQ) are stated in the following sections and their relationships to business, architecture, process or organization [90] are mapped in Figure 7.

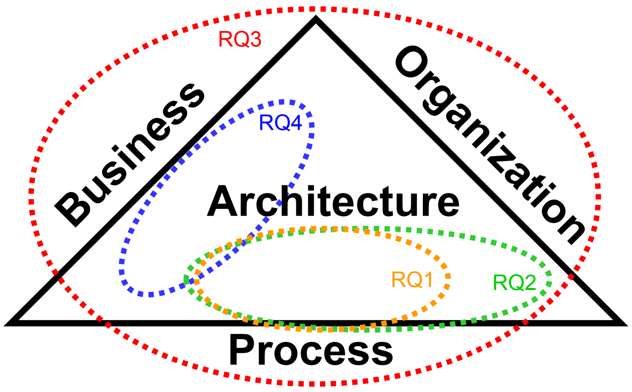

Figure 7 The relationships between the research questions (RQ).

1.2.1 Research question 1

Lean Thinking aims to improve the development process by creating a cadenced flow. Understanding the process and the methods that are used and available is important in order to improve the processes. The architecting process described in the documentation is generally not the same as the real process. The real process needs to be mapped to find what artefacts are produced and for what customer. If unnecessary iterations and artefacts can be eliminated, the process will be faster and more efficient. One hypothesis to be tested is whether Value Stream Mapping is a suitable method.

How can an architecting process be mapped in order to identify improvements?

1.2.2 Research question 2

To be able to compare different practices, one needs to understand the context of the specific architecture under observation. The context might be influenced by lifecycle, procurement strategy, organization, volume, and guiding principles. Different methods can help you find the best solution, but some will be more effective than others. To better understand how solutions are reached one needs to study how engineering tools are really used, for example, and how tasks are performed. Depending on the context, different methods will be more suitable than others and also affect what tasks that needs to be done. When you know what tasks are needed you can start improve in order to create an efficient flow through the architecting process.

What tasks are performed in the process of architecting automotive embedded systems?

1.2.3 Research question 3

Just because a method is used it does not mean it will be successful. How roles are distributed throughout the organization and how information is communicated is probably important. According to Cedergren [17] product development is to be considered successful if its products not only satisfy the needs of its customers, but also creates value to its stakeholders at large. The methods used within the architecting process should then be considered successful if the results are valuable to its stakeholders and the architecture fulfills the needs of the customer Success might also depend on issues such as what authority and responsibility is given to the involved stakeholders. The answer to this question can be used as a guideline for when to use different methods.

In what context are the methods used within the architecting process found successful?

1.2.4 Research question 4

Embedded systems are evolving and changes are continuously being introduced. A Lean architecture needs to be flexible in order to reduce the number of variants [63]. To cope with those changes, the system needs to be designed with the right amount of flexibility. A product that has an architecture that can absorb new functionality will be able to react quickly to new customer demands and thereby provide customer value. RQ 4 aims at developing methods that will aid the architect when making architectural decisions.

How can one value the flexibility needed to withstand an uncertain future in automotive embedded systems?

1.2.5 Contribution

The main contribution of this thesis is to present how Lean Thinking can be applied to system architecting. The contribution is presented in paper A, B, C and D:

- In Paper A, Value Stream Mapping is adapted to be suitable for identifying potential improvements to the architecting process. A case study presents, in general terms, what types of waste and improvements could be found.
- The different tasks performed when architecting automotive embedded systems are presented in Paper B. To understand how different methods are suitable in different contexts, a case study is conducted.
- The contexts of the different companies, as well as the architecting practices, are compared and analyzed in Paper C.
- To improve how decisions are made in the early phases of development, a method and process is presented in Paper D. The method shows how flexibility can be valued.

1.3 Thesis outline

The thesis contains an introductory part and a collection of the articles metioned previously. The introductory part is divided into seven chapters. Related work is presented in the next chapter. This is followed by the research methods used to study each of the questions presented in Chapter 1. The research results and their relation to the appended papers are described in Chapter 4. The appended papers are summarized in Chapter 5. Finally, the results are discussed in Chapter 6 and conclusions and future work is proposed in Chapter 7.

Chapter 2. Related work

This section describes research in the field covered by this thesis and provides a frame of reference for the concepts used. The architecture reflects the business goals of a company. Product development involves many stakeholders that have an interest in the system during its entire life-cycle. The development process will include stakeholders from departments such as purchasing, aftermarket and sales. How those stakeholders are organized within the organization will affect what solutions are chosen and also the system architecture. Changes in the concerns architecture (Section 2.1), business (Section 2.2), process or organization will, according to van der Linden et al [90], have an impact on other concerns. The competence and skills of the individual people working in the organization will influence how the work is performed (Section 2.3).

Architecting is affected by various support functions surrounding the development activities. Computer aided tools are necessary to handle the large amount of information needed to make the right decisions and to preserve knowledge of previous decisions. Processes are needed to aid the architecting activities and the process can be improved using different approaches, as discussed in Section 2.4.1. There are many different methods available; the methods most suitable for architecting are presented in Section 2.4.2.

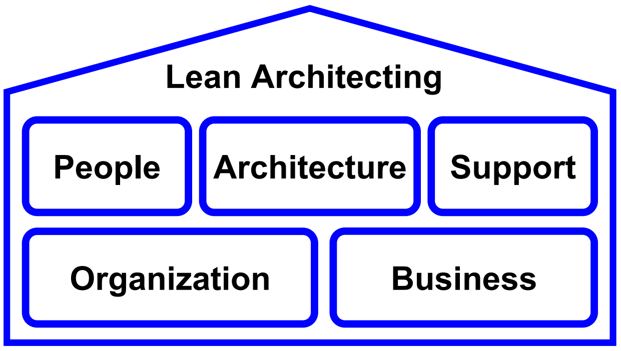

Figure 8 The context of Lean Architecting.

2.1 The architecture

Architectural trends in the automotive domain are currently changing, but there has been a philosophy of “one function – one ECU”. There have been various attempts to resolve these issues. Academic projects like DECOS [69] have proposed standardized system architectures. The EAST-EE project [27] proposed an architecture description language to support component-based development. In 2003, a number of automotive OEMs and suppliers launched the Automotive Open Systems Architecture (AUTOSAR) with the aim of creating an industry de-facto standard for automotive software. The main motivations behind AUTOSAR are very similar to those addressed by Ommering and Bosch [24], where the basic arguments for software product lines are listed; size, complexity, quality, diversity and lead time reduction. It is also pointed out that it is hard to succeed in combining components from different companies without a common global architecture. To address those topics, AUTOSAR both defines an architectural framework and a supporting component framework to achieve an instantiation between the basic hardware and the application software. Large parts of the automotive industry are now adapting to this standard [42] and AUTOSAR will change how software is integrated into embedded systems within the automotive industry [3]. The transition from a proprietary architecture to AUTOSAR involves many stakeholders, ranging from developers and testers to purchasing.

Most automotive software is now bought at a fixed price associated with a specific hardware. AUTOSAR will enable sales of pure software components; it will therefore change the acquisition process and the terms on which software is priced. The supplier structure of the automotive domain is in this case both a driver and a challenge.

Architectural changes in distributed embedded systems are either evolutionary or revolutionary [4]. Evolutionary changes to the system are continuous improvements and increasing functionality. Revolutionary changes are made when large fundamental changes are needed. A common scenario occurs when the technical debt [25] of the system has increased so much that the cost of evolutionary changes become too expensive. Due to the technical debt in terms of “spaghetti code” or undocumented system architecture, the cost of changes is increasing. Revolutionary changes could also be caused by market changes or when adapting to a new standard, as is the case with AUTOSAR.

One way to capture the knowledge stored in existing architectures and the vision of future needs is a concept known as Reference Architecture [20]. A

Reference Architecture is a standard or template based on the previous best practice of the domain. The evolution of an automotive Reference Architecture is described by Eklund et al. [30].

Another way to manage the architecture is through the use of Software Product Lines. The term Software Product Lines is defined by Clements and Northrop [19] as:

“A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.”

The practice of Software Product Lines involves long-term strategy and the management of the core assets in order to increase reuse and decrease time to market. Industrial uses of Reference Architectures [20] are in some cases very similar to the use of Product Line Architectures [9].

The use of Software Product Lines has a great deal in common with the traditional manufacturing of mass-produced products and, in this way, is nothing new. What differentiates Software Product Lines from component-based development is the larger scope, involving both business and management aspects.

Blackenfelt [10] discusses why modularization should be applied to product development. He concludes that the freezing of interfaces should be the last step in the modularization project. Similarly to software product lines engineering, it is argued that the product will benefit from the investment after the introduction of a few family members or variants (Figure 9).

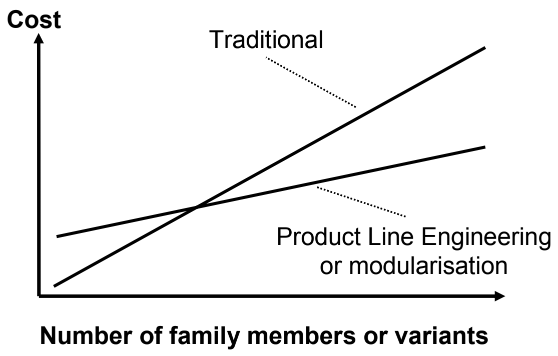

Figure 9 Total product cost in relation to the number of variants [93].

2.2 Business domain

The business domain of a company has a large impact on the architecture. Factors such as the type of customers, sales volume and the lifecycle of the product will influence the architecture. A company that builds a small number of variants of one system with low complexity and few developers is not likely to invest anything in developing the architecture. Companies who develop software-intensive systems have the common challenge of managing the complexity of mechatronic systems. The systems addressed in this work often have a lifecycle of 20-30 years and long term service contracts.

The strategy of using one common architecture for all variants seems to be a viable solution in order to keep cost and complexity at a low level, but this is not always the case. In the case of Bosch [81, 84] it was found that they needed to implement a low-end and a high-end architecture in order to stay competitive in different market segments.

The long term strategy of the company has great implications for the architecture. If the strategy is to lead technology development in one field, they need to keep the knowledge in-house or perform co-development with strategic partners. A component developed in-house or in close co-operation is much more likely to fit into the existing architecture than a purchased component. The procurement strategy of make or buy has great implications for architecture and the project's success [35]. The complexity of the component in comparison to customer value is one way to make the trade-off. Figure 10 illustrates a common strategy of keeping components with high complexity and high customer value in-house to guard the expertise. Simpler components with low customer value are found to be more beneficial to outsource because they are not viewed as core competence and are easier to specify.

Commercial vehicles and passenger cars are part of the automotive domain and similar in many ways, but even within one industrial domain there are many differences. The main purpose of commercial vehicles is transportation of goods, but the transport task differs with each customer and market. The customer requirements on the vehicles are very different. A commercial vehicle must manage to run 300,000 km per year; breakdowns do not just influence the driver, but also the delivery time of the goods being carried. Commercial vehicles have a lot in common with passenger cars, much of the functionality is found in both segments [100]. The differences in business aspects affect the architectural requirements.

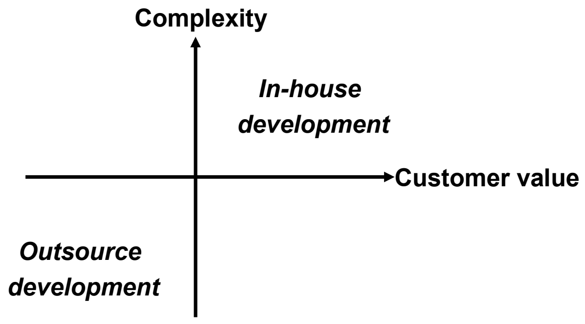

Figure 10 Procurement strategy - make or buy.

2.3 People and organization

In 1951, Arthur Raymond, architect of the DC3 aircraft, stated that in order to be successful the design area should be close to both testing and prototyping [72]. In 1968, the origin of what later would be known as Conway’s Law [21] was presented: “Any organization that designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure.”

More recently, Coplien [23] addressed this issue in software development with the following technique: “Be attentive to domain partitioning. In particular do not split a domain across geographic locations or architectural units.” This technique is often seen implemented in the automotive domain where development is divided into the architectural units of the vehicle, although this was done long before the introduction of software.

The architects carry knowledge across different functions. The geographic and organizational location of the architects is therefore important. Knowledge sharing between architects is affected, whether they are colocated or separated. The organizational structure of a company often mirrors the product architecture of the products it produces. Henderson and Clark [43] explain how this tie between product and organization causes difficulties when the architecture need to be changed. One example of how the performance of the organization is affected by the architecture is given by Reinertsen. He argues that developers will be more effective if they are organized around the modular structure of the product [74]. Unphon and Dittrich [86] conclude that one must consider the organization and business domain when adopting a product line architecture. In a study of eight different software development organizations [87], it was found that the architecture is maintained and evolved through face-to-face communication rather than documents. Coplien and Bjørnvig argue that developers and testers should be friends and there should be ongoing testing support during development [23].

The size of the R&D organization, as well as the size of the system development organization, will also affect the role of the architect. In a small organization everyone will be an architect. Eventually, when it is not practical for everybody to talk to each other someone will take on the role of an architect. Or, as stated by Rechtin; "architecting is a consequence of system complexity" [73]. As the number of architects grows they will also need coordination. Coplien and Harrison [22] have developed organizational patterns suitable for software development. Three of them are presented below:

Engage customers. This stresses the importance of the development organization ensuring and maintaining customer satisfaction. This can be achieved by encouraging communication between customers and the different roles in the development organization.

Function owner and component owner. To ensure responsibility, every function and component should have a dedicated owner.

Developer controls the process. The developer should be at the centre of the process and also be able to change the process.

The type of organization will affect how communication is carried out and how decisions are made. The power centers of an organization also affect how work with the architecture is done. Nedstam [66] presents the great differences between how work is done in an organization with strong line management and in an organization with strong projects. This has also been found to be true at the companies studied in Paper B of this thesis.

Kruchten [58] suggests that the productive time spent by architects can be sorted into three categories: internal (architecture design), inwards (input from outside world) and outwards (providing information) communication and that they should roughly have the ratio of 50% internal, 25% inwards, 25% outwards. In order to work effectively as an architect it is important to understand the organization and to know who is the right person to answer the current question. Different companies have different cultures and the culture varies depending on the geographic location.

In a survey of 279 IT architects in the Netherlands, Farenhorst et al. [32] conclude that architects are lone decision makers; not very willing to share architectural knowledge, but eager to consume. A study of decision-making in Swedish and German teams [65] concludes that Sweden has flatter organizational hierarchies and that the decision-making process is less formal. This is well-known to people working across those borders, but could cause unnecessary tension if forgotten. Rechtin [61] highlights the importance of architects understanding the cultural characteristics of the organization in order to be successful.

2.4 Architecting support

The architecting process needs to fit into an overall product development process and the work should be supported by methods and sub-processes. Tools will be used to document and analyze the architecture. Those tools will mostly provide support, but often also constrain and limit the way architecting is done. The architect will need to make choices and sometimes, more importantly, communicate the pros and cons of different solutions. Visualizing different possible solutions (Figure 11) is therefore important even for the solutions that are not chosen.

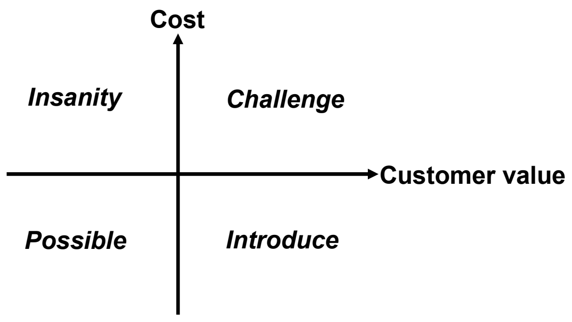

Figure 11 One way to visualize different possible solutions.

Different architectural frameworks exist in order to effectively describe and communicate architectures. Architectural frameworks are often used to describe the details of architecture. Greefhorst et al. [38] have performed a comparison of many of the existing frameworks. They found that there are many differences between the existing frameworks and conclude that the differences partly depend on different original goals and context. Describing an architecture is also standardized in IEEE 1471 [49]. The frameworks used by the defense industry, for instance, are shaped by a long systems engineering tradition and driven by requirements.

2.4.1 Process and process improvement

The importance of embedded systems, together with a growing organization developing a system that is becoming more complex every day, makes the performance of the development process crucial. High performance is the result of efficiency and effectiveness in each activity [16]. Effectiveness is defined as doing the right thing and efficiency as doing the thing right. The architecting process involves many stakeholders who all produce knowledge needed to develop the architecture. Liang et al. [60] present a process based on architectural knowledge for software architecting. Architecting of software systems are described in many pieces of literature; Eeles [29] presents the process of software architecting of IT system [29]. Most available architecting processes are software oriented, though the architecting method CAFCR [64] is one exception. This has a focus on what the internal and external customer wants on a system level of embedded systems. Through comparison of available processes, Hofmeister et al. [45] have developed a generic process (Figure 12) for creating and maintaining an architecture. Inspired by Scrum [82], the process emphasizes the need for a backlog to keep track of issues found in the architecture.

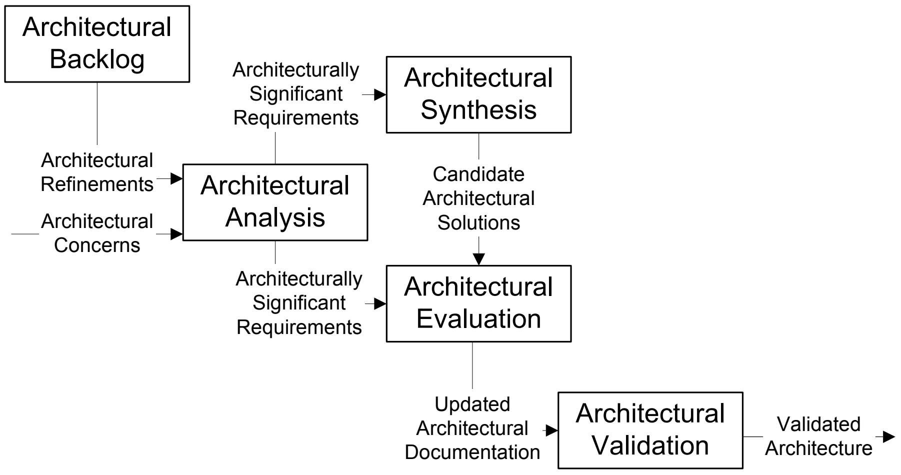
Architectural Backlog
Architecturally Significant Requirements
Architectural Synthesis
Architectural Refinements
Architectural Concerns
Architectural Analysis
Candidate Architectural Solutions
Architecturally Significant Requirements
Architectural Evaluation
Updated Architectural Documentation
Architectural Validation
Validated Architecture

Figure 12 A generic process for creating and maintaining an architecture, derived from [45].

Different methods can be used to assess the process and to find improvements. Quality systems such as the Integrated Capability Maturity

Model (CMMI) and SPICE can be used to perform process assessment. The results of a systematic literature review show that there is a great variety of maturity models and that the trend is towards more specialization in the models for specific domains [97]. In the European automotive domain, automotive SPICE is the most common model. Axelsson [5] presents a maturity model for architecting embedded system product lines based on CMMI.

Process improvements (Figure 13) can be made through radical (Kaikaku) and continuous (Kaizen) improvements. Continuous improvements are often based on a broad effort involving everyone in the organization, while radical improvements are top-down initiatives with the goal of dramatic results [98]. As with all large changes, the risk involved increases when making radical improvements. The same analogy can be made with the information batches flowing through the organization, large batches increase the risk. Small batch sizes reduce the variability of flow, reduce risk and accelerate feedback, which increases motivation [75]. This is very similar to the key drivers of SCRUM [82], which are to increase the speed of development and to add energy and focus.

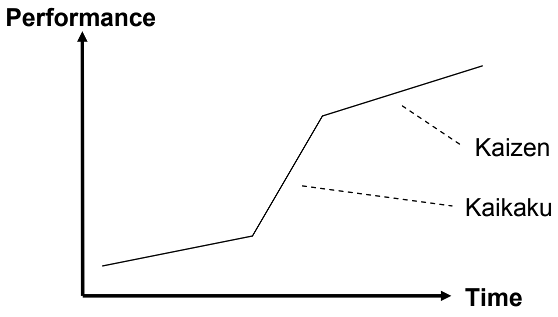

Figure 13 Making radical (Kaikaku) and continuous improvements (Kaizen) to improve performance.

It is hard to differentiate between Lean and Agile. The two practices share the same values; they both empower people to achieve results and are keen to adapt and improve the processes to fit current needs. One difference between the two is the scope of implementation. Lean is applied to all the different parts of a developing organization and Agile is focused on software development. Differences between implementations of Lean software development are more likely to occur because of cultural or organizational factors, compared to Agile software development. A common criticism of Agile is that architecting is insufficiently emphasized and that the architecture emerges during development [56]. However, a literature review

[12] of architecting within Agile development concludes that there is no empirical research to support or contradict this assumption.

A systematic review of agile software development methods [26] found that most empirical research studies focused on extreme programming. Very few studies were found on Lean software development or Scrum. Kettunen has performed a comparative study of manufacturing methods and Agile software product development. He concludes that the there are not many profoundly new ideas compared to earlier manufacturing methods [56]. Coplien [23] makes a comparison of Lean and Agile and claims that Agile is about doing and Lean about thinking and doing. He also claims that Lean focuses on process and Agile focuses on people, although seminal work [62, 79] on Lean says otherwise.

In the literature, there is little work on how Lean can be applied to the process of developing software-intensive systems. Recently, Lean has been applied to the overall systems engineering process of INCOSE [67]. Poppendieck and Poppendieck [70] present how Lean can be applied to the software development process. In their work, typical wastes to be found are hand-offs between individuals, switching between tasks and adding extra features.

Browning et al. [14] discuss how process modelling of a product development process is conducted and present a simple framework. Furthermore, they argue that process modelling should be tailored to that environment.

In [80] a model for evaluating the degree of leanness of manufacturing firms is presented. This model was based on the initial research done by Karlsson and Åhlström [51] who developed a method for measuring the change progress in production.

Value Stream Mapping is presented as a way to find waste. There are many different techniques available for process modeling, but Value Stream Mapping (VSM) is different to other process modeling tools due to its focus on value creation [14]. Value Stream Mapping (VSM) was initially a tool for improving the manufacturing process [77] and has been shown to be effective within manufacturing [48]. The method is now also used within many other disciplines. This method is explained in detail and adapted to the architecting process in Paper A of this thesis.

2.4.2 Analysis methods

In order to make an adequate design decision, one must consider numerous factors. There are obvious aspects such as size, cost and performance, yet other less tangible factors are very important; factors such as customer preferences, development cost, production volume and time to market. All these factors – and many more – influence the final decision.

Architectural decisions are made when selecting components and allocating them to subsystems that then are combined into a system. These decisions can be made on different levels with various impacts and predictability [33]. This section explores the available architecting methods.

SWOT analysis (Figure 14) is one common way [15] of evaluating design alternatives and enables a clear visual view of the trade-off. Trade-off curves (Figure 15) are used [39] to visualize the design space and enable efficient communication of knowledge. A practical example could be to compare the accuracy versus the cost of different sensors in order to make the best choice. An extension of trade-off curves using Data Envelopment Analysis [68] presents how resource utilization of different user functions can be evaluated. An example of one of the more commonly used [2, 50, 78] formal evaluation methods is Pugh's evaluation matrix, which was developed by Stuart Pugh in the 1980s [71].

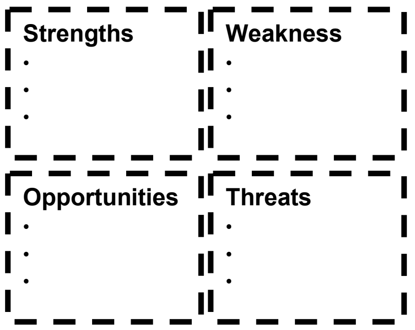

Figure 14 SWOT analysis.

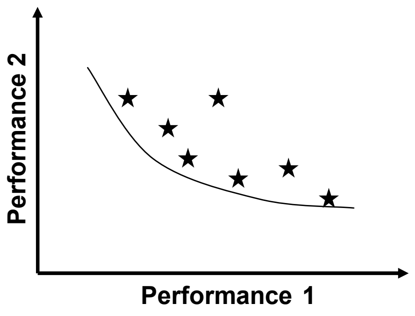

Figure 15 Trade-off curve.

The Design Structured Matrix (DSM) [13, 89] has been used to evaluate architectures in various cases. Larses [59] uses the balanced scorecard to balance the important perspectives in system design for the complete E/E system, in combination with a cluster analysis using DSM. He found that the combination of quantitative and qualitative data can provide good decision support. Problems that were found were the lack of input data and the need to consider procurement and change aspects when reusing the architecture.

The Architecture Trade-off Analysis Method (ATAM) is a method for evaluating different architectural approaches and was developed by the Carnegie Mellon Software Engineering Institute [54].

The goal of ATAM is to assess the consequences of architectural decisions in the light of quality attribute requirements and to perform an analysis in a repeatable manner. Each stakeholder has different quality attributes that they consider to be the most important ones. The top level attributes are typically attributes like safety, performance, maintenance and maintainability but the number of attributes can vary from case to case.

A utility tree is created with input from all stakeholders. The utility tree is only constructed by the architects and the project leader and will therefore only show the architects' view of what is important to the system. The next step is to perform a brainstorming for scenarios. The scenarios are made up by all stakeholders. The scenarios are comparable to the leaves of the utility tree.

Each stakeholder is given a number of votes, typically 30% of the total number of scenarios, and then votes for what each stakeholder considers to be the most important one. The result of the vote is then compared with the result from the utility tree. If the result is the same, it is quite certain that the most important attributes are being considered in the architectural decision. If not, the view of the most crucial attributes for a successful architecture differs between system architects and other stakeholders. In this case, some kind of reasoning is necessary between the system architects and other stakeholders in order to conclude which are the most important parts.

ATAM is a structured method that is tailored for analyzing architectures, which ensures that the right questions are asked. The main result of ATAM is an identification of the potential architectural risks. The greatest benefit of the method is that it can provide a common understanding of the importance of different quality attributes.

ATAM does not provide any support for evaluating different design alternatives. An extension of ATAM made by Wallin et al. [88] provides support for evaluating different design alternatives with the use of paired comparison of scenarios.

The Cost Benefit Analysis Method (CBAM) is an extension of the ATAM and was also developed by the Carnegie Mellon Software Engineering Institute [53]. It uses the quality attributes derived from the ATAM, but also considers cost when reasoning around the most suitable architecture. A second iteration of CBAM also takes the uncertainty of the used figures into account, but it does not consider flexibility and architectural evolution.

Using options theory is one approach to dealing with the high level of uncertainty when making design decisions in the early phases. The theory derives from finance, where an option is the right but not the obligation to exercise a feature of a contract at a future date [46]. Real Options are discussed in detail in Paper D.

The benefits of using a structured method are widely accepted in academia, but various studies [2, 39, 78] indicate a very low industrial usage. The proposed solution is to present success stories and to further investigate the needs of the industry [78]. An article published in the Journal of Engineering Design attempts to answer the question of why industry ignores design science [34]. The article claims that industry solves problems by using the knowledge of experienced engineers, which is often faster than using a structured method. One of the answers presented is that many structured methods require information which is often not present or very resource consuming to generate. Ken Hurst [47] presents the following reasons for why a structured method should be used:

- Time wasted in pursuing wrong alternatives to the detailed design stage is avoided.
- Causing visible decision-making helps to ensure the process is repeatable.
- The ability to evaluate the thought processes of others is developed.
- The designer can defend decisions made in discussions with managers or clients.
- A designer with no previous experience can carry out a sensible evaluation of alternative concepts.
- The process of concept selection stimulates new concepts or encourages a combination of concepts.

Ulrich and Eppinger [85] present a similar list of benefits and emphasize that the use of a structured method provides customer focus and a more competitive design.

2.5 Lean development

Lean development focuses on creating re-useable knowledge - knowledge that contributes to the profitability of future operational value cycles and that can ideally be used for many projects [91]. The concept of Lean production was defined in the literature by Womack et al. [96], but derives from the working methods developed by Toyota in the 1950s.

Lean methods focus on increasing customer value and on the people who add value. A Lean-based company encourages its employees to perform continuous improvement and to learn. This is done by cross-functional and parallel work and a high degree of standardization in order to improve and to share knowledge across the organization. Lean production is achieved by the careful planning of a production line in order to optimize the production flow to meet customer needs. Each assembly station is arranged to minimize unnecessary motion and material transportation. Each assembly station is assigned defined tasks to be finalized at a specific time in order to achieve a balanced flow throughout the production line. A balanced flow means that the results are delivered on time without waiting or over-production.

An important starting point for lean product development is to view product development as a process, and like any other process there are repeated cycles of activity [63]. From a process perspective, there are many activities that are shared between different development projects. An increased flow is achieved by eliminating the waste in a process, thus new products can be brought to the market at a higher pace.

There are two main differences between manufacturing and the early phases of product development. Firstly, in product development the flow does not consist of materials, but more often of information and knowledge in different forms. The different organizational and geographical locations of the stakeholders influence how this knowledge is shared. Secondly, the product development process does not consist of one flow, but instead iterations are frequent and different concepts are developed in parallel. For coping with the rapid changes made in product development, Ward [92] makes an analogy to surfing: in order to be in control you need to constantly adjust, changing direction and shifting from wave to wave instead of trying to control the waves. Creating a cadenced flow of information is one way to be able to react to the changes. One practical example is how short meetings are used at Scania to quickly distribute information to the organization (Figure 16).

Kennedy et al. argue that Toyota standardizes their knowledge into checklists and reviews all their design against these standards. Those checklists are updated after every project. Product development consists of two value streams (Figure 17) [55]:

- The product value stream is unique to each project. Project X is not started until the alternative designs have been evaluated and decided upon. When the project starts, the risk should be very low. Knowledge acquired during and after the project is fed back into the knowledge value stream.
- The knowledge value stream consists of knowledge generalized for visual flow across projects and organizations. Checklists and A3 documentation are used to carry this knowledge. Architectural knowledge such as patterns and guidelines should be part of the knowledge value stream.

Figure 16 Short weekly stand-up meetings at Scania.

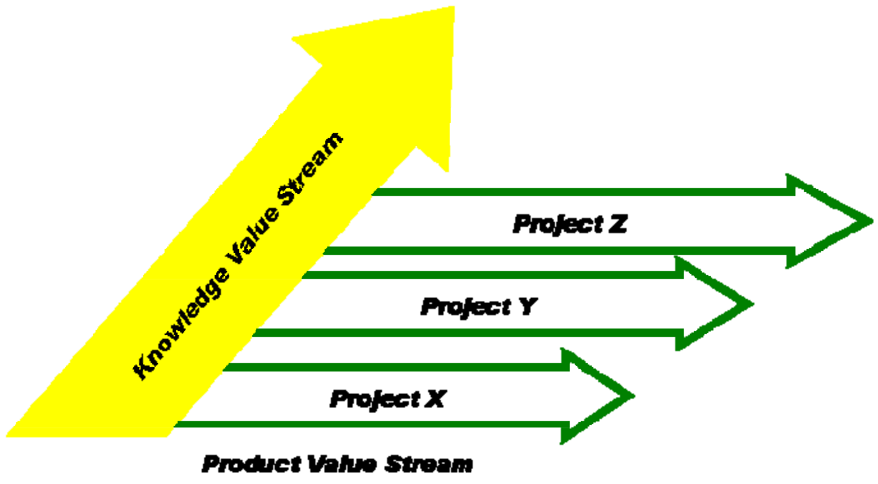

Figure 17 The two value streams of product development [55]

Allen Ward [92] claims that 20% of the time spent in product development is value adding time. Nonvalue-creating time such as administration work occupies 20% and the remaining time is waste. This fact would suggest that optimization is possible if we identify the wasteful activities. It is common to define seven types of waste [63] and value stream mapping is one method to identify the waste within any process.

According to Allen Ward [92], the most frequent waste in development is waste of knowledge. He divides knowledge waste into three categories: scatter, hand-off, and wishful thinking. Scatter is described as actions that disrupt the flow of knowledge. This disruption can be due to communication barriers and the use of inappropriate tools. An example of knowledge waste created by hand-offs is to functionally move people around rather than assigning them to one task from beginning to end. Waste due to wishful thinking is, for instance, testing according to specification rather than testing to learn about the limits of the product. An example of waste in terms of discarded knowledge is testing to specifications (rather than testing to failure), which throws away the opportunity to find out when and how the design actually fails [91]. Knowledge from testing would then be fed back to the knowledge value stream (Figure 17) as A3 documentation and engineering checklists. Liker [63] points out the difficulty of transferring tacit knowledge compared to explicit knowledge. Explicit knowledge, such as mathematical equations and historical facts, is often easier to store. Tacit knowledge is often more diffuse, similar to what is taught through apprenticeship. Toyota creates their learning network through activities such as technology demonstrations, checklists, know-how databases, mentoring and lessons learned [62].

Seven wastes	**Examples**
Overproducing more or earlier than the next process needs	Batching, unsynchronized concurrent tasks
Waiting for materials, information, or decisions	Waiting for decisions, information distribution
Conveyance - Moving material or information, or decisions	Waiting for decisions, information distribution
Processing - Doing unnecessary processing on a task or an unnecessary task	Stop-and-go tasks, redundant tasks, reinvention, process variation-lack of standardization
Inventory - A build up of material or information that is not being used	Batching, system over utilization, arrival variation
Motion - Excess motion or activity during task execution	Long travel distances, redundant meetings, superficial reviews
Correction - Inspections to catch quality problems or fixing an error already made	External quality enforcement, correction and rework

Table 1 Applying the seven wastes within product development [63].

Chapter 3. Research methodology

In this research project the system architecting process of several large companies has been studied. Traditional research is often done through the use of quantitative methods. In natural science, experiments are used to validate a model through measuring a series of samples in a controlled environment. Experiments are also applied to software engineering and are used to investigate the qualities of different programming languages or code inspection techniques [94]. However, experiments on real industrial processes are difficult because of the large number of dependent variables. The variables are often very hard to control and to measure. The cost of performing an experiment that captures the complexity found in an industrial setting would be very high. The methods considered in this research are surveys and case studies.

The typical feature of a survey is, according to Robson [76], the collection of a small amount of standardized data from a relatively large number of individuals in a known population. Disadvantages of surveys sent to different organizations include the risk of misunderstanding and the risk of the respondents not taking the exercise seriously [76]. Different companies perform architecting in various ways and there are many different factors that have an influence. Many of those factors are thought to be soft factors [31] that are hard to find through a survey.

System architecting is, as previously described, a cross functional activity, which makes it very difficult to measure or control the process. According to Yin [99], case studies are especially suitable when the boundaries and context are not clearly evident, and case studies can be both quantitative and qualitative [94]. System architecting has boundaries to many other processes and is influenced by its context. Case studies are often used to investigate real industrial processes and therefore suitable for this research. Evidence can be collected from different sources in a case study. Documentation or archival records can be used if they are retrievable [99], but access to architectural information in industrial settings is often found to be limited. This is primarily because of commercial issues, but also due to the limitation of available documentation. Another source of case study evidence can be retrieved through either direct observation or participant observation [99]. The strength of observation is that the process can be studied in its context and in real time, though this is often time-consuming. Industrial projects are not always on time. Observing real projects would add uncertainty to the research plan. Interviews are the main case study methodology chosen for this research and are discussed in detail in the following section.

3.1 Research design

Case studies have been used to answer all research questions, but the method has been applied somewhat differently for different questions. This section describes the methods used and how different threats to validity have been treated. “Case study is a strategy for doing research which involves an empirical investigation of a particular contemporary phenomenon within its real life context using multiple sources of evidence.” [99] Semi-structured interviews have been used to answer RQ 1, 2 and 3. A semi-structured interview has predetermined questions, but the order can be modified based upon the interviewer’s perception of what seems most appropriate. Question wording can be changed and explanations given [76].
To answer RQ 4, a case study was developed to investigate the usage of the developed method.

3.1.1 Method used for research questions 1 and 2

The data used to answer RQ 1 and 2 were obtained through analysis of semi-structured interviews at Volvo Cars and Scania. In total, 11 interviews were performed by the two authors who are native to Scania and Volvo Cars respectively (see [11] for the definition of “native” in this context).

1. The questions were developed and tested on people with similar roles at both companies, who were not included in the study.
2. All the architects who were available and willing to participate were interviewed, which resulted in more than half of the architects at each company participating, 4 at Scania and 5 at Volvo Cars. In addition to this, the managers for the architecture groups were interviewed at both companies, bringing the number of interviews to 11.
3. The interview was led by one person while the other took notes.

4. Each respondent had the possibility to read and comment on the notes from their interview in order to correct any misunderstandings, errors or other mistakes in the transcriptions.
5. The results were gathered in a database and analyzed.
6. The final results were reviewed by the manager at each participating company. The results of the study were presented for a broader audience at both companies.

3.1.2 Method used for research question 3

In order to answer RQ 3, the previous study was extended to include four additional companies. The same set of questions was used but a slightly different procedure was used:

1. The same set of questions was used.
2. The companies were selected through established contacts. All companies have significant development of software-intensive systems, but are different in size and production volume.
3. The architects were identified in collaboration with the contact person.
4. At least two interviews were held with architects at each company. With the respondents' permission the interview was audio recorded.
5. The results of the study were presented to a broader audience at each company. During the presentation the situation at the visited company was also discussed.
6. Questions about the characteristics of each company were answered by the contact person.
7. The results were gathered in a database and analyzed.
8. The results were reviewed by the contact person at each participating company.

3.1.3 Method used for research question 4

A methodology has been developed to value the flexibility of design alternatives. To analyze the developed methodology and its industrial worth, the methodology has been applied to a real case in the automotive industry. Information about a current architecting case was acquired through discussion with the responsible architect. All available written information regarding the case was obtained and reviewed. The real case was then used to test the developed methodology and to evaluate the actual case. The result from the case study was presented to the architect in order to be able to discuss its usefulness.

3.2 Validity

There are different types of threats to validity that need to be considered when conducting research. The different threats to validity are presented below, including the countermeasures made to improve validity.

3.2.1 Construct validity

Construct validity ensures that the studied artifacts can be applied to analyze this exact problem. To avoid bias in the respondents, a minimum of two people were interviewed at each company. All the written documentation of the interviews conducted in this research has been reviewed by the respondents. In the study used to answer RQ 3, the interviews were conducted by the author alone. Those interviews were recorded with the respondents' permission and were therefore not reviewed. The final papers have also been reviewed by representatives of the companies in order to limit the risk of misunderstanding. The results of both studies have been presented for a broader audience at the participating companies, including a larger number of architects and managers. The presentations were another way of limiting the risk of misunderstanding. The working experience of the author will also help to ensure construct validity.

3.2.2 Internal validity

Internal validity ensures that the conclusions we draw from a study are the only ones possible and have not been affected by another possible cause. Internal validity is ensured by doing pilot interviews with informants similar to the ones questioned in the study. The questions can thereby be altered to ensure internal validity. The working experience of the author as an architect limits the risk of misunderstanding when talking to other architects. The analysis of the interview data used for RQ 1 and 2 was performed by the author, who is native to Scania, and another researcher, who is native to Volvo Cars. The analysis of the interview data used for RQ 3 was performed by the author, but using the method developed in the previous study.

3.2.3 External validity

External validity is the degree to which the conclusions in the study would hold for other organizations and at other times. Scania is used as a case-company in all parts of this research. The major threat to external validity is the degree to which the conclusions would hold for other companies developing software-intensive systems outside the automotive industry; therefore five other companies are included in the study of RQ 3. The companies studied are all developing long-lived software-intensive systems, but in different domains, and are of different size. All the companies have a very long history in Sweden, which is a disadvantage for external validity. This weakness is somewhat compensated by the fact that three of the companies have development outside of Sweden and all of them offer their products on a global market. Related work from other areas will also serve to support the validity of our studies.

3.2.4 Reliability

Increasing reliability is about minimizing faults and biases in a study and making the results repeatable. Reliability is ensured by well documented and planned case studies and interviews. Interview data has been stored in a database that has been used to analyze the data. Only a few questions used in the interviews have been published due to page limitations. The interview questions are to be made available online in the submitted journal paper. The study would be repeatable if it was repeated under the same circumstances. Unfortunately, circumstances do change; new people are employed, projects are dealing with different current issues, company strategies are changed and organizations are restructured. So for the part of the research that studies processes this might be difficult, because industrial processes are and should be continuously evolving. The results are therefore momentary views of a specific organization. The knowledge gained can still be transferred to another organization. The method and analysis would be repeatable and could therefore be used at another time for a longitudinal study.

Chapter 4. Research results

This section summarizes the main contribution made by the thesis. The system development process at Scania was studied in prior work [39], to identify current practice within the automotive industry. The contribution of the thesis is based on the four publications. Figure 18 shows how the contributions relate to four dimensions of software engineering [90]. Paper A answers RQ 1, presenting how the architecting process can be improved by the use of Value Stream Mapping, and it thus mainly relates architecting to the process dimension. The different tasks found in an architecting process are presented in Paper B in order to answer RQ 2, and apart from the process dimension there is also a connection with organizational issues. To answer RQ 3, the successes of different methods related to the context in which they are used are discussed in Paper C, and this broad study relates to all four dimensions. RQ 4 is answered in Paper D, which presents how flexibility can be economically valued, thereby relating to the business dimension.

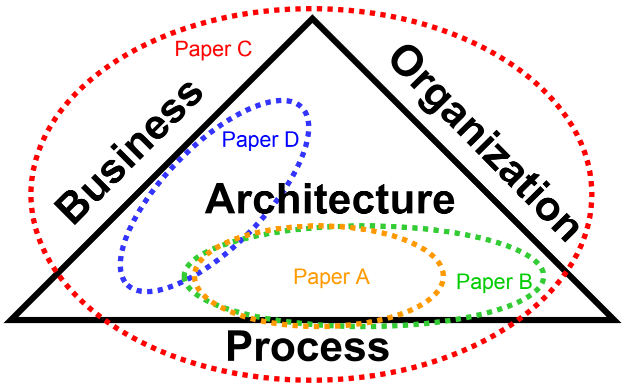

Figure 18 Relationships between the appended papers.

4.1 Paper A: Improving the system architecting process through the use of Lean tools

Value Stream Mapping (VSM) is a Lean tool used to improve value creation within a process. Using VSM on a manufacturing process has been promoted in the literature [63] and also evaluated [48]. Using VSM on a development process has also been promoted [70], but has to our knowledge not been evaluated within systems development.

RQ 1: How can an architecting process be mapped in order to identify improvements?

This paper is based on a case study at Scania and Volvo Cars and presents how Value Stream Mapping can be used to analyze the architecting process. Furthermore, it presents in general terms what types of wastes and improvements could be found. Waste could, for example, be due to handoffs, task switching, technology debt or delays. One result of this paper is to show how waste can be eliminated and maximize the value creation of the process through the use of Value Stream Mapping. An adapted version of Value Stream Mapping is found to be a suitable method for identifying improvement in the architecting process.

4.2 Paper B: Architecting automotive product lines: industrial practice

This paper aims at answering RQ 2 and presents an in-depth view of how architects work with maintaining product line architectures in the automotive industry. The study has been performed at Scania and Volvo Cars.

RQ 2: What tasks are performed in the process of architecting automotive embedded systems?

The striking conclusion and the answer to RQ 2 is the similarity between the two companies in the tasks performed when maintaining and changing architecture. The tasks mentioned by the architects at both companies are virtually identical; need ⇨ impact analysis ⇨ solution ⇨ decision ⇨ validation. The tasks do not seem to be different for architecture maintenance compared to developing a new architecture. Likewise, they seem to be the same whether it is updating a product line architecture or updating the architecture of a single-shot system. The study indicates what effect differences, such as a strong line organization or a strong project organization, have on the architecting process. It also shows what consequence technical choices and business strategy have on the architecting process.

4.3 Paper C: A comparative case study of architecting practices in the embedded software industry

In Paper C, the methods used to solve the tasks within the architecting process are mapped to the context used in the industry. The results from Paper B are generalized by performing semi-structured interviews at four additional companies. One hypothesis was that in order to understand different architecting processes one must first understand the surrounding circumstances, the context. The attributes that should be gathered in order to understand the context were derived from the literature.

RQ 3: In what context are the methods used within the architecting process found successful?

To answer RQ 3, the paper studies the current state of architecting practices in three different industrial segments that are characterized by being software-intensive. An analysis of the case study indicates how different methods are more suitable to different environments. The context of the different companies, as well as the architecting practices, are compared and analyzed. Many of the successful practices found in the study can be explained by the context of the different companies. The use of global architects with their own budget in one company is a solution for initiating long-term architectural projects without having a customer order. The high degree of documented reasoning in the studied defense company is caused by the high degree of customer-specific demands and large orders of very similar products. This forces the architects to make branches of the architecture to fulfill customer demand, and the reasoning is then used to ensure quality. The defined architecting process found at one of the automotive companies and the use of visualization tools to track progress is explained by the strong influence of Lean Thinking. Other examples of practices, such as divided architectural teams and the lack of formal architects, are more difficult to explain.

4.4 Paper D: Evaluation of design options in embedded automotive product lines

Decision-making under uncertainty is influenced by a number of factors [52], and some of them lead to less rational decisions. The use of structured methods (design reviews, checklists, and expert support) is one way to improve decision-making.

RQ 4: How can one value the flexibility needed to withstand an uncertain future in automotive embedded systems?

A method and process is developed to answer this question. The developed method evaluates flexibility, using a concept called Real Options. The method is motivated and described by using an example from automotive embedded systems. To improve the usability of the method, a structured evaluation process is defined to aid practitioners such as developers and architects. The evaluation process provides a way of valuing system designs and enables the practitioner to think about the future in a systematic manner. Our literature survey has found three research contributions [6, 8, 13] that involve the usage of real options in system design involving software or hardware. None of them explicitly addresses embedded systems or the automotive domain.

Chapter 5. Discussion

In this chapter I take the opportunity to be less formal and give my view on some topics related to this research. I believe this chapter can be an input in the current discussion and perhaps provide ideas for future research. In the first section I try to explain what Lean architecting would be. The following section presents the difficulties of researching industrial processes. Finally the industrial impact of the research results is discussed.

5.1 Lean architecting

So what is Lean architecting? There is no right or wrong answer to that question, based on reasoning about the effect of context on how work is done. Based on the knowledge gained during this research I will still give my opinion on what Lean architecting of software-intensive systems would look like. A Lean product development process is frontloaded; much work is done early in the project in order to lower the risk later when the cost is much higher. Early design decisions often have a greater impact on the overall system than the developer alone can foresee. Architecting efforts should therefore be started early and in close collaboration with the development team. In the early phases, it is important to keep the design flexible without investing too much in architecture, bearing in mind the uncertainty.

Challenges in different development projects are very different, as is the necessary architecting. The process should therefore not be fixed, although each task can be described in detail to ensure nothing is forgotten. In this way, every architecting effort will be tailored to fit the needs of the specific case. Each task will be repeated regularly and checked for improvements after each use.

Reusable architectural knowledge such as heuristics, principles and patterns should be transferred through lightweight documentation, education and mentoring. By empowering people and giving them the necessary tools, but without being obsessed by tools, quality will be designed into the system. The resulting artefacts should be challenged regularly to identify what documentation the internal customer needs. In an evolving complex system there will always be an increasing technical debt. Changes to the system will be more and more difficult to realize, therefore the technical debt must be monitored to plan for radical changes. Last but not least: in order to get a full return on the investment in Lean, the whole organization must apply Lean Thinking.

5.2 Identifying best practice

In an industrial setting, it is very difficult to produce empirical evidence to support that one method is better than another. In order to show anything, you need to measure the performance of the process before and after the change. If this is done you will get an indication of whether the change improved the process or not. If the tasks performed are comparable and made by the same group of people you will be more confident of the results. In order to study whether one improvement is better than another you would need to evaluate the alternatives. People or organizations that promote one practice often show how the performance of the development process has been radically improved by introducing a toolbox i.e. Six Sigma, Agile or Lean (Figure 19). The problem is that companies do not tend to fix something that is not broken. A development organization that is sinking in mud is very likely to accept a helping hand. The performance improvements will probably be very positive, but that does not mean that it was the best possible solution. Almost any toolbox handed to them would have helped them out of the mud and improved performance! Based on this reasoning, this research does not claim that the practices found are best practices. They are found suitable in a specific context and should be critically reviewed before being used in other contexts.

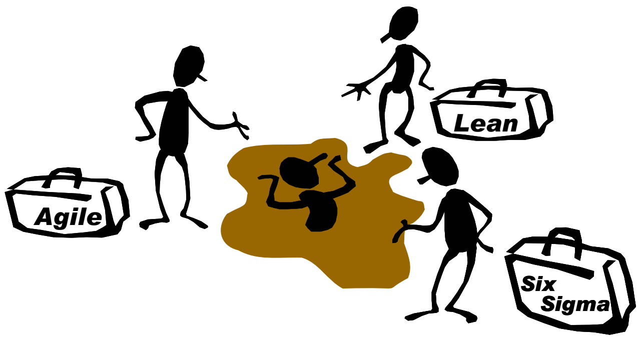

Figure 19 A drowning organization will happily extend its hand to any type of tools and methods.

5.3 Industrial impact

Some time has elapsed since some of the studies included in the research project were finalized. Below is my understanding of the changes that have been made since the studies were presented.

In a case study included in the licentiate thesis [39], the system development process at Scania was studied to identify current practice. The result of the study was presented at Scania in different groups and forums. The following three improvements were suggested and prioritized:

1. Strengthen the role of the technical career path
2. Improve knowledge transfer by documenting design know-how
3. Educate engineers in the use of structured methods

Three years later, a great deal of work has been done to strengthen the role of the technical career path. There are now more specialists within system development and appointments are highlighted to the rest of the organization. Different activities have been carried out to improve knowledge transfer. Design guidelines are more frequently updated and Wiki solutions and A3 documents are becoming a more common way of sharing knowledge. Of course, this is a never-ending story, but there have been a few steps in the right direction. To my knowledge very little has been done to improve the usage of structured methods.

Paper A presented how Value Stream Mapping could be used to improve the architecting process. Since that time there have been various efforts to use Value Stream Mapping for different processes within system development.

A paper [83] co-authored by me shows how Value Stream Mapping can be implemented in practice at Scania.

One and a half years after the case study leading to Paper B was finalized and presented at the two companies, some interesting changes have taken place. The architecting group at Scania has grown, both in the number of architects and their experience. Tool support for the architects has been significantly improved. The two separated architecting groups at Volvo Cars have merged into one, resulting in only one single architectural description. The group responsible for testing and validation at Volvo Cars is now part of the same section as the architecting group.

I do not believe the changes were made because of the results presented from those two studies, even if I hope the ideas presented due to the studies inspired the change. It does however indicate the correctness of the results or at least indicate that the results correspond to industrial reasoning.

An evaluation process using Real Options was presented and tested in Paper D. The evaluation process provides a way of valuing system designs. I believe that the method is correct and will provide improved decisions support. The problem, as with many other methods, is that the information needed is rarely available in industry. When presenting the evaluation process, I have often been given a positive response to the thoughts behind the method. Architects and people responsible for parts of the system like the idea that the increased cost of a flexible design could be argued using financial measures. To be used in industry it would need to be even more lightweight and used to guide discussion, rather than the decision itself. The greatest contribution to industry is probably a structured way of reasoning about design alternatives as options that can be valued.

Chapter 6. Conclusions and future work

The overall goal of this research has been to investigate how system architecting is performed in the automotive industry and how it can be improved by the use of Lean Thinking. This chapter presents conclusions and future work.

6.1 Summary of results

An adapted Value Stream Mapping was tested on a case study at two different companies. A comparison between the two companies shows that there are a number of value-adding methods that could be borrowed from one company to the other. It also highlighted how no formal evaluation step of architectural alternatives were made; evaluation was only mentioned as occurring in rare cases. The results of the case study have been presented at the two companies, which found them interesting but most of all inspiring for their future process improvement. The indicator that best shows that the mapping was valuable to the companies is that the presentation was requested to be held twice.

One of the case studies reveals that the studied architects see themselves as interacting much more with other stakeholders than architects in general. The results indicate how the company's different core values influence the architects when defining and maintaining the architectures over time. It also indicates the consequences that technical choices and business strategy entail for the architecting process.

This work provides a current view of the architecting process for software-intensive systems. Many of the architecting practices found in the study can be explained in the context of the different companies. A list of practices is provided for the industry reader and can be used as an inspiration or as a benchmark for improving current architecting practice.

A method has been developed to improve decision-making when making architectural changes in early phases within the automotive industry. The developed method uses real options to provide guidance when making system design decisions and, more importantly, also shows that it can be used and accepted by system engineers.

It is important to stress that success is not achieved through the use of specific tools or methods. Using the right tools and methods will often simplify or enhance the process, but having the right people with the right mindset aligned toward a common goal is much more important; developing the employees and creating an organization that never stops improving is far more important.

6.2 Future work

During this research we have seen how the balance of power between line and project has a strong influence on how work is done. This relationship would be of interest for a future study. The connection between business strategy for Cost, Quality and Time-to-Market and architecting could also be further analyzed.

Value Stream Mapping is a frequently used tool for identifying improvements in a process, but there are few industrial examples of when it has been applied to parts of the development process.

The thesis has shown that communication is a large part of architecting activities and, in order to be Lean, the communication must be effective. Kruchten [58] suggests that the productive time spent by architects can be classified into three categories of communication: internal (architecture design), inwards (input from outside world) and outwards (providing information). He argues that they should roughly have the ratio 50% internal, 25% inwards, and 25% outwards. It is very hard to measure this in practice and we have not done so in this study, but communication patterns can still be observed. Even if no extreme variation can be seen, the understanding from this study is that there is a clear difference between the companies. The architects tend to be more satisfied when the inward and outward communication is distributed evenly and where the internal work is of significant size. Future research on how communication patterns vary depending on different contexts could improve the process, aid cross-cultural-teams and enable Lean architecting.

The use of Open Innovation is growing in many different domains. The software industry is moving more and more towards different types of open solutions. Open source software enables end users to add features to the product and user involvement, such as Wikipedia, is very common, although few or no attempts are made within the automotive industry. One exception is the App-My-Ride contest arranged by Volkswagen [95]. A future research question is therefore how open innovation will enter the automotive domain and what new challenges the industry will face. The architecture would need to be adapted to accept new features being added in the aftermarket while keeping the same quality.

Working as an industrial Ph.D. student means that I have been employed as a researcher in industry and enrolled as a Ph.D. student in academia. This position means that I have experienced how industry and academia demand very different outputs from architecting research. The industry asks for best practice or success stories, while academia looks for practices proven in general. This difference could be further discussed and enable more effective knowledge transfers from academia to industry.

References

[1] "Applying the IEEE 1471-2000 Recommended Practice to a Software Integration Project."

[2] C. S. Araujo, "The utilization of product development methods- A survey of UK industry," *Journal of Engineering Design,* vol. 7, pp. 265-277, 1996.

[3] AUTOSAR,AUTOSAR specification, http://www.autosar.org/, Accessed: 2010-12-08.

[4] J. Axelsson, "Evolutionary Architecting of Embedded Automotive Product Lines: An Industrial Case Study," in *Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA*, 2009, pp. 101-110.

[5] J. Axelsson, "Towards a process maturity model for evolutionary architecting of embedded system product lines," in *Proceedings of the Fourth European Conference on Software Architecture* Copenhagen: ACM, 2010, pp. 36-42.

[6] R. Bahsoon and W. Emmerich, "ArchOptions- A Real Options-Based Model for Predicting the Stability of Software Architectures," 2003, pp. 38-43.

[7] T. Baines, H. Lightfoot, G. Williams, and R. Greenough, "State-of-the-art in lean design engineering: a literature review on white collar lean," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* vol. 220, pp. 1539-1547, 2006.

[8] P. Banerjee, *Describing, assessing and embedding flexibility in system architectures with application to wireless terrestrial networks and handset processors*: Master Thesis, Massachusetts Institute of Technology, System Design and Management Program, June 2004, 2004.

[9] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 2nd ed.: Pearson Education Inc, 2003.

[10] M. Blackenfelt, *Managing complexity by product modularisation: balancing the aspects of technology and business during the design process* vol. Doctoral Thesis. Stockholm: Royal Institute of Technology, 2001.

[11] T. Brannick and D. Coghlan, "In defense of being “native”: the case for insider academic research," *Organizational Research Methods,* vol. 10, p. 59, 2007.

[12] H. Breivold, D. Sundmark, P. Wallin, and S. Larsson, "What Does Research Say About Agile and Architecture?," in *Proceedings of The Fifth International Conference on Software Engineering Advances* Nice, France, 2010.

[13] T. R. Browning and A. Engel, *Designing Systems for Adaptability by Means of Architecture Options*. Orlando, USA: 16th Annual International Symposium Proceedings, INCOSE 2006, 2006.

[14] T. R. Browning, E. Fricke, and H. Negele, "Key concepts in modeling product development processes," *Systems Engineering,* vol. 9, pp. 104-128, 2006.

[15] L. Buchanan and A. O Connell, "A brief history of decision making," *Harvard Business Review,* vol. 84, pp. 32-41, 2006.

[16] S. Cedergren, *Evaluating Performance in Product Development - The Case of Complex Products*: Mälardalen University, School of Innovation, Design and Engineering, 2010.

[17] S. Cedergren, "Performance in Product Development - The Case of Complex Products," Västerås: Mälardalen University, 2011, p. 231.

[18] K. Clark and T. Fujimoto, *Product development performance: Strategy, organization, and management in the world auto industry*: Harvard Business Press, 1991.

[19] P. Clements and L. Northrop, *Software product lines : practices and patterns*. Boston, Mass.: Addison-Wesley, 2001.

[20] R. Cloutier, G. Muller, D. Verma, R. Nilchiani, E. Hole, and M. Bone, "The Concept of Reference Architectures," *Systems Engineering,* vol. 13, pp. 14-27, 2010.

[21] M. Conway, "How do committees invent," *Datamation,* vol. 14, pp. 28-31, 1968.

[22] J. Coplien and N. Harrison, *Organizational Patterns of Agile Software Development*: Prentice Hall PTR, 2004.

[23] J. Coplien and G. Bjørnvig, *Lean architecture: for agile software development*. New York: Wiley, 2010.

[24] I. Crnkovic and M. Larsson, "Building reliable component-based software systems," Artech House Publishers, 2002.

[25] W. Cunningham, "The WyCash portfolio management system," in *Proceedings of Conference on Object Oriented Programming Systems Languages and Applications* New York, 1992, pp. 29-30.

[26] T. Dyba and T. Dingsoyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology,* vol. 50, pp. 833-859, 2008.

[27] EAST-EEA,Definition of lanuage for automotive embedded electronic architecture, http://www.east-eea.net/start.asp, Accessed: 2007-03-29.

[28] A. Edström, "Urban på Volvo hyllar säkerheten," in *Elektroniktidningen*, 2006.

[29] P. Eeles and P. Cripps, *The process of software architecting*. Upper Saddle River, NJ: Addison-Wesley, 2010.

[30] U. Eklund, Ö. Askerdal, J. Granholm, A. Alminger, and J. Axelsson, "Experience of introducing reference architectures in the development of automotive electronic systems," in *Proceedings of the second international workshop on Software engineering for automotive systems*, 2005, pp. 1-6.

[31] R. Farenhorst and R. C. d. Boer, *Architectural knowledge management : supporting architects and auditors*. [S.l.: s.n.], 2009.

[32] R. Farenhorst, J. Hoorn, P. Lago, and H. v. Vliet, "The lonesome architect," in *Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA)*: IEEE, 2009, pp. 61-70.

[33] B. Florentz and M. Huhn, "Architecture Potential Analysis:A Closer Look inside Architecture Evaluation," *Journal of Software,* vol. 2, pp. 43-56, October 2007.

[34] R. B. Frost, "Why Does Industry Ignore Design Science?," *Journal of Engineering Design,* vol. 10, pp. 301-304, 1999.

[35] J. Fröberg, M. Åkerholm, K. Sandström, and C. Norström, "Key factors for achieving project success in integration of automotive mechatronics," *Innovations in Systems and Software Engineering,* vol. 3, pp. 141-155, 2007.

[36] T. Fujimoto, *The evolution of a manufacturing system at Toyota*: Oxford University Press, USA, 1999.

[37] R. Gemmerich, S. Semmelrodt, C. Reckord, A. Zündorf, J.Leohold, L. Brabetz, U. Schrey, and H.-G. Weil, "Ein ganzheitlicher Ansatz zur Generierung und Optimierung von Fahrzeugbordnetzen," in *12. Internationaler VDI-Kongress Elektronik im Kraftfahrzeug*, Baden-Baden, 2005.

[38] D. Greefhorst, H. Koning, and H. V. Vliet, "The many faces of architectural descriptions," *Information Systems Frontiers,* vol. 8, pp. 103-113, 2006.

[39] H. Gustavsson and J. Sterner, "An Industrial Case Study of Design Methodology and Decision Making for Automotive Electronics," in *Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference* New York, 2008.

[40] H. Gustavsson and J. Axelsson, "Evaluation of Design Options in Embedded Automotive Product Lines," in *Applied Software Product Line Engineering*, K. C. Kang, V. Sugumaran, and S. Park, Eds.: Auerbach Publication, 2009, pp. 478-495.

[41] C. Hallam, J. Muesel, and W. Flannery, "Analysis of the Toyota Production System and the Genesis of Six Sigma Programs: An Imperative for Understanding Failures in Technology Management Culture Transformation in Traditional Manufacturing Companies," in *Proceedings of Portland International Conference on Management of Engineering and Technology*, 2010.

[42] H. Heinecke, K. Schnelle, H. Fennel, J. Bortolazzi, L. Lundh, J. Leflour, J. Maté, K. Nishikawa, and T. Scharnhorst, "Automotive open system architecture-an industry-wide initiative to manage the complexity of emerging automotive e/e architectures," in *SAE Convergence*, 2004.

[43] R. Henderson and K. Clark, "Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms," *Administrative Science Quarterly,* vol. 35, pp. 9-30, 1990.

[44] D. Hoch, W. Huhn, U. Naher, and A. Zielke, *The race to master automotive embedded systems development*. Germany: McKinsey Company, Automotive and assembly sector business technology office, 2006.

[45] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran, and P. America, "Generalizing a Model of Software Architecture Design from Five Industrial Approaches," in *Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture*: IEEE Computer Society, 2005, pp. 77-88.

[46] J. C. Hull, *Options, Futures, and other derivative securities 2nd edition*. New Jersy, USA: Prentice Hall International Editions, 1993.

[47] K. Hurst, *Engineering Design Principles*: Elsevier Science & Technology Elsevier Science & Technology Books, 1999.

[48] C. O. L. Ibon Serrano Lasa, Rodolfo de Castro Vila, "An evaluation of the value stream mapping tool," *Business Process Management Journal,* vol. 14, pp. 39 - 52, 2008.

[49] IEEE-1471, "IEEE Recommended practice for architectural description of software-intensive systems," *IEEE Std 1471-2000,* 2000.

[50] J. Janhager, S. Persson, and A. Warell, "Survey on Product Development Methods, Design Competencies, and Communication in Swedish Industry," Wuhan, China, 2002.

[51] C. Karlsson and P. Åhlström, "Assessing changes towards lean production," *International Journal of Operations & Production Management,* vol. 16, pp. 24 - 41, 1996.

[52] J. Karlsson, "Marknadsdriven Produktledning-Frn kundbehov och Krav till Lönsamma Produkter," *VI report, Focal Point AB,* 2003.

[53] R. Kazman, J. Asundi, and M. Klein, *Making Architecture Design Decisions: An Economic Approach*: Carnegie Mellon Software Engineering Institute, 2002.

[54] R. Kazman, M. Klein, and P. Clements, "ATAM: Method for Architecture Evaluation," 2002.

[55] M. Kennedy, K. Harmon, and E. Minnock, *Ready, Set, Dominate: Implement Toyota's Set-Based Learning for Developing Products and Nobody Can Catch You*: Oaklea Press, 2008.

[56] P. Kettunen, "Adopting key lessons from agile manufacturing to agile software product development--A comparative study," *Technovation,* vol. 29, pp. 408-422, 2009.

[57] Y. Konno, "Enhancement of the advanced R&D cooperation between automakers and suppliers in the Japanese automobile industry," *Annals of Business Administrative Science,* vol. 6, pp. 15-34, 2008.

[58] P. Kruchten, "What do software architects really do?," *Journal of Systems and Software,* vol. 81, pp. 2413-2416, 2008.

[59] O. Larses, *Architecting and modeling automotive embedded systems*. Stockholm: Doctoral Dissertation, Royal Institute of Technology, 2005.

[60] P. Liang, A. Jansen, and P. Avgeriou, "Collaborative software architecting through knowledge sharing," in *Collaborative Software Engineering*, I. e. a. Mistrik, Ed.: Springer-Verlag, 2010.

[61] M. W. Maier and E. Rechtin, *The art of systems architecting*. Boca Raton: CRC Press, 2002.

[62] J. M. Morgan, *High performance product development: a systems approach to a lean product development process*. Ph.D. dissertation: University of Michigan, 2002.

[63] J. M. Morgan and J. K. Liker, *The Toyota product development system : integrating people, process, and technology*. New York: Productivity Press, 2006.

[64] G. Muller, "CAFCR: A Multi-view Method for Embedded Systems Architecting; Balancing Genericity and Specificity," in *Technology, Policy and Management* vol. PhD thesis: Technische Universiteit Delft, 2004.

[65] R. Muller, K. Spang, and S. Ozcan, "Cultural differences in decision making in project teams," *International Journal of Managing Projects in Business,* vol. 2, pp. 70 - 93 2009.

[66] J. Nedstam, *Strategies for management of architectural change and evolution*. Lund: Lund University, Department of Communication Systems, Faculty of Engineering, 2005.

[67] B. W. Oppenheim, E. M. Murman, and D. A. Secor, "Lean enablers for systems engineering," *Systems Engineering,* vol. February, 2010.

[68] E. Persson and H. Gustavsson, "A Framework for the Evaluation of Resource Efficiency in Automotive Embedded Systems," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, New York, USA, 2008, pp. 87-96.

[69] P. Peti, R. Obermaisser, F. Tagliabo, A. Marino, and S. Cerchio, "An Integrated Architecture for Future Car Generations," *Proceedings of the Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'05)-Volume 00,* pp. 2-13, 2005.

[70] M. Poppendieck and T. Poppendieck, *Implementing lean software development : from concept to cash*. Upper Saddle River, N.J.: Addison-Wesley, 2007.

[71] S. Pugh, *Total design : integrated methods for successful product engineering*. Wokingham: Addison-Wesley, 1990.

[72] A. E. Raymond, "The well-tempered aircraft," *Journal of Royal Aeronautical Society,* October 1951.

[73] E. Rechtin, *Systems architecting : creating and building complex systems*. Englewood Cliffs, N.J.: Prentice Hall, 1991.

[74] D. Reinertsen, *Managing the Design Factory: A Product Developers Tool Kit*: Simon &amp; Schuster Ltd, 1998.

[75] D. Reinertsen, *The Principles of Product Development Flow: Second Generation Lean Product Development*: Celeritas Publishing, 2009.

[76] C. Robson, *Real World Research-Second edition*: Blackwell Publishers Ltd., Oxford, UK, 2002.

[77] M. Rother and J. Shook, *Learning to see : value stream mapping to create value and eliminate muda*. Brookline, MA: Lean Enterprise Institute, 2003.

[78] M. Salonen and M. Perttula, *Utilization of concept selection methods – a survey of finnish industry*. Helsinki, Finland: Helsinki University of Technology, 2005.

[79] I. Sobek, Durward Kenneth, "Principles that shape product development systems : a Toyota-Chrysler comparison." vol. Ph.D. dissertation: University of Michigan, 1997.

[80] H. Soriano-Meier and P. L. Forrester, "A model for evaluating the degree of leanness of manufacturing firms," *Integrated Manufacturing Systems,* vol. 13, pp. 104 - 109, 2002.

[81] M. Steger, C. Tischer, B. Boss, A. Müller, O. Pertler, W. Stolz, and S. Ferber, "Introducing PLA at Bosch Gasoline Systems: Experiences and Practices," in *Software Product Lines*, 2004, pp. 34-50.

[82] J. Sutherland and K. Schwaber, "The Scrum Papers: Nuts, Bolts, and Origins of an Agile Process," 2008.

[83] J. Tingström, H. Gustavsson, and P. Palmér, "Implementing Value Stream Mapping - VSM in a R&D organisation," in *Proceedings of NordDesign2010 - International Conference on Methods and Tools for Product and Production Development* Goteborg, 2010.

[84] C. Tischer, A. Muller, M. Ketterer, and L. Geyer, "Why does it take that long? Establishing Product Lines in the Automotive Domain," in *11th International Software Product Line Conference*, Kyoto, Japan, 2007, pp. 269-274.

[85] K. T. Ulrich and S. D. Eppinger, *Product design and development*: McGraw-Hill New York, 1995.

[86] H. Unphon and Y. Dittrich, "Organisation matters: how the organisation of software development influences the development of product line architecture," in *Proceedings of International Conference on Software Engineering*, Innsbruck, Austria, 2008, pp. 178-183.

[87] H. Unphon and Y. Dittrich, "Software architecture awareness in long-term software product evolution," *Journal of Systems and Software,* 2010.

[88] P. Wallin, J. Froberg, and J. Axelsson, "Making Decisions in Integration of Automotive Software and Electronics: A method based on ATAM and AHP," 2007, p. 5.

[89] T. van Beek, M. Erden, and T. Tomiyama, "Modular design of mechatronic systems with function modeling," *Mechatronics,* 2010.

[90] F. van der Linden, J. Bosch, E. Kamsties, K. Känsälä, and H. Obbink, "Software Product Family Evaluation," in *Third International Software Product Lines Conference*. vol. Volume 3154 Boston: Springer Berlin, 2004, pp. 110-129.

[91] A. Ward, *The Lean Development Skills Book*. Ann Arbor: Ward Synthesis, 2002.

[92] A. C. Ward, *Lean product and process development*. Cambridge, Mass.: Lean Enterprise Institute, 2007.

[93] D. Weiss and C. Lai, *Software product-line engineering: a family-based software development process*: Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1999.

[94] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Introduction to Experimentation in Software Engineering*: Kluwer Academic Publishers, 2000.

[95] Volkswagen,App my Ride, www.app-my-ride.com, Accessed: 2010-12-08.

[96] J. P. Womack, D. T. Jones, and D. Roos, *The machine that changed the world*. New York: Rawson Associates, 1990.

[97] C. von Wangenheim, J. Hauck, C. Salviano, and A. von Wangenheim, "Systematic Literature Review of Software Process Capability/Maturity Models," in *Proceedings of International Conference on Software Process. Improvement And Capability dEtermination*, 2010.

[98] Y. Yamamoto, *Kaikaku in production*. Västerås: Mälardalen University, 2010.

[99] R. Yin, *Case Study Research : Design and Methods (Applied Social Research Methods)*: SAGE Publications, 2002.

[100] W. Zientz, "Electronic systems for commercial vehicles," in *ATZ AutoTechnology*. vol. 5, 2007, pp. 40-43.

Paper A

Improving the system architecting process through the use of Lean tools

Håkan Gustavsson
Mälardalen University
School of Innovation, Design and Engineering

Jakob Axelsson
Mälardalen University
School of Innovation, Design and Engineering

Abstract

The impact of embedded systems within the automotive industry has grown very rapidly and is today influencing most part of the product development process. This technological change puts high demands on the development process in order for the company to stay competitive.

The architecting process is performed during the early phases of the development process when uncertainty is very high. The architecting process will not create immediate value to the end customer, but rather create the architecture on which value in terms of product features can be developed. The architecture will enable value creation when working properly or, in the worst case, prevent value creation.

Lean is a product development philosophy that aims at creating value for the end customer. A Lean tool used to improve the value creation within a process is Value Stream Mapping (VSM). VSM has in this work been adapted and evaluated to analyze and identify improvements of the architecting process within embedded systems development. In this paper we present practical experiences from using this adapted VSM. The evaluation was conducted through interviews at two automotive manufacturers. VSM is shown to be a valuable tool to identify waste and thereby improve the architecting process.

I. Introduction

The product development process is often depicted as a straight forward process starting with an idea and ending with a validated product. The reality is often not as stringent [3], and iterations and rework is part of most product development processes. There are methods such as real options [4] available to evaluate different technical design decisions. To make the right technical decisions is very important, but to stay competitive this most be done in the right way. According to Ward [20], 60% of the time invested within product development is waste.

To stay competitive in the automotive industry vehicle manufacturers are forced to release new models more often. At the same time the product portfolio must be further diversified in order to satisfy individual customer demands. The shorter development cycle and increased number of concurrent models brings an increased need for transfer of design knowledge. In this study a car manufacturer (Volvo Cars) is compared with a manufacture of commercial vehicles (Scania). A commercial vehicle must manage to run 300 000 km per year and breakdowns do not just influence the driver, but also the delivery time of the goods it carries. Commercial vehicles have a lot in common with passenger cars, much of the functionality are found in both segments. The passenger car industry has traditionally been adopting new technology earlier. This can be explained by the different needs of the customer.

Today most innovations made within the automotive domain are driven by electronics. Future functions that enable vehicles to communicate with not just other vehicles, but also the infrastructure [2]. Those future demands are increasing the complexity and the boundaries of the automotive electronic and electrical (E/E) system. The architecture of the E/E system has a large impact on how expensive or difficult those changes will be to implement. The architecture will enable value creation when working properly or, in the worst case, prevent value creation. The process of architecting the E/E system is therefore an important process to improve.

In our work, architecting is viewed as the process of shaping the architecture to meet customer demand by balancing requirements, guiding principles and product vision. As we see the architecting process is central to and dependent on many factors within the organization. In order to improve the process the involved activities would need examining. With this in mind the following research question is studied in this paper:

How can the system architecting process be mapped in order to identify improvements?

A hypothesis to be tested is whether VSM is a suitable method.

The literature review explains the concept of lean and how it relates to system architecting. VSM is then reviewed in Section III followed by a description of the adapted method for performing VSM on the system architecting process. This method is then utilized on a case study described in Section V. The results of the case study are then discussed followed by a presentation of future work to be done.

II. Method and Methodology

The literature on Lean and Value Stream Mapping (VSM) has been studied to understand the concepts. This knowledge has been used in the process of defining the case study. After the case study was constructed, it was tested on one person at each company who previously has been employed as system architect. The chosen format of the interview was semi-structured and the answers were recorded by a person with deep knowledge of the architecting process. A semi-structured interview has predetermined questions, but the order can be modified based upon the interviewer's perception of what seems most appropriate. Question wording can be changed and explanations given [15]. The interviews at both companies followed the same template and the answers given were then used to describe the process.

III. Literature review

Lean

Lean is a practice that considers the usage of resources for any goal other than the creation of value for the end customer to be wasteful, and thus a target for elimination. Working from the perspective of the customer, who consumes a product or service, value is defined as any action or process that an internal or external customer would be willing to pay for. The concept of Lean production was defined in the literature by Womack et al. [21], but derives from the working methods developed by Toyota in the 1950s.

Lean methods focus on increasing customer value and on the people who add value. A Lean-based company encourages its employees to perform continuous improvement and to learn. This is done by cross-functional and parallel work and a high degree of standardization in order to optimize across organizations. The concept of Lean production has today moved from manufacturing into various sectors, such as maintenance, purchasing, logistics, and to product development which is the topic of this paper. Lean production is achieved by careful planning of a production line in order to optimize the production flow to meet customer needs. Each assembly station is arranged to minimize unnecessary motion and transportation of material. Each assembly station is assigned defined tasks to be finalized on a specific time in order to achieve a balanced flow throughout the production-line. A balanced flow means that the results are delivered on-time without waiting or over-production.

An important starting point of lean product development is to view the product development as a process, and like any other process there are repeated cycles of activity [10]. This is important even though the resulting artifact is per se novel to some degree. From a process perspective, there are many activities that are shared between different development projects. By eliminating the waste in a process, an increased flow is achieved, thus new products can be brought to the market at a higher pace.

There are two main differences between manufacturing and the early phases of product development. The flow does not consist of materials but more often information and knowledge in different shapes. Different organizational and geographical locations of the stakeholders influence how this knowledge is shared. The process does not consist of one flow, but instead iterations are often made and different concepts are developed in parallel.

Allen Ward [20] claims that 20% of the time spent in product development is value adding time. Nonvalue-creating time such as administration work occupies 20% and the remaining time is waste. This fact would suggest that optimization is possible if we identify the wasteful activities. It is common to define seven types of waste [10] and value stream mapping is one method to identify the waste within system architecting. According to Allen Ward [20] the most frequent waste in development is waste of knowledge. He divides knowledge waste in three categories: scatter, hand-off, and wishful thinking. Scatter is described as actions that disrupt the flow of knowledge. This disruption can be due to communication barriers and the use of inappropriate tools. Example of knowledge waste created by hand-offs is to move people around rather than assigning them from the beginning to the end. Waste due to wishful thinking is for instance to test according to specification rather than to test to learn about the limits of the product.

In the literature, there is little work on how Lean can be applied to the process of developing software-intensive systems. Poppendieck and Poppendieck [13] present how Lean can be applied to the software development process. In their work, typical wastes to be found are hand-offs between individuals, switching between tasks and adding extra features. Value Stream Mapping is presented as one way to find waste.

IV. Value Stream Mapping

There are many different techniques available for process modeling, but Value Stream Mapping (VSM) differentiates in focusing on value creation. Value Stream Mapping (VSM) was initially a tool for improving the manufacturing process [16] and has shown to be effective within manufacturing [7]. The method is today also used within many other disciplines. The process includes four steps which are described in the next sections.

Value Stream Scope

The purpose of scoping is to determine what process (value stream) is to be improved and to create a common view of the process to be analyzed. This means understanding what processes are included and where the process starts and ends. It should also be decided upon who will perform the VSM and who will support the event, including management. The output of the scoping is therefore an input-output view (Figure 1) of the process and its control parameters, but also a working plan [5]. Control parameters could be a common strategy or business goals. Enablers are resources consumed by the process such as available people and tools.

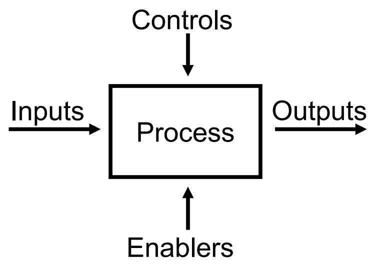

Figure 1 A input-output view of a process.

Current State

The aim of this step is to understand how things currently operate. This is done through a walk-through of the entire process from beginning to end, usually in a workshop manner. The demands of the internal and external customers must be identified. The flow of material and information is then mapped, identifying each process time and lead time.

To illustrate how this is done, a fictive example is presented in Figure 2. The sub process of updating a communication interface in a document and a database is mapped with the recommended symbols [8]. Figures of the process are given through a walkthrough of the process. The process time is the required time it takes to complete a specific task when working without interrupts. The task of creating an interface description takes 120 minutes from start to finish. The number of people and resources normally available for a task are given after the symbol in the middle. In this example, we find out that the dedicated employees normally have 30% time available for creating interface description.

It then normally takes half a day from the handover until the work to update the database is started, which is indicated below in the IN process box. The task to update the interface database is then started, taking an average of 30 minutes to perform with one person available at 50%.

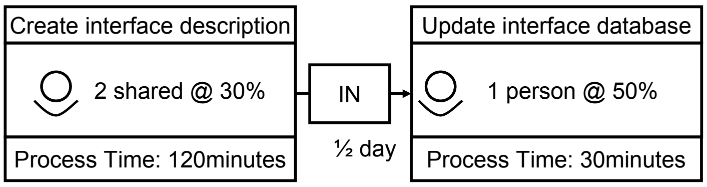

Figure 2 The subprocess of updating a communication interface in a document and a database.

Future State

The purpose of this step is to improve the process, i.e., to design a lean flow. This is done by analyzing the process with regards to the Lean principles. There are a number of questions that can be asked to find those improvements [8]. What does the customer really want? Which steps create value and which steps are waste? How can we design a flow of work with fewer interruptions? Using this set of question some additional issues will arise in our example: Are the interface description what the customer really wants or are some parts not necessary (e.g. waste)? Does the information need to be added to two different sources or would the database be enough? Can the task be done by the same person and thereby reduce the lead time?

With the guidance of those questions a future state of the example can be drawn. If the document is not needed and the task can be done by the same person the following future state can be drawn. The lead time is reduced by half a day and the process time with 30 minutes (Figure 3).

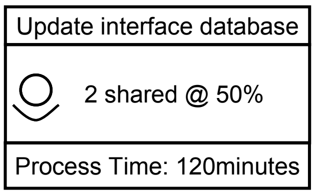

Figure 3 The result of the future state.

Work plan and implementation

This last task is the final goal of the VSM, namely to ensure that the improvements are implemented. It is done by describing the specific improvements that are chosen to be implemented from the previous step. A work plan is made showing what will be done by whom at what time. The work plan is used to follow-up that the tasks are being performed. The planned changes must be communicated to everybody involved in the process. To make the necessary changes it is crucial to have management attention. Summarizing what is learned in the VSM event is done in order to ensure that knowledge is carried to the next time (lessons learned).

V. VSM for System Architecting

In this section we will present an adapted VSM for a system architecting process of a software intensive system. The purpose of creating an adapted VSM is to enable comparison between different organizations and thereby improve knowledge transfer. In order to make this adaptation, a literature review of the architecting process has been carried out. The authors' previous practical experiences as system architects has also aided the work.

Value Stream Scope

The architecting processes is influenced by many different factors [18]. To be able to understand different architecting processes one must first understand the surrounding circumstances. The attributes that are important to gather in order to understand the context were derived from the literature [1, 9, 14]. The attributes in table 1 are derived to make a comparison possible of the architecting process and grouped according to the BAPO-model [18].

Table 1 Attributes describing the context of the system architecting process, with examples given in parenthesis.

Business
Number of products produced per year
Number of product variants
Procurement strategy (make or buy)
Lifetime of the system in number of years
Organization
Geographical distribution of the R&D organization
Number of employees in the R&D organization
Number of employees of the system development organization
Type of organization (matrix, project)
Balance of power (line, project)
Organizational location of architects (co-located, separated)
Number of system architects
Architectural power (line, project, architects)
Process
Development process (Stage-gate)
R&D Organization (national, one location)
Guiding principles
Culture (consensus)
Methods in use
Architecture
Level of SW/HW architecture
Type of architecture (product-line, single product)
Principles or architectural rules
Architectural lifecycle (continuous, revolutionary)
Number of parallel architectures

When those attributes are known and understood a comparison can be made and the right conclusions can be drawn. The architecting process (Figure 4) starts when a change request reaches the architecting team and ends when a solution is presented and decided upon. The input of a legacy architecture and customer requirements are transformed into a revised architecture, which adds customer value and knowledge to the organization. The process is controlled by business attributes and enabled by the organizational attributes. A generic input-output view of the system architecting process can be seen in Figure 4. When the attributes are known the value stream scope is also clearly defined. In our case study, figures about the different companies were gathered from financial reports and through a company contact. Less exact attributes such as "balance of power" were obtained after analyzing the interview data.

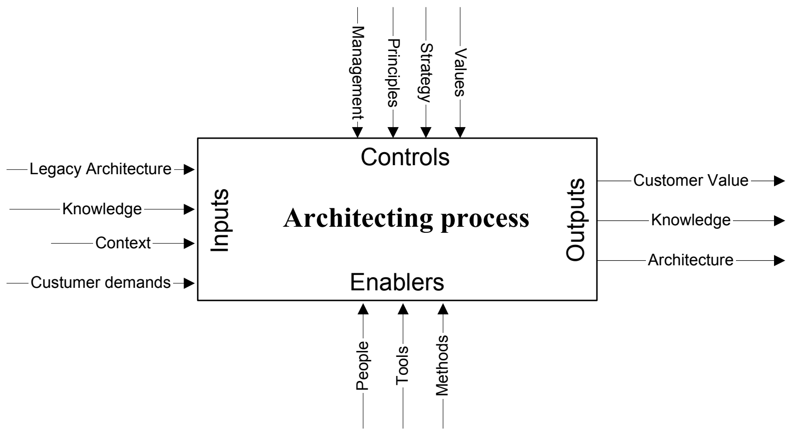

Figure 4 The attributes affecting the architecting process.

Current State Drawing

Depending on the process maturity of the organization, estimations of lead and process time will be hard to find, but might be interesting in a second VSM iteration. Therefore a first VSM is chosen to be made lightweight. The architecting process is a support process that usually aids an overall development process. The current state was obtained through semi-structured interviews at two companies. Through the answers to the interview questions the system architecting process of the two organizations were analyzed. The differences in the two organizations ways of working were then mapped to a reference process (Figure 5) derived from the best practice according to the literature [6, 9, 11, 14]. Waste and deviations from the reference process were then documented. Available performance measurements such as throughput, customer satisfaction or first pass yield were also taken into account. The output of this step is an image of the created value stream map.

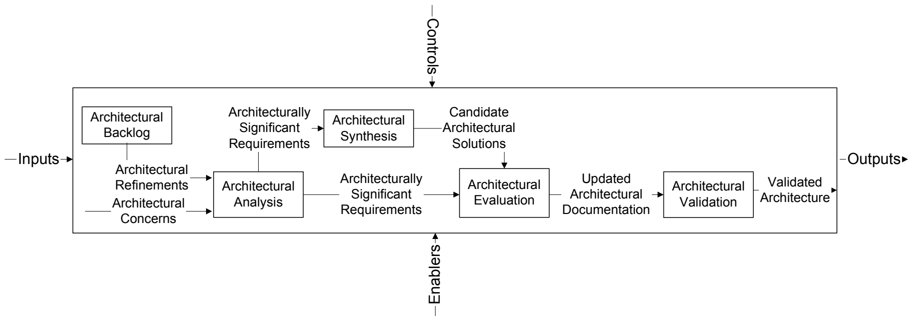

Figure 5 Reference architecting process.

A special difficulty when analyzing the system architecting process is that much of the value created, and waste removed, is actually seen in other subprocesses of the product development process. The architecture organizes the work of many activities, and a good architecture provides clear and simple interfaces between subsystems, making the system development for these parts more efficient. Finding the best balance between the amount of architecting vs. system development is one of the most difficult parts in product development management.

Future State Drawing

Most of the customers of the architecting process are internal and the customer value is difficult to calculate. The questions applied in a traditional VSM (section IV) might therefore be hard to answer but will none the less be important. To make the results comparable the different categories of waste found should be documented. The architecting process is supportive and inputs are given at various times, which make rework hard to avoid. Waiting until all inputs are available could stall the overall development process. The difficult task in this step is therefore how to cope with this uncertainty and maximize the value creating activities.

The future state of the process is achieved in two steps. The first is to find countermeasures to remove non-value-adding activities found in previous step. Those can be as simple as to stop producing a document that is not used, but in most cases it will be more complicated i.e. changing the way architects interact with other stakeholders. The second step is to benchmark the current state, in our case the other company. For future users of the method the two case-companies documented in this paper can be used as comparison. The output of this step is an improved process.

Work plan and implementation

A work plan is made showing what improvements could be done. To ensure success of the work the suggested improvements should be prioritized. It is important not to overload the organization with changes. Improvements leading to fast return on invested time are a way to encourage further work on improvements.

VI. Case study

The case study was conducted at two different automotive OEMs using semi structured interviews. In the study the researchers interviewed all architects available and willing to participate, which resulted in more than half of the persons working as architects at each company were interviewed, 4 at Scania and 5 at Volvo Cars. In addition to this the managers for the architecting group were interviewed at both companies, totalling the number of interviews to 11. Of the 11 respondents 2 were women. The interview started with some introductory questions to get some background about the respondent followed by a set of predefined questions. To ensure participation the length of the interview were kept to one hour.

Scope

The two companies are similar in both being automotive OEMs in the premium segment and both being located in Sweden, but different in aspects concerning organization, business and architecture. A clear difference is the types of products being produced, cars and commercial vehicles.

The main differences in business attributes are the production volume and procurement strategy. Volvo Cars buys a much larger part of the EE system and is also producing a much higher number of vehicles per year. Even if both cost and quality are important for both companies, Volvo Cars has a stronger focus on cost and quality is found more important at Scania.

Scania has chosen to have one common architecture which is continuous evolving and Volvo Cars has several parallel architectures. The two matrix organizations are very similar in size and their R&D department is both located in one single location. The biggest difference is found in the balance of power between the line and project. At Volvo Cars the main power is in the project organization and at Scania the line organization has the main power.

The process is managed differently, Volvo Cars uses traditional methods for communication and process follow-up and Scania uses visual planning and

Obeya rooms [10]. An Obeya room is a place where cross functional knowledge is visualized and is used to show progress and to get a overall view. Respondents at both companies think that the decision making is slow. The architects at Volvo Cars and Scania have similar experience within the field, but the architects at Volvo Cars have been within the company significantly longer.

The inputs to the process were different in how changes affecting the architecture were entering the process. Scania has a well defined process into which all changes are entered. Volvo Cars has a similar process, but the process is not as settled and changes are therefore sometimes stumbled upon.

Current state

Both companies mapped easily to the reference process, with one exception. No formal evaluation step was made; evaluation was only mentioned to be made in rare cases. It is important to note that the process is not as sequential as it might appear in Figure 5, iterations are made between all steps and especially between the analysis and the synthesis. Those iterations lead to waste in terms of waiting for information, which delays the process in both companies.

The tools used for documenting the architecture at Scania are not integrated which leads to waste when the same information needs to be entered more than once. Definitions of important concepts such as architecture are not defined at Scania, and this is waste caused by a communication barrier. The shorter employment time of the architects at Scania could also cause waste because of lack of company knowledge. The architects at Volvo Cars are assigned to a single architecture and knowledge sharing between them is therefore limited.

Future state

The decision making process in Sweden is known to be based on consensus decisions which leads to more meetings and communication [12] than areas with other culture. More meetings are not necessarily waste as long as knowledge is shared and the right people are attending well prepared meetings. It is important though to ensure the meetings to be effective. The frequent iterations are often due to loss of information in previous development steps. This waste could probably be eliminated through improved knowledge transfer of design rationale. Both Volvo Cars and Scania could document design rationale using the A3-technique [17]. A3 is a practical knowledge sharing mechanism using one single page to report e.g. decision-making or problem-solving.

A comparison between the two companies shows that there are a number of value-adding methods that could be borrowed. Scania is today using workshops as a method during the synthesis, and this could be one way to improve knowledge sharing at Volvo Cars. A similar tool chain as the one used at Volvo Cars could eliminate the waste caused by multiple entries of data at Scania. Scania uses visual planning [10] to keep track of the progression of tasks and workload of the architects, and this could improve how the backlog is handled at Volvo Cars. Working in pairs and in different areas increases knowledge sharing at Scania, this could also be tested at Volvo Cars. This type of knowledge sharing also provides a more flexible staff that can help out and reduce workload of other architects. Common understanding of different important concepts in the architecture should be improved at Scania to make the knowledge sharing more effective. Design reviews are made regularly at both companies and provide value as a knowledge sharing activity. Scania also uses feedback from the test department to validate the architecture; this can be improved at Volvo Cars.

Work plan and implementation

The suggested work plan was to first of all present the result for the two companies and to let them prioritize the suggested improvements. As this case study was made on a real process with real people it will take some time before a possible change take place. This is therefore not included in this work.

VII. Discussion and future work

In this paper the theory of Lean and VSM has been explained and a adapted VSM has been presented. The adapted VSM was then tested on a case study through 11 interviews at two different companies. The result of the case study has been presented at the two companies, who found them interesting, but most of all inspiring for their future process improvement. The indicator best showing that the mapping was valuable to the companies is that the presentation was asked to be held twice.

During the interviews it was important to ask and understand the previous experience of the respondents. Depending on their background respondents will have different perspectives. The answers of the respondents at each company were surprisingly similar. The author’s knowledge of the field was found important to make the interviews effective and to understand the acronyms and technical terms used. Improvements before a future case study will be to reduce the number of questions in the interview template that were found redundant.

In future work the interviews will be further explained and the case study expanded to include more companies. This will provide academia with knowledge of how architecting is performed. The industry can use the methods found for comparison and inspiration of process improvements.

Acknowledgment

We would like to thank the architects and managers at the two companies for their interest and cooperation. This work has been financially supported by the Knowledge Foundation and the Swedish Agency for Innovation Systems (VINNOVA) as part of the FFI program.

References

[1] J. Axelsson, J. Fröberg, H. Hansson, C. Norström, K. Sandström, and B. Villing, "A Comparative Case Study of Distributed Network Architectures for Different Automotive Applications," in The Industrial Information Technology Handbook, R. Zurawski, Ed. Boca Raton, USA: CRC Press, 2005, pp. 57-1 to 57-20.

[2] M. Broy, I. H. Kruger, A. Pretschner, and C. Salzmann, "Engineering Automotive Software," Proceedings of the IEEE, vol. 95, pp. 356-373, 2007.

[3] H. Gustavsson and J. Sterner, "An Industrial Case Study of Design Methodology and Decision Making for Automotive Electronics," in Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference New York, 2008.

[4] H. Gustavsson and J. Axelsson, "Evaluation of Design Options in Embedded Automotive Product Lines," in Applied Software Product Line Engineering, K. C. Kang, V. Sugumaran, and S. Park, Eds.: Auerbach Publication, 2009, pp. 478-495.

[5] C. Haskins, "Systems engineering handbook," 3rd ed Seattle: INCOSE, 2007.

[6] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran, and P. America, "Generalizing a Model of Software Architecture Design from Five Industrial Approaches," in Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture: IEEE Computer Society, 2005, pp. 77-88.

[7] C. O. L. Ibon Serrano Lasa, Rodolfo de Castro Vila, "An evaluation of the value stream mapping tool," Business Process Management Journal, vol. 14, pp. 39 - 52, 2008.

[8] D. Locher, Value Stream Mapping for Lean Development: A How-To Guide for Streamlining Time to Market Taylor & Francis 2008.

[9] M. W. Maier and E. Rechtin, The art of systems architecting. Boca Raton: CRC Press, 2002.

[10] J. M. Morgan and J. K. Liker, The Toyota product development system : integrating people, process, and technology. New York: Productivity Press, 2006.

[11] G. Muller, "CAFCR: A Multi-view Method for Embedded Systems Architecting; Balancing Genericity and Specificity," in Technology, Policy and Management vol. PhD thesis: Technische Universiteit Delft, 2004.

[12] R. Muller, K. Spang, and S. Ozcan, "Cultural differences in decision making in project teams," International Journal of Managing Projects in Business, vol. 2, pp. 70 - 93 2009.

[13] M. Poppendieck and T. Poppendieck, Implementing lean software development : from concept to cash. Upper Saddle River, N.J.: Addison-Wesley, 2007.

[14] E. Rechtin, Systems architecting : creating and building complex systems. Englewood Cliffs, N.J.: Prentice Hall, 1991.

[15] C. Robson, Real World Research-Second edition: Blackwell Publishers Ltd., Oxford, UK, 2002.

[16] M. Rother and J. Shook, Learning to see : value stream mapping to create value and eliminate muda. Brookline, MA: Lean Enterprise Institute, 2003.

[17] D. K. Sobek and A. Smalley, Understanding A3 Thinking. New York: Taylor & Francis, 2008.

[18] F. van der Linden, J. Bosch, E. Kamsties, K. Känsälä, and H. Obbink, "Software Product Family Evaluation," in Third International Software Product Lines Conference. vol. Volume 3154 Boston: Springer Berlin, 2004, pp. 110-129.

[19] A. Ward, The Lean Development Skills Book. Ann Arbor: Ward Synthesis, 2002.

[20] A. C. Ward, Lean product and process development. Cambridge, Mass.: Lean Enterprise Institute, 2007.

[21] J. P. Womack, D. T. Jones, and D. Roos, The machine that changed the world. New York: Rawson Associates, 1990.

Paper B

Architecting Automotive Product Lines: Industrial Practice

Håkan Gustavsson
Scania
Södertälje, Sweden

Ulrik Eklund
Volvo Car Corporation
Göteborg, Sweden

Abstract

This paper presents an in-depth view of how architects work with maintaining product line architectures in the automotive industry. The study has been performed at two internationally well-known companies, one car manufacture and one commercial vehicle manufacture. The results are based on 12 interviews with architects performed at the two companies. The study shows what effect differences such as a strong line organization or a strong project organization has on the architecting process. It also shows what consequence technical choices and business strategy have on the architecting process. Despite the differences the results are surprisingly similar with respect to the process of managing architectural changes as well as the information the architects maintain and update, especially in the light that the companies have had no direct cooperation.

Keywords: Architecting, Process, Case study, Automotive industry

1 Introduction

Software and electronics are today an important part in the development of automotive products. Experts [1] estimate that 80 percent of all future automotive innovations will be driven by electronics. Scania [2] claims that electronics in trucks and buses makes up 10-15 percent of the value and is increasing. Volvo Cars [3] estimates the value of electronics of a high-end car to 30 percent.

Architectural changes of distributed embedded systems are either evolutionary or revolutionary [4], and the architecture plays a vital role to the success of the product line. The main purpose of this paper is to understand how architecting is performed to keep up with evolutionary changes. This is summarized in the research question to be answered: What tasks are performed in the process of architecting automotive embedded systems?

Decisions in the development process [5] and within the architecting process [6] has been previously studied. Dobrica and Niemela [7] makes a comparison of eight different available software architecture analysis methods. Experience reports of introducing product lines in the automotive domain for the first time has been done previously [8] as well as showing the benefits of the introduction [9]. In a survey of 279 IT architects in the Netherlands Farenhorst et al. [10] concludes that architects are lonesome decision makers; not very willing to share architectural knowledge, but eager to consume.

This paper presents a comparison of how architects at two different companies work with maintaining existing product lines. The case study has been performed at two automotive companies, the truck and bus manufacturer Scania and the car manufacture Volvo Cars. In the next section a brief presentation is given of a general automotive electrical system. In Sec. 3 the method used in the study is presented. An outline of the case study is given in Sec. 4 followed by the results in Sec. 5. Finally we discuss the findings from our work.

2 Background

2.1 The Systems and Their Architecture

The electrical system in both cars and trucks/buses are an embedded software system consisting of 30-70 different Electronic Control Units (ECUs), each with a microprocessor executing in the order of 1 MByte compiled code[2]. These ECUs control the behavior of virtually all electrical

[2] A few safety-critical ECUs have two microprocessors for redundancy or internal monitoring.

functions, from power windows to valve timing of the engine. The in-vehicle software share a number of characteristics common to the automotive domain (see e.g. [11], [12] and [13] for further elaboration):

- A large number of vehicle models with varying feature content and configurations which must be supported by the software
- Highly distributed real-time system
- Distributed development at vehicle manufacturers and suppliers
- Low product cost margins
- Stringent dependability requirements

This combination of characteristics, together with a steady growth of features realized by electronics and software, makes the electrical system in a vehicle a highly complex software system.

Almost all ECUs have a number of sensors and actuators connected to them depending on purpose and location, and these can be shared among distributed functions. Most ECUs are reprogrammable, i.e. has flash memory and not ROM, which allows programming both in the manufacturing plant as well as at dealers and workshops after delivery to the end-user. The layout of which ECUs are connected to which bus and what ECUs are acting as communication gateways between the buses is the network topology of a vehicle, of which Fig. 1 is a representative example.

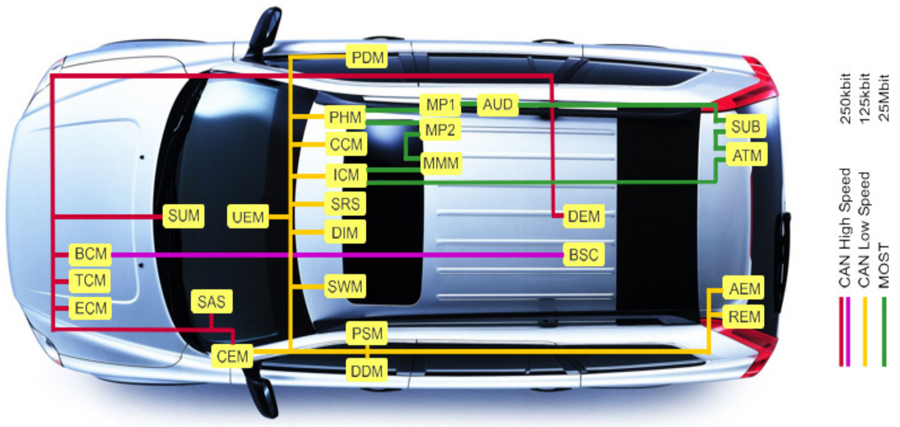

Fig. 1 The network topology of a Volvo XC90. The ECUs connected to CAN and MOST and the main multiplexed networks are seen in their approximate physical location. See [16] for a more in-depth description of the network topology of both Scania and Volvo vehicles.

The interface between the software application on each ECU is in a Scania vehicle defined by the J1939 standard [14], which is very detailed in what information is. Volvo Cars uses a proprietary solution for the multiplexed communication which allows a high degree of flexibility in defining and maintaining interfaces on the buses [15]. Much of the activities regarding the logical architecture at both companies are focused on these interfaces.

2.2 Related Work

Almost all of the cases we found regarding product lines focused either on the prerequisites for a successful product line approach or the change management of an organization adapting a product line where it previous not had one. Some examples from the automotive industry are [17], [8] and [18].

Buhrdorf et al. [19] reports about the transition Salion did to a product line with a reactive approach where the necessary variations was not explored when introducing the product line, but rather handled in what they call the "steady state". The architecting work in this paper is also reactive with the same definition, since it is about updating the systems and their architectures to comply with prerequisites not known when the architecture was first designed.

3 Methodology

The data used in this study is based on interviews with the persons most involved in the activities of maintaining architectures, i.e. the architects themselves. Neither Scania nor Volvo makes a distinction of the roles for system and software architects. All architects available and willing to participate were interviewed, which resulted in more than half of the architects at each company participating, 4 at Scania and 5 at Volvo Cars. In addition to this the managers for the architecture groups were interviewed at both companies, totaling the number of interviews to 11. Of the 11 respondents 2 were women.

The interviews were performed by the two authors, which are native to Scania and Volvo Cars respectively (see [20] for the definition of "native" in this context). One lead the interview while the other took extensive notes, which was later edited for spelling and grammar. The respondents had the possibility to read and comment the notes from their respective interview to correct any misunderstandings, purse errors or other mistakes in the recordings. This was done before the analysis took place.

The interviews were semi-structured with open-ended questions. The researchers paid special attention to not use any terminology that had special or different meanings at the two companies to avoid the respondents perceive the same question differently depending in which company they were working. After the interview was constructed, it was tested on one person at each company who had worked as a system architect to evaluate the relevance.

The interview questions were defined in English and then translated to the native language of the interviewers and respondents, Swedish, for a more natural and fluent setting. Whenever a quote from the interviews is presented in the article the translation to English was done post mortem.

The interview started with some introductory questions to get some background about the respondent, like education, professional experience of embedded systems, time employed and a general idea of how they would define architecture. The majority of each interview was based on a set of questions directed at exploring the respondent's view of their work with the architecture. The set of questions were aimed to cover all stages of an architecting process from [21] to make sure no vital information was missed. All 11 interviews progressed in essentially the same order.

3.1 Analysis Procedure

The analysis was made by the two researchers jointly looking for common themes based on the interview questions. Also answers relating to these themes given in other questions were including in this analysis. The themes were also analyzed if they showed a close similarity between the two companies or significant differences. The two authors used their insider knowledge about respective organization and products in making the analysis and to enrich the conclusions made.

4 The Case Study

The main objective of this study was to get the richest insight possible into how architects maintain an existing architecture in practice. The selection of the two automotive companies was made for three reasons. The first is that the authors already had inside access to the subjects and the support of middle management to perform this and similar studies. Second the two companies are similar enough for a comparison to be manageable, such as each company having a product line architecture approach, but still different enough for the interviews not to be a duplicate. The third, and not least, reason is the possibility for the authors to use their knowledge as insiders to augment the analysis of the data to provide an even richer insight into the two cases.

4.1 Context

Both companies studied are situated in Sweden and share characteristics common among Swedish engineering industries such as; solid knowledge about the product among the developers, putting value on personal networks, and similar educational and demographic background in the development departments. The overall product development process at both companies follows a traditional stage-gate model. An important difference is the balance of power; Scania has a stronger line organization [22] while at Volvo Cars the project organization is stronger.

All participants had a similar educational background with an engineering master degree from a Swedish university. They had worked with embedded systems between 5 and 25 years. They also had similar experience working as architects, with a majority being an architect for 4-6 years. The main difference was that the architects at Volvo Cars had on average worked twice as long in the company, compared to Scania.

Scania is one of the world's leading manufactures of heavy commercial vehicles selling on a global market with a solid reputation of designing and producing vehicles with the core values of “Customer first”, “Respect for the individual” and “Quality”. During 2008[3] Scania produced 66,516 trucks and 7,277 buses. Scania is a public company with Volkswagen AG as the largest stockholder. The development of all critical parts of the product, such as engine, transmission, cabs and chassis are centralized to the research and development centre in Södertälje, Sweden.

Volvo Car Corporation is a manufacturer of premium cars with core values[4] of “safety”, “environment” and “quality”. Volvo Cars produced 374,297 vehicles in 2008[5]. Volvo Cars is a subsidiary company to Ford Motor Company (as of 2010 February 23), sharing technical solutions with other brands within FMC.

---

[3] http://www.scania.com/scania-group/scania-in-brief/key-figures/

[4] http://www.volvocars.com/intl/top/about/values/pages/default.aspx

[5]http://www.volvocars.com/SiteCollectionDocuments/TopNavigation/About/Corporate/Volv Sustainability/VolvoCars_report_2008_ENG.pdf

4.2 The Scania Product Line

Scania has a tradition of working with a modular product design since the early 1960's. The modular system has claimed to be the main reason why the company stayed profitable every year since 1934 [23]. The internal training program teaches the three basic corporate principles of modular thinking [24]:

1. Standardized interfaces between components
2. Well-adjusted interval steps between performance classes
3. Same customer-need pattern = same solution

These principles are today also applied on the electrical and electronic system, besides the traditional mechanical parts. Scania does all design work towards the product line, there is no work done towards a specific product model. A project at Scania is an addition or update to one or more modules towards a specific time when it goes into production, and there is no difference if the update is purely mechanical or includes software as well, the product line approach is identical [24]. The Scania product line uses the same architecture, as well as components, for all of its three product categories; trucks, buses and engines, seen in Fig. 2. Every sold product is customer ordered and unique which is made possible through the modular system.

The software adaptation of each product is made during production. This is done by extracting a configuration file from the manufacturing product specification, which is then downloaded onto the unique product.

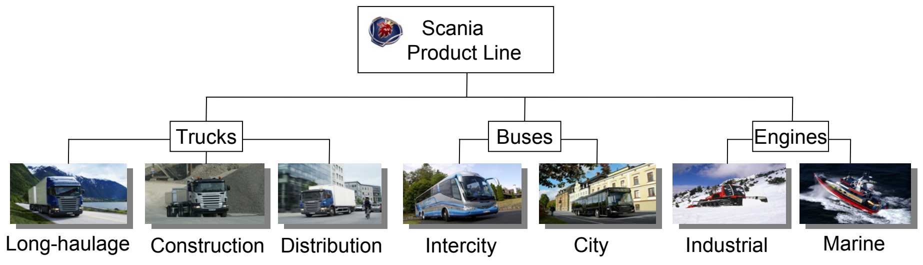

Fig. 2 The product line at Scania and the different products built on it.

4.3 The Volvo Cars Product Lines

Presently Volvo Cars maintains 3 electrical architectures for the 3 platforms in production. All vehicles in a platform are said to share the same architecture, which includes the software as well as the hardware it is executing on.

Volvo does most engineering work towards a new vehicle model, or model year, but with the intention that a solution should later be used for other vehicles on the same platform. In contrast to Scania, Volvo defines the product requirements for the individual car models and not the product line as a whole. The development of the architecture and sub-system solutions are shared between the platform and the individual products, an approach driven by the developers at the Electrical and Electronic Systems Engineering department themselves rather than a company-wide business strategy.

All vehicles produced are made to order. With the possibility for the customer to select optional features and packages the theoretical number of possible software configurations surpasses the actual number built by orders of magnitudes.

4.4 Comparison of the Product Lines

Both companies can be said to have a product line, including both hardware and software, and how they develop and maintain architectures. The electrical system share a common set of features aimed at a particular market segment, e.g. premium cars or heavy commercial vehicles, and is developed from a common set of assets (e.g. a common architecture and shared systems between vehicle models). The architectures prescribe how these shared systems interact. Since these criteria are fulfilled the software are a software product line according to [9].

The two approaches to product lines were not driven by a business decision but by the development organizations adapting to their environment. Both companies were also early adopters of the practice of building several different vehicles on the same manufacturing line, implemented years before the introduction of complex electrical systems.

Supporting factors for establishing a product line of the electrical system were in Volvo's case having a rather narrow spread in vehicle models together with an explicit single options marketing strategy (versus fixed packages). This lead to a system with a high degree of configurability. In Scania's case the supporting factors were the organization wanting to develop vehicles tailored to their customers, maximizing customer value without having to redo similar development work over and over again.

Both companies handle variability in very similar way. The architecture is predominantly implemented with two mechanisms according the taxonomy by Svahnberg et al [25]: Binary replacement—physical, where different binaries can be downloaded to the flash memory of all ECUs depending on the configuration of customer-chosen optional features such as adaptive cruise control. This can be done in the manufacturing plant using the plant's product data system with separate article numbers for software as well as hardware (including nuts and bolts) and in the aftermarket using proprietary systems. At Volvo this is accomplished by the Product Information Exchange system for software [26].

The most common variability mechanism is *Condition on variable* where all ECUs get information from a central on-board file defining the configuration of that vehicle. This file is generated automatically in the manufacturing plant and flashed as a separate binary to a central ECU. Some ECUs also store local variables similarly used in a separate binary file with its own article number as well.

5 Results

The interviews yielded results mostly regarding the process for managing an architectural change.

5.1 The Process

The process for managing changes to the architecture is very similar at the two organizations with five distinct activities:

1. need
2. impact analysis
3. solution
4. decision
5. validation

This is a fairly general process, easily mapped to a generic process for architecture work seen in Fig. 3, based on [21].

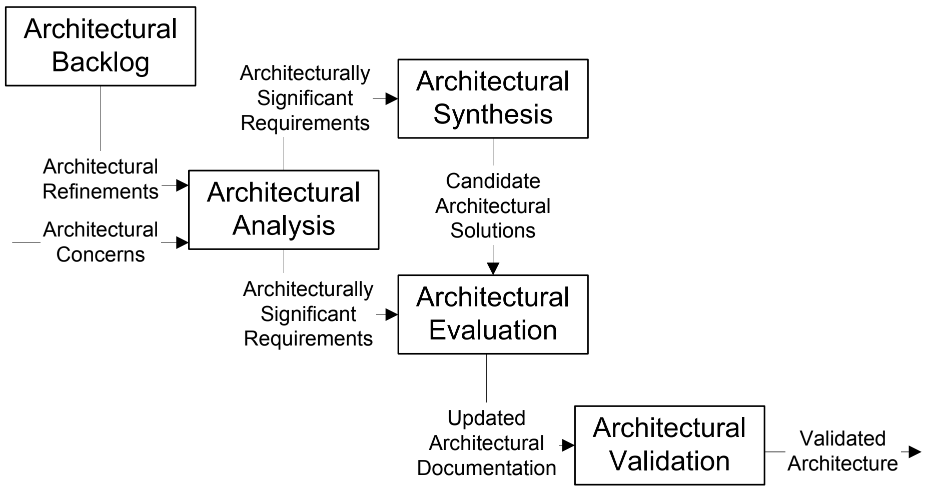
Architectural Backlog
Architectural Refinements
Architectural Concerns
Architectural Analysis
Architecturally Significant Requirements
Architectural Synthesis
Candidate Architectural Solutions
Architecturally Significant Requirements
Architectural Evaluation
Updated Architectural Documentation
Architectural Validation
Validated Architecture

Fig. 3 A generic process for creating and maintaining an architecture, adapted from [21].

At Volvo Cars there is a greater emphasis on “why” the architecture needs to be changed, as described by one of the architects on what is done first:

“Do a need analysis on what is driving the change. What isn't good enough? What change is needed?”

At Scania the architects' focus is on “how”, i.e. the impact of an architectural change. One possible conclusion is that the “why” is seen as a strategic responsibility of the senior architect at Scania, and the other architects are more concerned with “how”. Another possible reason is that the Scania architecting group has chosen to be more supportive than controlling.

5.2 Needs to Change the Architecture

Architects at both companies mention functional changes and functional growth as common reasons to update the architecture. This is not surprising since most new features are realized by electronics and software, and that the number of features grows almost exponentially [13].

At Volvo Cars all architects mention cost or cost reduction as a common reason to change the architecture, this is not surprising since the cost margins are very small and if an opportunity presents itself it is considered. At Scania cost was only mentioned by the manager, and then only in the context of how much the architectural change would cost. The most common reason to change the architecture mentioned by the architects at Scania was to adapt it to hardware changes, as described by one Scania architect: “Control units become too old; there is no room for development”.

5.3 Architecture Impact Analysis

The architects at Scania clearly seek to identify who is concerned by a change and what parts of the system are impacted by a proposed change. At Volvo Cars the architects request information about non-functional requirements or quality attributes and use cases when analyzing the impact, as described by one architect:

“I need a good description of what the customer should expect form the system. If it concerns a ready solution or if it is something we should develop internally, it could be a supplier offering something which we should integrate in the system. If there is a system solution which should be integrated I want to see that as well, if there are variants and if it is to be sold as option or standard. . . “

A possible explanation to this could be that the architects at Scania are involved earlier in the development of new features or systems, while at Volvo Cars the architects are more often given a proposed technical solution, for example by a supplier. The managers at both Scania and Volvo Cars mentioned the motive for the change as important information for understanding the change, but no other architects mentioned this. We have no explanation why this is so...

The time it takes to understand the impact on the architecture from a change seems to be similar between the two companies, a few weeks to a month calendar time. It seems to depend more on finding the right stakeholders and set up appointments with them than the actual effort in man hours from the architects. Some architects at Volvo Cars also say some architectural changes takes only minutes to evaluate the impact. This could be explained by fact that such a question would not require a official Change Request at Scania and therefore the respondents have not included these issues in their answers, or that the architects at Volvo Cars usually have a more final solution to evaluate.

5.4 Design Alternatives

Not very surprising, but notable no architecture analysis methods [7] were used or mentioned. Evaluation was in rare cases made using methods very similar to Pugh evaluation matrix [27]. Volvo architects seem to more evaluate how well different design alternatives fit into the present architecture, as mentioned by one of the architects:

“Put some different alternatives against each other and evaluate from different aspects which is best. Cost is one example. Often the need does not come from the architecture, but from different sub-systems, from the outside. When you know what needs to be done the implementation phase begins. I follow long into the project and follow up that verification is done.”

In comparison to this the Scania architects are more involved in developing different alternatives in the modelling activity. The architects see themselves as having a supporting role to function and sub-system developers. This is exemplified by

“Requirements on new functionality are often what we start with. We then balance that against the present architecture, layout of electronics and the electrical system and weigh it against our (architectural) principles. How can we enable the functionality? Sometimes it is easy to fit in and sometimes we realize we don't have the necessary hardware and that requires a bigger effort and we go through a number of steps.”

This difference in how involved the architects are in the development of subsystems is probably driven by Volvo Cars having a much larger percentage of purchased sub-systems than Scania.

5.5 Deciding on the Architectures

Architects at both companies stated that most (all?) decisions when updating the architecture were driven by non-functional requirements, quality attributes or constraints. However the attributes differed between the two companies even though the products are fairly similar, trucks/buses versus cars. The attributes deciding what update to make to the architecture could in most cases be derived from the core values for each company, for Scania Customer First, Respect for the Individual and Quality, and for Volvo Cars Safety, Environment and Quality. The attributes mentioned by Scania architects were time (to implementation), personnel resources, system utilization, including network bus load, safety, evolvability, usability, robustness, maintainability and commercial effectiveness (of which cost is a factor).

The architects at Volvo Cars unanimously mention cost as the most important factor when deciding between architectural alternatives. Other factors they mention are if the solution can realize the desired functionality, time and resources for implementation, environment friendliness exemplified by current consumption, weight, network bus load, including timing aspects, driveability, comfort and safety requirements. Risk, or minimizing the risk of a change, was also mentioned as a constraint by Volvo architects. The risk of change was not mentioned at Scania, possibly due to being obvious to think about.

A common constraint, which was mentioned by architects at both companies, was a clear wish of minimizing the effect of any architectural changes to any already existing sub-systems. The architects usually made a point of considering how a change would affect all sub-systems and not only the one proposing the change. There was a common architectural concern to have as small changes as possible, to quote one architect from Volvo Cars:

“. . . if we need to compromise so much it hurts we have not done a good job. If we don't need to compromise so much it is good.”

5.6 Validation

The most interesting result found was that none of the architects at the two companies validated the result of the implemented change themselves. Many of the architects at Scania had a clear idea of which stakeholder they would get feedback from, the integration test group. The architects at Volvo Cars were more vague when expressing how they follow up an architectural change:

“If it isn't a good solution we get to know there is a problem which we correct. Normally we assume that testing finds (anything).”

Common between the two companies was that the architects mentioned review of specifications on how a change in the architecture is followed up, but it is unclear exactly what documents the architects are reviewing.

5.7 The Resulting Artefacts from the Architects' Work

The resulting artefacts from the architects’ work on the changes to the architecture are very similar between Scania and Volvo Cars. It is the responsibility of the architects to update the network topology if a requested change affects how and where an ECU is connected to a network. At Volvo Cars the view of the topology is part of the officially released Architecture Description, one for each platform or product line, which is edited by the architect for the platform. At Scania the view of the topology is a separate document which is updated at every new release.

At both companies there will be a model describing the logical architecture captured in an UML tool. At Scania this model grows when a change concerns an area or function not previously modelled. Volvo Cars already has a more comprehensive model covering the complete existing system, so if the feature is not completely new it is more of a question of updating the existing model. Another artefact that gets updated is the signal database mentioned above. At Scania the architects defines message sequence charts (MSC) defining the interaction between ECUs, something that is not done at all by the architects at Volvo Cars.

The general conclusion is that the architects at both companies work with essentially the same type of information, but packaged slightly differently. Meetings are more emphasized at Scania, as stated from one of the architects;

"…there is more eye-to-eye communication than document communication compared to other companies I have worked at."

5.8 The Timing

The timing of when a change is introduced in the architecture varies and is driven by different factors at the two companies. At Scania the most important factor mentioned is when all concerned developer stakeholders are able to update their design. All concerned developers synchronize the changes of their assets in the product line towards a common start-of-production (SOP). These change projects are tracked on visual planning boards [28].

The timing of architectural changes at Volvo Cars is usually driven by the project timing for launching new car models (also called start-of-production at Volvo Cars), or updating a new year model of an existing car. The architects respond to these change requests if they are technically possible to do within that time frame. However, in the interviews two architects expressed hesitation when claiming that it was only the project that determined the timing. To summarize: At Scania the timing of a change of the architecture is determined by the contingence of the line organization while at Volvo Cars it is determined by the need of the vehicle model project.

5.9 Other Observations

The architects at Volvo Cars had on average worked twice as long in the company, while all architects at Scania except one had worked 4 years or less at the company. The conclusion is that at Volvo Cars the architects were recruited internally form other roles while at Scania the architects were employed specifically into that role. One noticeable difference to this is the senior architect at Scania with 21 years in the company; he is also the only one of the 11 interviewees with an official recognition as senior or expert in the two organizations.

The difference in work tasks between Scania and Volvo Cars is that at Scania the architects usually works with a specific domain, e.g. HMI or chassis systems, while at Volvo the architects were responsible for a platform and the entire system on it, e.g. the large platform (S80, V70, XC60, . . .).

6 Discussion

The striking conclusion and the answer to the stated research question is the similarity between the two companies in the tasks performed when maintaining and changing architecture. The *tasks* mentioned by the architects at both companies are virtually identical; need ⇨ impact analysis ⇨ solution ⇨ decision ⇨ validation.

The tasks do not seem to be different for architecture maintenance compared to developing a new architecture. Likewise they seem to be the same whether it is updating a product line architecture or updating the architecture of a single-shot system. Also the types of information the architects work with, one could say the viewpoints, is almost identical between the two companies. The difference being sequence charts are only used at one company but there the architects say they maintain them as a service to other stakeholders and they are not architecturally relevant. The description of the architects as lonesome decision makers made by Farenhorst et al. [10] could not be seen in this study. One possible reason for this could be the cultural differences between Sweden and the Netherlands.

The similarity in process and information is surprising since the present processes of the two companies have evolved almost independently at respective company. The similarities could be explained by the systems in cars and commercial vehicles are similar and that the companies are not too different in the demographics of their architects in terms of experience, education etc. One reason could be that the processes found can easily be mapped to a general process for architecture work, as found in [21].

As shown by Nedstam [6] there is large difference of how work is done in an organization with strong line management and a organization with strong projects. Several of the observed differences between the two companies could have affected how they work with architectural change, such as the differences in their product line approaches, the focus on project versus line organization and differences in quality attributes.

The fact that Volvo Cars has a higher degree of tool support while Scania are more conscious with respect to processes was also expected to affect the work of the architects more than was found in this study.

Acknowledgments. We would like to thank the architects and managers at the two companies for their interest and cooperation.

This work has been financially supported by the Knowledge Foundation and the Swedish Agency for Innovation Systems (VINNOVA) as part of the FFI program.

References

1. Grimm, K.: Software Technology in an Automotive Company - Major Challenges. International Conference on Software Engineering (2003) 498--503
2. Edström, A.: Hasse vill ha mer processorkraft. Elektroniktidningen (2008) 26-29
3. Edström, A.: Urban på Volvo hyllar säkerheten. Elektroniktidningen (2006)
4. Axelsson, J.: Evolutionary Architecting of Embedded Automotive Product Lines: An Industrial Case Study. In: Rick Kazman, F.O., Eltjo Poort and Judith Stafford (ed.): Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA (2009) 101-110
5. Gustavsson, H., Sterner, J.: An Industrial Case Study of Design Methodology and Decision Making for Automotive Electronics. Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, New York (2008)
6. Nedstam, J.: Strategies for management of architectural change and evolution. Lund University, Department of Communication Systems, Faculty of Engineering, Lund (2005)
7. Dobrica, L., Niemela, E.: A Survey on Software Architecture Analysis Methods. IEEE Transactions on software engineering 28 (2002) 638-653
8. Steger, M., Tischer, C., Boss, B., Müller, A., Pertler, O., Stolz, W., Ferber, S.: Introducing PLA at Bosch Gasoline Systems: Experiences and Practices. Software Product Lines (2004) 34-50
9. Clements, P., Northrop, L.: Software product lines : practices and patterns. Addison-Wesley, Boston, Mass. (2001)

10. Farenhorst, R., Hoorn, J., Lago, P., Vliet, H.v.: The lonesome architect. Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA). IEEE (2009) 61-70

11. Schulte-Coerne, V., Thums, A., Quante, J.: Challenges in Reengineering Automotive Software. IEEE Computer Society, Kaiserslautern, Germany (2009) 315-316

12. Pretschner, A., Broy, M., Kruger, I.H., Stauner, T.: Software Engineering for Automotive Systems: A Roadmap. International Conference on Software Engineering (2007) 55-71

13. Broy, M.: Challenges in automotive software engineering. Proceedings of the 28th international conference on Software engineering. ACM, Shanghai, China (2006) 55-71

14. SAE: Standard J1939 - Recommended Practice for a Serial Control and Communications Vehicle Network. Society of Automotive Engineers (2009)

15. Casparsson, L., Rajnak, A., Tindell, K., Malmberg, P.: Volcano-a revolution in on-board communications. Volvo Technology Report, Vol. 1 (1998) 9-19

16. IEEE-1471: IEEE Recommended practice for architectural description of software-intensive systems. IEEE Std 1471-2000 (2000)

17. Voget, S., Becker, M.: Establishing a software product line in an immature domain. Software Product Lines, Vol. 2379. Springer (2002) 121-168

18. Tischer, C., Muller, A., Ketterer, M., Geyer, L.: Why does it take that long? Establishing Product Lines in the Automotive Domain. 11th International Software Product Line Conference, Kyoto, Japan (2007) 269-274

19. Buhrdorf, R., Churchett, D., Krueger, C.: Salion's Experience with a Reactive Software Product Line Approach. Software Product-Family Engineering (2004) 317-322

20. Brannick, T., Coghlan, D.: In Defense of Being" Native": The Case for Insider Academic Research. Organizational Research Methods 10 (2007) 59

21. Hofmeister, C., Kruchten, P., Nord, R.L., Obbink, H., Ran, A., America, P.: Generalizing a Model of Software Architecture Design

from Five Industrial Approaches. Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture. IEEE Computer Society (2005) 77-88

22. Bergsjö, D., Almefelt, L.: Supporting requirements management in embedded systems development in a lean influenced Proceedings of International Conference on Engineering Design, Dubrovnik, Croatia (2010)

23. Johnson, H.T., Senge, P.M., Bröms, A.: Profit beyond measure : extraordinary results through attention to work and people. Nicholas Brealey, London (2000)

24. Kratochvíl, M., Carson, C.: Growing modular : mass customization of complex products, services and software. Springer, Berlin ; (2005)

25. Svahnberg, M., Van Gurp, J., Bosch, J.: A taxonomy of variability realization techniques. Software: Practice and Experience 35 (2005) 705-754

26. Melin, K.: Volvo S80: Electrical system of the future. Volvo Technology Report, Vol. 1 (1998) 3-7

27. Pugh, S.: Total design : integrated methods for successful product engineering. Addison-Wesley, Wokingham (1990)

28. Morgan, J.M., Liker, J.K.: The Toyota product development system : integrating people, process, and technology. Productivity Press, New York (2006)

Paper C

A Comparative Case Study of Architecting Practices in the Embedded Software Industry

Håkan Gustavsson
Mälardalen University
School of Innovation, Design and Engineering

Jakob Axelsson
Mälardalen University
School of Innovation, Design and Engineering

Abstract

The goal of this study is to improve the understanding of how architecting is performed within the field of software-intensive systems. Architects at six different internationally well-known companies have been interviewed to understand their way of working. This paper presents the practices that are found most successful. The context of the different companies as well as the architecting practices are compared and analyzed. Many of the architecting practices found in the study can be explained by the context of the different companies. The study shows that architects at all companies mention a general lack of understanding of software-intensive systems within industries that used to be mechanical. The architects' view of their work is very similar independently of where they work. Also the way architecting is performed is very similar, but surprisingly only one company has a defined process for architecting.

I. Introduction

Many traditionally mechanical companies in industries such as automotive, telecommunication, process automation, and defense are becoming more software intensive. The rapid increase of new functionality implemented through software enhances the burden of the system architecture to enable future growth of the system. The architecture of those software-intensive systems describes its building blocks and their relationships to each other and to the environment [10].

Architecting is defined by Maier and Rechtin [16] as the process of creating and building architectures. In our work, architecting is viewed as the process of shaping the architecture to meet customer demand by balancing requirements, guiding principles and product vision. As we see it, the architecting process is central to, and dependent on, many factors within the organization. The architects are constantly forced to make decisions on opposing factors such as continuous evolution versus product stability [20]. To stay competitive, companies need to adapt their processes to include the new discipline of software engineering.

In order to understand how different external factors affects the architecting process and to look for successful practices, the following research question is stated:

In what contexts are the methods used within the architecting process successful?

This paper presents a comparison of how architecting is performed at different companies. In the following section, related work is presented. System architecting is further defined in Section 3. In Section 4 the methodology of the case study is presented. The characteristics of the case companies are presented in Section 5. Analysis of each company in the study is then presented in Section 6 followed by general case study findings in Section 7. The results are discussed in Section 8, and the final section summarizes the conclusions and give some indications of future work.

II. Related work

There are many methods and tools available to aid the architects in their work. Examples of structured methods mentioned in industry surveys [1] are Pugh evaluation matrix [19] and the analytical hierarchy process (AHP) [22]. Dobrica and Niemela [4] make a comparison of eight different available software architecture analysis methods. The study found the Architecture Trade-off Analysis Method (ATAM) [12] to be the most suitable. The Cost Benefit Analysis Method (CBAM) [11] is an extension of ATAM and uses the quality attributes from ATAM but also considers cost when reasoning around the most suitable architecture. In a study of 46 companies in Finland [23] it was shown that the most common (76%) used concept selection method was concept review meetings, and similar results where shown in [8].

There are very few publications on how architecting of software-intensive systems are done in practice. Decisions in the development process [8] and within the architecting process [18] have been previously studied. Axelsson et al. [2] compare network architectures of three different automotive manufacturers and concludes that business and product characteristics have a large impact on the network architecture. Unphon and Dittrich [24] concludes that one must consider the organization and business domain when adopting a product line architecture. In a study of eight different software development organizations [25] it was found that the architecture is maintained and evolved through face-to-face communication rather than documents.

In a survey of 279 IT architects in the Netherlands, Farenhorst et al. [6] conclude that architects are lonesome decision makers, not very willing to share architectural knowledge, but eager to learn from others. A study made by Wallin and Axelsson [24] on architecture development at a car manufacturer presents a number of issues found within the process.

A generic process for creating and maintaining an architecture is presented by Hofmeister et al. [9]. That process is based on a comparison of five different software architecture design methods.

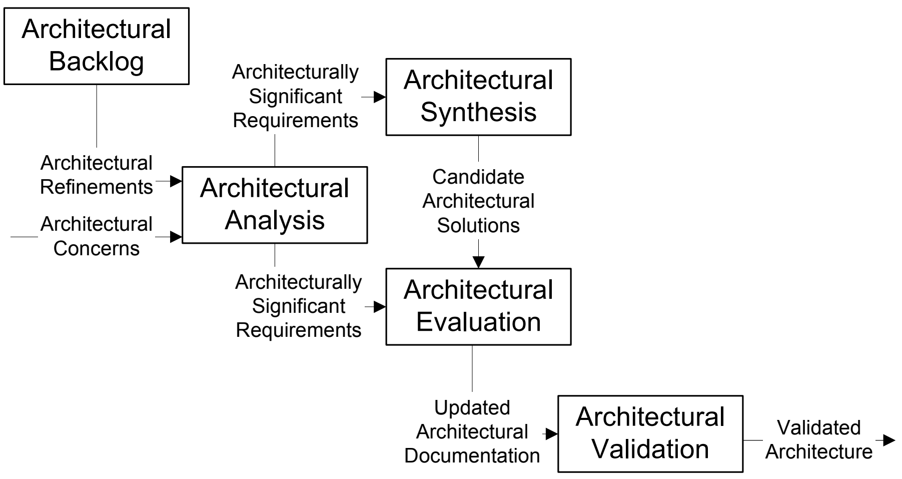
Architectural Backlog
Architecturally Significant Requirements
Architectural Synthesis
Architectural Refinements
Architectural Concerns
Architectural Analysis
Candidate Architectural Solutions
Architecturally Significant Requirements
Architectural Evaluation
Updated Architectural Documentation
Architectural Validation
Validated Architecture

Figure 1 A generic process for creating and maintaining an architecture, adapted from [9].

III. System architecting

This paper will study how architecting is performed in different companies. The study was made on companies developing embedded systems including both hardware and software. These systems are mechatronic which adds complexity since many issues cross several engineering disciplines. The systems are resource constrained and trade-offs between the system behavior and the resources required are of great importance. Both hardware and software are mixtures of in-house development and deliverables from external suppliers. The systems are distributed on different hardware platforms and are sold in a large number of variants.

Architects will make different types of decisions depending on the companies' definition of their role. Decisions will range from choosing quality attributes to mapping communication [7]. The impact of the decision will also vary depending on how decoupled software is from hardware.

IV. Method

Different companies perform architecting in various ways and there are many different factors that influence. Many of those factors are thought to be soft factors [5] that are hard to find through, for example, a questionnaire. In order to understand the context in which different methods are being used, personal interviews was found to be the most appropriate method.

The case study was performed in seven steps:

1. The questions were developed and tested on people with similar roles, who were not included in the study.
2. Companies were chosen and a connection was established through a contact person. In collaboration with the contact person the architects were identified.
3. At least two interviews were held with architects at each company.
4. The current results of the study were presented to a broader audience at each company visited. During the presentation the situation at the visited company was also discussed.
5. Questions about the characteristics of each company were answered by the contact person.
6. The results were gathered in a database and analyzed.
7. The results were also reviewed by the contact person at each participating company.

The professional network of the authors was in many cases used to establish connections with the right persons and at one company the respondents were previously known to the interviewer.

The chosen format of the interview was semi-structured and the answers were audio recorded. A semi-structured interview has predetermined questions, but the order can be modified based upon the interviewer's perception of what seems most appropriate. Question wording can be changed and explanations given [21]. The interviews at all companies followed the same template and the answers given were then used to compare the companies.

To be able to compare the companies, a number of metrics were used that are presented in Table I. Every company and organization is different in many ways, and they may use different definitions of these metrics. We choose to use each company's own definition, rather than to enforce a common definition, since this increased the likelihood of getting good responses. The values have been given by asking, for instance, how many employees are working within the company's R&D organization. The answers will not be exactly comparable since R&D is not the same in all companies, e.g. supporting units are sometimes included or not. Even if the organizations would be the same, different companies count people differently, e.g. with or without consultants. The goal of the metrics is to give an overall picture of the different companies and that goal is thought to be fulfilled even if the definitions of the metrics are not exact.

V. Case companies

The companies were chosen on three criteria:

- They do significant development of software-intensive systems.
- They are different in size and production volume.
- Together, they represent a mixture of different types of products and customers.

These criteria were chosen to give a broad spectrum of differences in business and organization, with the hypothesis that this should reflect differences in process and architecture [15, 25].

The studied companies are common in many ways. They are all financially successful and all have a very long Swedish history. They are also internationally well-known and considered premium brands within their business segments. The products are all software-intensive with a long lifecycle (15-30 years) that may include multiple owners. In the following subsections, the characteristics of each company will be presented. The comparison is summarized in Table I and some clarifications of the measures are given below:

Table 1 A comparison of the characteristics of the studied companies (all values are approximations).

Company / Context	Automotive 1	Automotive 2	Automotive 3	Defense 1	Industrial Automation 1	Industrial Automation 2
Size of R&D organization	Large	Very large	Large	Medium	Small	Small
Relative size of the embedded systems organization in comparison to total R&D	20%	13%	8%	18%	67%	24%
Number of architects	6	10	4	3+6	3+4	0-5
Management levels between architects and CEO	5	6	4	4	**2 and 4**	3
The power center of the organization	Line	Project	Project	Line	Project/Line	Project
Geographical locations of R&D organization	1	1	**~10**	1	2	3
Product variants	Very high	High	Very high	**Low**	Medium	Medium
In-house system development	50%	10%	80%	50%	**95%**	**90%**
Main customer	Business (small/large)	Private	Business (small/large)	Government	Business (large)	Business (small/large)
Magnitude of the investment for the customer	Medium/High	**Very high**	Medium/High	Small	Small	Medium/High

The size of the R&D organizations and the number of product variants is relative in comparison to the other case companies.

- The relative size of the embedded systems organization is in comparison to the total number of employees within R&D.
- The measure "number of architects" shows how many architects that are working on a complete system level.
- The power centre of the organization describes if the architects consider the organizations to be project-oriented or line-oriented.
- The magnitude of the investment for the customer indicates the size of investment relative to the economy of the most common customer.

A. Automotive 1 (A-1)

This company produces commercial vehicles. The customers of the vehicles are both small and large companies. The product can be configured in a very high number of product variants. This is done using a common product line architecture that supports all different variants. The company has its R&D centralized to one location and has for a long time applied the thoughts of Lean [17] onto its development.

B. Automotive 2 (A-2)

This company is a car producer. The customers of the vehicles are mostly individuals and in some cases companies. This makes the magnitude of the investment for the customer often very high. The company has the largest R&D organization of the companies included in the study and its R&D centralized to one location. The relative size of the electronic and electric system development organization is 13 percent, which is explained by a low degree of in-house development. The architecting is divided into two groups responsible of traditional electrical systems and software-intensive systems.

C. Automotive 3 (A-3)

This is another producer of commercial vehicles. The company has R&D located at more than 10 different locations worldwide. As with A-1 the product can be configured in a very high number of product variants. The different product lines use the same software and hardware architecture on most in-house developed subsystems, but the interface between subsystems are not standardized between the different product lines.

D. Defense 1 (D-1)

As with most companies in the defense industry, the main customers are governments in different countries. The product variants are in comparison low. Customers usually purchase a unique variant of an existing product. The customer requirements are often detailed and may include demands on using a specific supplier of subsystems. The company has its R&D centralized to one location. There are three architects working on the complete system and six who work only with embedded systems.

E. Industrial Automation 1 (I-1)

The customer is mostly large companies. The development is mainly in Sweden, but some development is also done in Asia. The relative size of the electronic and electric system development organization is 67 percent, which is explained by a high degree of in-house development. The system is often integrated into a larger system. There are three architects working on the complete system and four who work only with embedded systems.

F. Industrial Automation 2 (I-2)

The customer of the systems is both small and large companies. As with I-1 the development is mainly in Sweden, but some development is also done in Asia and the US. The system is usually a major investment for the customer. The electronic and electric system development organization is the smallest of the companies included in the study.

VI. Analysis

The key architecting practices that differentiate from how work is done in the compared companies are presented below.

Company A-1 has a defined documented process for architecting. The progress of each task is visualized and controlled during a weekly follow-up meeting. Knowledge sharing is performed through lessons learned sessions after each large release. The high acceptance of processes makes architecting easier at A-1. The threats lie instead in the lack of tool support.

Company A-2 has separated the roles of modelling and architecting. The architects are not responsible for updating the architectural model. This is done by a group of people specialized in modelling. The architectural task is discussed at a weekly follow-up meeting. Company A-2 is also the only company in the study having a complete and updated model of the entire system. The division of the architecting into two groups does not seem to have any positive effects. Instead it causes friction and prevents the flow of information between the architects.

Company A-3 has been using a common software and hardware platform for a long time. This enables easy change of software components. The different product organizations are making decisions which affect the overall architecture without consulting the architects. The reason for this might be the relatively small amount of available architects.

The defense company D-1 was, not surprisingly, a master of requirement management. Requirement management is performed in the other companies, but not at the same detailed level. The requirement management system is also used to document reasoning of the design decisions. That knowledge is then used when changes are made to the design. The company manages to balance a strong system engineering practice with the agility of a medium size company. As with all the companies in the study this is historically a mechanical company, but the management's understanding of software-intensive system seems to be lower in D-1.

Company I-1 has two different types of architects: system architects and global architects. The global architect is the connection between strategy and business goals. The global architect has a budget and is thereby in a position to make larger architectural changes without being part of a project. Company I-1 uses roadmaps to communicate and create a common vision. This work is also a task performed by the global architect.

Company I-2 does not have the formal role of an architect, but is currently reviewing their way of working with electronic and electric system development. The need of some kind of coordinating role is very obvious and they are very aware of this fact. The different product lines have been developed more or less independently from each other and there has been little reuse of components. The company is very agile and innovative. In a future transformation those abilities must be given attention in order to keep that positive climate.

VII. Case study findings

Architects at all companies mention a general lack of understanding of software-intensive systems within industries that used to be mechanical. The issue exists both at management level as found in [24] and with other stakeholders.

A. The Role of the Architect

The architects' view of their work is very similar independently of where they work. The architects primarily view themselves as facilitators, involving the right stakeholders in the architectural decisions or problem solving. They also consider themselves as coordinators and communicators of changes influencing the overall architecture.

B. Defining Architecture

When asked to explain what architecture means to them, most architects mention structure and form, some mention the building blocks and its interfaces. The user of the system is not often mentioned, only 40 percent. Only two architects mention business aspects and those two are both very senior:

The architecture is what connects the technology with the business model and culture of the company.

The architecture is the way we put the parts together to achieve our goal, but it also includes the organization and business.

C. Architectural Analysis and Synthesis

The most common methods used are design review meetings and safety analysis. Simulation of network utilization is also performed. One company has a predefined form for describing alternatives, but it is very rarely used. Alternative solutions are rarely documented or, as stated from one respondent:

Alternative solutions are often documented on a whiteboard or in some cases in an email.

D. Architectural Evaluation and Validation

There are no formal evaluation methods used as the ones mentioned in Section 2. Only one company mentioned feedback from test as a way of validating the architecture:

If it isn't a good solution we get to know there is a problem which we correct.

E. Process Improvement

The processes at all companies are very similar to the one described in Figure 1, with one big exception: there is no structured synthesis available at any company. It is also interesting that only company A-1 has a defined process for architecting. When asked what they would like to change in their way of working in order to improve, most mentioned how architectural knowledge [13] is managed.

The following answers to the question “How do you know if the architecting process is working well?” presents the architects’ view of a healthy architecting process:

We do not really know, but the number of changes that are flowing the right way through the change review meeting is an indication.

When new functionality can be absorbed by the architecture without the need of large changes.

When the architecture is clearly communicated and there is no discussion about small issues.

F. Organization

As seen in Table I, the architectural teams are located on approximately the same hierarchical level relative to the size of the organization. The number of architects in A-1 and A-3 is significantly lower than A-2. This is mentioned as a problem by the architects at both companies. The two global architects at I-1 is the only case where architects have a clear responsibility for coordinating roadmaps.

A-3 is the only company with a large distributed development organization including sites worldwide. They experience difficulties in getting feedback on architectural changes. In the case of I-1 and I-2 the development made on other sites is very capsulated and they did not experience any large difficulties. I-2 had representatives from the other development sites on the main site. This made the cultural barrier less of a problem.

VIII. Discussion

The findings presented in the previous chapter are facts found analyzing the answers in the interviews. During the visits to the companies the authors have also built their own understanding of what the differences in how architecting is done depend upon. Those thoughts are presented below.

We see a clear correlation between the perceived maturity level [3] of the different organizations and how knowledge is shared. All companies have a very high degree of informal communication, but architects at the companies that have recurrent meetings are more pleased with the information available.

The different types of customer of the final products create different architectural concerns. The magnitude of the investment for the customer of products delivered by companies D-1 and I-1 are mostly small (Table I). This might be the reason why cost seems to be of lower priority at those companies. In contrast, at A-2 where the magnitude of the investment for the customer of the product is very high (Table I), cost is mentioned very often.

Kruchten [14] suggests that the productive time spent by architects can be classified into three categories of communication: internal (architecture design), inwards (input from outside world) and outwards (providing information). He argues that they should be roughly in the ratio 50% internal, 25% inwards, and 25% outwards. It is very hard to measure this in practice and we have not done so in this study, but communication patterns can still be observed. Even if no extreme variation can be seen, the understanding from this study is that there is a clear difference between the companies. The architects tend to be more satisfied when the inward and outward communication is distributed evenly and where the internal work is of significant size. Company A-3 and I-2 are examples of where the low number of architects supporting a large organization makes the time available for architecting too short. This results in architecting being performed by the developing groups without taking into account the overall system.

The power centers of an organization also affect how the work with the architecture is done. Nedstam [18] shows that there is a large difference in how work is done in an organization with strong line management and an organization with strong projects. This is found to be true also in this study. In the companies with a strong line organization, the line controls the architecting process, while in the companies with a strong project organization the process is controlled by the project. At company A-2 the power of development lies in the projects (Table I). The pressure from the projects might be the reason why the end customer is sometimes neglected. This could be the reason of the over-the-wall tendency, meaning that the deliveries of the documents are more important than the knowledge within.

IX. Conclusions and future work

This paper has presented the current state of architecting practices in three different industrial segments characterized by being software-intensive. For academia it presents a current view of how architecting is performed. The industrial reader is given a list of practices that can be used as an inspiration to improve the current architecting practice.

Many of the differentiating practices found in the study can be explained by the context of the different companies. The use of global architects with their own budget in I-1 is a solution to initiate long term architectural projects without having a customer order. The high degree of documented reasoning in D-1 is caused by the high degree of customer specific demands and large orders of very similar products. This forces the architects to make branches of the architecture to fulfill the customer demands and the reasoning is then used to ensure quality. The defined architecting process found at A-1 and the use of visualization tools to track progress is explained by influences of Lean. Other practices such as the divided architectural teams in A-2 and the lack of formal architects in I-2 are more difficult to explain.

During the study it has been seen how the balance of power between line and project strongly affects how work is done. This relation would be of interest in a future study. The connection on how business strategy concerning Cost, Quality and Time-to-Market affects architecting could also be further analyzed.

The description of the architects as lonesome decision makers made by Farenhorst et al. [6] could not be seen in this study. One possible reason for this could be the cultural differences between Sweden and the Netherlands. Future work could therefore include studying companies in other countries. The methodology used was found to work very well. The presentation after the interviews at the visited company was found to be much appreciated. It was also an efficient way to validate the understanding given through the interviews.

Acknowledgement

We would like to thank all of the interviewees for contributing with all their experience and knowledge. Special thanks go to the contact persons at the different companies for coordinating the visits and reviewing the final result. This work has been financially supported by the Knowledge Foundation and the Swedish Agency for Innovation Systems (VINNOVA).

References

[1] C. S. Araujo, "The utilization of product development methods- A survey of UK industry," Journal of Engineering Design, vol. 7, pp. 265-277, 1996.

[2] J. Axelsson, J. Fröberg, H. Hansson, C. Norström, K. Sandström, and B. Villing, "A Comparative Case Study of Distributed Network Architectures for Different Automotive Applications," in The Industrial Information Technology Handbook, R. Zurawski, Ed. Boca Raton, USA: CRC Press, 2005, pp. 57-1 to 57-20.

[3] J. Axelsson, "Towards a process maturity model for evolutionary architecting of embedded system product lines," in Proceedings of the Fourth European Conference on Software Architecture Copenhagen: ACM, 2010, pp. 36-42.

[4] L. Dobrica and E. Niemela, "A Survey on Software Architecture Analysis Methods," IEEE Transactions on software engineering, vol. 28, pp. 638-653, 2002.

[5] R. Farenhorst and R. C. d. Boer, Architectural knowledge management : supporting architects and auditors. [S.l.: s.n.], 2009.

[6] R. Farenhorst, J. Hoorn, P. Lago, and H. v. Vliet, "The lonesome architect," in Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) & European Conference on Software Architecture (ECSA): IEEE, 2009, pp. 61-70.

[7] B. Florentz, Software and System Architecture Evaluation and Analysis in the Automotive Domain. Braunschweig: Technische Universität, 2008.

[8] H. Gustavsson and J. Sterner, "An Industrial Case Study of Design Methodology and Decision Making for Automotive Electronics," in Proceedings of the ASME International Design Engineering

Technical Conferences & Computers and Information in Engineering Conference New York, 2008.

[9] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran, and P. America, "Generalizing a Model of Software Architecture Design from Five Industrial Approaches," in Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture: IEEE Computer Society, 2005, pp. 77-88.

[10] IEEE-1471, "IEEE Recommended practice for architectural description of software-intensive systems," IEEE Std 1471-2000, 2000.

[11] R. Kazman, J. Asundi, and M. Klein, Making Architecture Design Decisions: An Economic Approach: Carnegie Mellon Software Engineering Institute, 2002.

[12] R. Kazman, M. Klein, and P. Clements, "ATAM: Method for Architecture Evaluation," 2002.

[13] P. Kruchten, P. Lago, and H. van Vliet, "Building up and reasoning about architectural knowledge," Quality of Software Architectures, pp. 43-58, 2006.

[14] P. Kruchten, "What do software architects really do?," Journal of Systems and Software, vol. 81, pp. 2413-2416, 2008.

[15] S. Larsson, A. Wall, and P. Wallin, "Assessing the influence on processes when evolving the software architecture," in Ninth international workshop on Principles of software evolution: in conjunction with the 6th ESEC/FSE joint meeting Dubrovnik: ACM, 2007, pp. 59-66.

[16] M. W. Maier and E. Rechtin, The art of systems architecting. Boca Raton: CRC Press, 2002.

[17] J. M. Morgan and J. K. Liker, The Toyota product development system : integrating people, process, and technology. New York: Productivity Press, 2006.

[18] J. Nedstam, Strategies for management of architectural change and evolution. Lund: Lund University, Department of Communication Systems, Faculty of Engineering, 2005.

[19] S. Pugh, Total design : integrated methods for successful product engineering. Wokingham: Addison-Wesley, 1990.

[20] E. Rechtin, Systems architecting : creating and building complex systems. Englewood Cliffs, N.J.: Prentice Hall, 1991.

[21] C. Robson, Real World Research-Second edition: Blackwell Publishers Ltd., Oxford, UK, 2002.

[22] T. L. Saaty, "How to make decisions: The analytic Hierarchy Process," Journal of Operational Research, vol. 48, pp. 9-26, 1990.

[23] M. Salonen and M. Perttula, Utilization of concept selection methods – a survey of finnish industry. Helsinki, Finland: Helsinki University of Technology, 2005.

[24] H. Unphon and Y. Dittrich, "Organisation matters: how the organisation of software development influences the development of product line architecture," in Proceedings of International Conference on Software Engineering, Innsbruck, Austria, 2008, pp. 178-183.

[25] H. Unphon and Y. Dittrich, "Software architecture awareness in long-term software product evolution," Journal of Systems and Software, 2010.

[26] P. Wallin and J. Axelsson, "A case study of issues related to automotive E/E system architecture development," in 15th IEEE International Conference on Engineering of Computer Based Systems (ECBS) Belfast: IEEE, 2008, pp. 87-95.

[27] F. van der Linden, J. Bosch, E. Kamsties, K. Känsälä, and H. Obbink, "Software Product Family Evaluation," in Third International Software Product Lines Conference. vol. Volume 3154 Boston: Springer Berlin, 2004, pp. 110-129.

Paper D

Evaluation of Design Options in Embedded Automotive Product Lines

Håkan Gustavsson
Mälardalen University
School of Innovation, Design and Engineering

Jakob Axelsson
Mälardalen University
School of Innovation, Design and Engineering

Introduction

In many industries, complex embedded product lines are designed. In theory, this follows a structured and well-organized process, where a set of given requirements are step-by-step transformed into an optimal product. However, in reality the complexity of the products and markets often lead to much less stringent ways of working. Let us consider a fictive, but not atypical, scenario.

Improvements of the existing product are debated during coffee breaks and in the hallways. Ideas are discussed and eventually a new function is developed. The new function is not part of any project and no budget exists. Instead it is the creation of highly motivated developers and their ambition to improve the product.

To implement this new functionality they need management support which is created through prototype demonstrations. From this point in time everything moves very rapidly. Customers are invited to workshops and it turns out that they are willing to pay for the functionality, but it will probably only be sold in low volumes to a high end segment. The decision is made to introduce the function as fast as possible based on very uncertain information. The function that has been demonstrated is developed using components made for an experimental environment. That does not fit to the *current system architecture and is not suitable for production. Management stresses that time-to-market is important and it is assured that the quality of the product will not be affected if implemented as is. Therefore the decision is made to integrate the function rapidly even if the chosen solution does not follow the common design rationale and removes many degrees of freedom for the future evolution of the system.*

This was system development as a short fictive story. It is hopefully not too common in practice, but it still includes many of the issues that most system developers have experienced in different projects. The solution solves the problem today, but could cause difficulties in the future (a situation referred to as "technical debt" by Cunningham (1992)).

The developers did not have the methods available to evaluate and show the economical value of a longer term solution. Such methods would be very useful early in the design process when uncertainty is high. Functions developed in this fashion are likely to be innovative and meeting the demands of the customer but could severely impact the future flexibility and adaptability of the system. A valuation of the resources early in the design process could remedy this problem and reduce the lifecycle cost of the system. If the designs are made in a structured manner, the design decision will be traceable and continuous improvements are more likely to occur.

This chapter discusses how to deal with scenarios like this by putting a value on flexibility in the system solution. Thereby, it becomes clearer when to focus on short term solutions and when to keep the long term evolution of a product line in mind. The approach taken is to evaluate flexibility using a concept called Real Options. The method is motivated and described by using as example an industrial area where very complex product lines occur, namely automotive embedded systems. To improve the usability of the method a structured evaluation process is defined to aid practitioners such as developers and architects. The evaluation process provides a way of valuing system designs and enables the practitioner to think about the future in a systematic manor. The value of a flexible design can thereby be quantified and the proposed process shows how it can be accepted by practitioners within the automotive industry.

In the next section, an overview of automotive electronics and software is given. Then the concept of Real Options is introduced, followed by a discussion on how it can be used in embedded system design. The following two sections present a step-by-step approach to evaluating flexibility in embedded systems, first as a theoretical process and then applied to a case from automotive electronics. The final two sections discuss related work and summarize the conclusions of the chapter.

Automotive embedded systems

Today most innovations made within the automotive domain are driven by electronics. According to a study made by Hoch et al. (2006) the total value of electronics in automobiles is expected to rise from the current 25% to 40% in 2010. The automotive customers demand new functionality with every new product release and the time-to-market is constantly shortened.

An example of new functions is Advanced Driver Assistance Systems (ADAS) that help the customer to drive the vehicle safety. Those systems typically use information about the surrounding to increase road safety. This is done by using sensors to identify nearby objects or communication with other vehicles or infrastructure to attain more information. The increased interaction between various components and the wider boundaries of the system increases the complexity and demand flexibility to be easily integrated.

There are many other new functions that are about to be introduced or are already introduced that have a large impact on the electrical system of automotive vehicles. To cope with this continuous change the system needs to be designed with the right amount of flexibility. It is crucial that each of those functions can be implemented without causing large system-wide changes.

Further complexity is added by the fact that the vehicle developers strive to use a product line approach, where the same embedded system is used in a wide range of vehicles. The base system thus needs to be able to evolve over a long time and be adaptable to very different surroundings.

System architecture

The building blocks of an automotive electrical and electronic (E/E) system consist of electronic control units (ECUs) executing the software modules that implement the functionality. ECUs are connected to communication networks. As shown in Figure 1 the communication networks are usually divided into sub networks and the communication between those are made through gateway ECUs connected to a backbone. Different sensors and actuators are connected to the ECUs depending on the function allocated to the ECU.

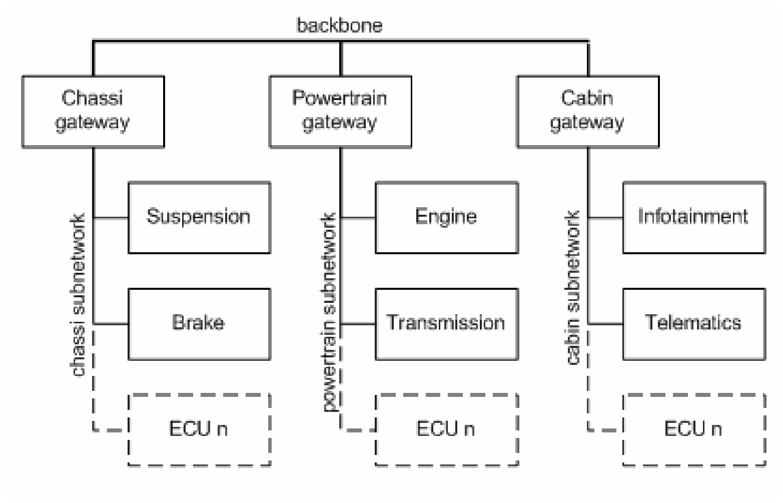

Figure 1 A typical vehicle communication network.

Most design decisions of automotive E/E architectures are done during the early phases. Often, the E/E architecture needs to support a full product line of vehicles or vehicle variants that are released over a number of years. They must allow a large degree of variability to cope with the demands of different customers. The long life-cycle of automotive products demand that changes to the product can be made with as little impact to the different components as possible.

To be able to satisfy the growing demand on functionality the Original Equipment Manufacturer (OEM) needs to develop architectures that can evolve throughout its lifetime without forcing premature architectural changes. Similar products in some other industries solve this problem by simply adding extra resources to cope with future demands. The cost sensitive automotive industry has to optimize the use of the system's limited resources, but in the meantime also be flexible. The design decisions are usually based on many factors that pull in different directions such as maintenance, portability, usability etc. The complexity of the system and the many uncertain factors create a need to define methods which can provide guidance in the design process.

Decision levels

Architectural decisions are made when selecting components and allocating them to subsystems, which then are combined into a system. The decisions can be made on different levels which have various impacts and predictability. Florentz et al. (2007) group the decisions into three levels; top-level, high-level and low-level (Figure 2). Top-level decisions concern the quality and function attributes and have the largest impact. Choosing architectural patterns and technologies are found to be high-level decisions. The most predictable decisions are those concerning the hardware architecture and function mapping. The impact of the decision will vary depending on how decoupled software is from hardware. This work has been focused on the low-level decisions concerning function and communication mapping.

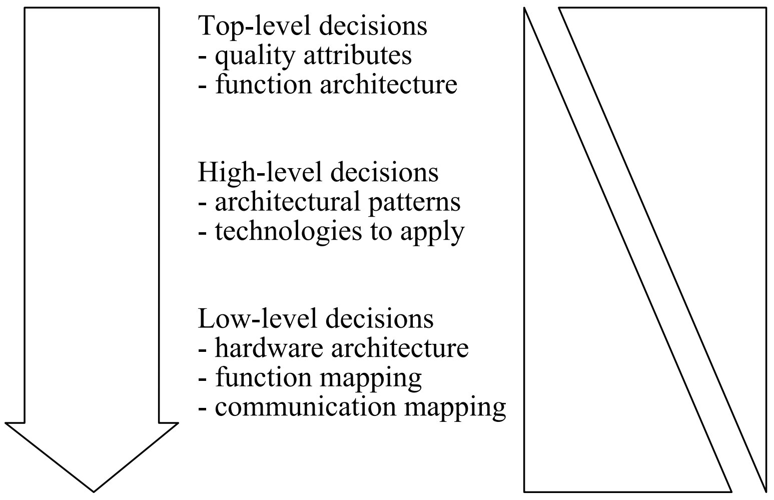

Figure 2 Decisions made during the development of the architecture will have different impact and the outcome will be more or less predictable (Florentz and Huhn. 2007).

Introducing real options

In this section, the concept and background of options in general and Real Options in particular is introduced.

Financial options

Using options theory is one approach to deal with the high level of uncertainty when making design decisions in the early phases. The theory derives from finance where an option is the right but not the obligation to exercise a feature of a contract at a future date (Hull 1993). A typical example is a stock option which gives the right but not the obligation to buy a certain stock at a given price on a predefined date. An option has a value because it gives its owner the possibility to decide in the future whether or not to pay the strike price for an asset whose future value is not known today. An option therefore provides a right to make the costly decision after receiving more information.

There are two different types of options, American and European. A European option may only be exercised on the predefined exercise day whereas an American option can be exercised any time until the exercise date.

Real options

Since the 1990s options theory has started to be utilized within the field of engineering. It is then called Real Options and was developed to manage the risk of uncertain design decisions. Real Options could be seen as an extension of financial option theory to options on real (nonfinancial) assets. Copeland and Antikarov (2001) defines a real option as:”the right, but not the obligation, to take an action (e.g. deferring, expanding, contracting, or abandoning) at a predetermined cost called the exercise price, for a predetermined period of time - the life of the option.”

In 2001 de Neufville coined the expressions Real Options "in" and "on" projects. Real Options "on" projects treat the enabling technology as a black box while Real Options "in" projects are options created by changing the actual design of the technical system. Real Options on projects provide a more accurate value of the project and Real Options in projects support the decision on what amount of flexibility to add. ”Real Options on projects are mostly concerned with an accurate value to assist sound investment decisions, while Real Options in projects are mostly concerned with go or no go decisions and an exact value is less important.” (de Neufville 2001)

Social considerations

Real Options do not only provide a way of valuing system designs, but it also forces the developer to think about the future in a systematic manor. By giving future flexibility a value it assists the developing organization in making decisions and also enables a way of predicting the growth of the complete system (Larses 2005). Leslie and Michaels (1997) concludes the article ”The real power of Real Options” with ”The final, and perhaps greatest, benefit of Real Option thinking is precisely that - thinking”. The possibility of changing the way people think might also be the hardest part in bringing acceptance to new methods such as using Real Options. The new method must not only be better than the one it is replacing, it should also be triable, observable and have low complexity (Copeland and Antikarov 2001).

Valuing real options

One of the advantages with Real Options compared to many other architecture evaluation methods is the possibility to value different system designs and thereby finding the most economically sound investment. This is probably the most complicated part of using Real Options, and over the years several approaches to calculating its value have been proposed. They all have various assumptions and we will in this section evaluate the most appropriate for our case. Amram and Kulatilaka (1999) propose three general solution methods:

- *Black-Scholes-Merton model.* This method calculates the option value by solving a partial differential equation including the value of a replicating portfolio.
- *Binomial model.* The dynamic programming approach lays out the possible future outcomes and folds back the value of an optimal future strategy.
- *Monte Carlo* simulation. The simulation approach averages the value of the optimal strategy at the decision date for thousands of possible outcomes.

We will now present the first two models in more detail, whereas the third model is beyond the scope of this study. (It should be pointed out that the method described later in this paper does not require the practicing engineer to understand, or even be aware of, the calculation method.)

Black-Scholes-Merton model

The Black-Scholes-Merton (BSM) model, for which they later received the Nobel price, was created by Black and Scholes 1973 and is widely used on financial options. The BSM model makes two major assumptions that concern our case: it demands a replicating portfolio and it only supports European type options.

A replicating portfolio contains assets with a value matching those of the target asset. The replicating portfolio of financial options can easily be found on the stock exchange as the stock value, but when looking at Real Options that are not traded it can be very difficult to find.

Considering our case it seems very unlikely that the assets needed are exercised at a predefined time. Sullivan et al. (1999) discuss the assumptions made and write:”They will not hold for some, perhaps many, software design decisions.” More recently (Copeland and Antikarov 2001) argue: ”There are valuation methodologies that effectively capture the complexities and the iterative nature of managerial decisions, and the Black-Scholes-Merton model is not the only, or even the most appropriate, way to value Real Options.” Also Amram and Kulatilaka (1999), who provide a four step solution using BSM, state: ”The Black-Scholes solution is appropriate for fewer Real Options applications, but when appropriate it provides a simple solution and a quick answer.” The conclusion is that the BSM model is suitable for financial options, but hard to use in our case.

Binomial model

The binomial model does not need a replicating portfolio (Banerjee 2004) and also supports American type options. The initial value, A, changes with each time interval and either goes up with the probability p to Au or down to Ad until its final date (Amram and Kulatilaka 1999). The value of the asset (A) at each decision point is given through Equation (1) with r being the risk free interest rate and σ the volatility and the time period Δt.

$$A = (pA_u + (1-p)A_d)e^{-r\Delta t} \qquad (1)$$

Assuming that the underlying asset has a symmetric up and down movement u = 1 / d, then the up and down factors are given through:

$$u = e^{\sigma\sqrt{\Delta t}} \qquad (2)$$

$$d = e^{-\sigma\sqrt{\Delta t}} \qquad (3)$$

The probability of an up movement is then:

$$p = \frac{e^{r\Delta t} - d}{u - d} \qquad (4)$$

Looking back at our case the value of the flexibility option would change during the development stages (see Figure 3).

Real options in embedded system design

There are as many Real Options in embedded system design projects as in any other engineering project. Those systems contain a large amount of design variables and parameters that can be valued as Real Options in projects.

Suitability of real options

To find out if Real Options would be a support in embedded system design one needs to clarify the characteristics of this domain. As stated earlier (Hoch et al 2006) the large volume and cost of the product makes errors in the design very expensive. Also, conflicting requirements found late in the development phase cause a high cost. At the same time there is a very high level of uncertainty during this design phase and important decisions are made by a small group of engineers (Axelsson 2006). The automotive embedded systems are characterized by being mechatronic systems which adds complexity. The systems are often resource constrained and trade-offs between the system behavior and the resources required are of great importance (Larses 2005).

When to use Real Options is explained by many authors. Copeland and Antikarov (2001) state that ”It is making the tough decisions - those where the Net Present Value is close to zero - that the additional value of flexibility makes a big difference.” This is in our case true when developing a new functionality where the market demand is very uncertain. If the design would include a real option to abandon or change course the risk taken could be minimized. Under these conditions, the difference between real option valuation and other decision tools is substantial.

Real options in automotive systems

There are many new functions that are about to be introduced or are already introduced that have a large impact on the electrical system of automotive vehicles. Using Real Options as a method to evaluate alternative solutions gives the possibility to value the flexibility of the technical solution. A solution that is more likely to withstand change due to future demands has therefore a higher value when evaluated using real options compared to traditional evaluation methods. To enable the possibilities of future reuse the system needs to be designed with interfaces between components (both SW and HW) that are prepared for future needs.

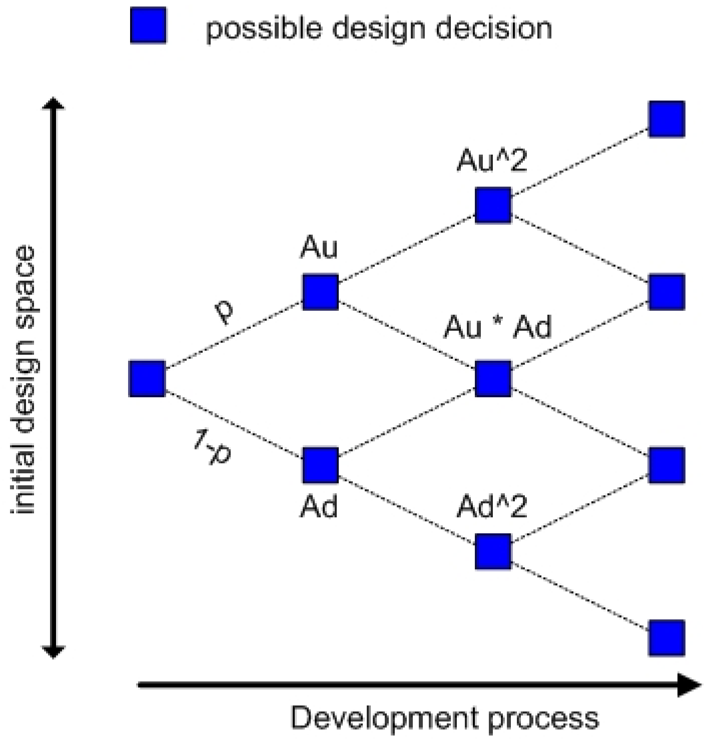

Figure 3 The decisions made narrow the initial design space.

The design will be different depending on how long the system is planned to withstand future change. To evaluate what level of flexibility is appropriate one must therefore first provide the rough requirements of future needs. Given the estimated value of the future functionality a Real Options analysis will then show what amount of flexibility should be added to make the investment adequate. Current and future technical demands of the system together with economical and organizational demands call for a systematic evaluation process.

Evaluation process

To improve the usability we have defined an evaluation process that can aid practitioners such as developers and architects of embedded automotive systems. Practitioners working with embedded systems are often not used to value design alternatives with economic valuation methods. To make the practitioners utilize and trust the method it is important to present a step-by-step process how to carry out the valuation. During the evaluation process the different stakeholders will have to specify their gut-feeling in figures and consider if flexibility has an added value. The evaluation process presented in Figure 4 below consists of eight steps with a description and some concrete advices. (In the next section, the steps will be exemplified in a small case study.)

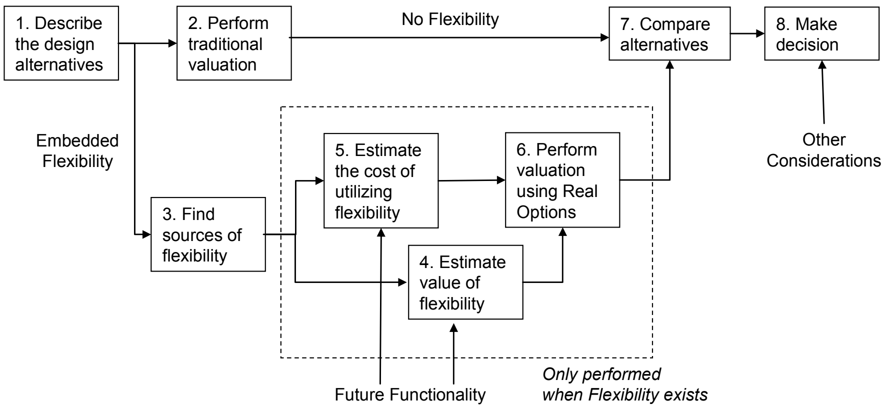
1. Describe the design alternatives
2. Perform traditional valuation
No Flexibility
7. Compare alternatives
8. Make decision
Embedded Flexibility
5. Estimate the cost of utilizing flexibility
6. Perform valuation using Real Options
Other Considerations
3. Find sources of flexibility
4. Estimate value of flexibility
Future Functionality
Only performed when Flexibility exists

Figure 4 The eight step evaluation process.

Step 1 - Describe the design alternatives

Each valid design alternative is described to identify what resources are used. This can be simplified by reusing patterns from previous designs.

Step 2 - Perform traditional valuation

The traditional method to derive the value of an investment is by calculating its Net Present Value (NPV) taking into account the value today of cash received or paid in the future. To calculate NPV a discount rate is used, often corresponding to the current interest rate.

Step 3 - Find sources of flexibility

It would not be wise to analyze all the real options available. When designing a function distributed over a communication network there are some assets that are generic and can easily be used by other functions. Those represent the source of flexibility or Real Options. Commonly they are hardware assets such as inputs, outputs or communication capacity. If there is such an asset, the difference in NPV could be due to the cost of designing for flexibility. If there is no source of flexibility the result given through the valuation in Step 2 is true, and the evaluation is completed.

Step 4 - Estimate value of flexibility

Each resource is analyzed to distinguish if it has a future value. When available it provides an increased amount of flexibility or available design space and thereby an added value.

The value will often be due to the revenue of future functions which represent the underlying asset (S) and can be calculated through a simplified model (5). The product cost is the estimated costs during the system lifecycle.

$S = volume \times (customer\ price - product\ cost)$ (5)

Of course, a more elaborate model can also be used, if more detailed information is available.

Step 5 - Estimate the cost of utilizing flexibility

Utilizing the flexibility is usually a question of implementing a future function or extension of an existing function. The price to be paid is therefore the added cost of implementing this future functionality. Figure 5 illustrates how the added cost (the exercise price of the real option) will be paid later in the lifecycle of the system when the flexibility is utilized.

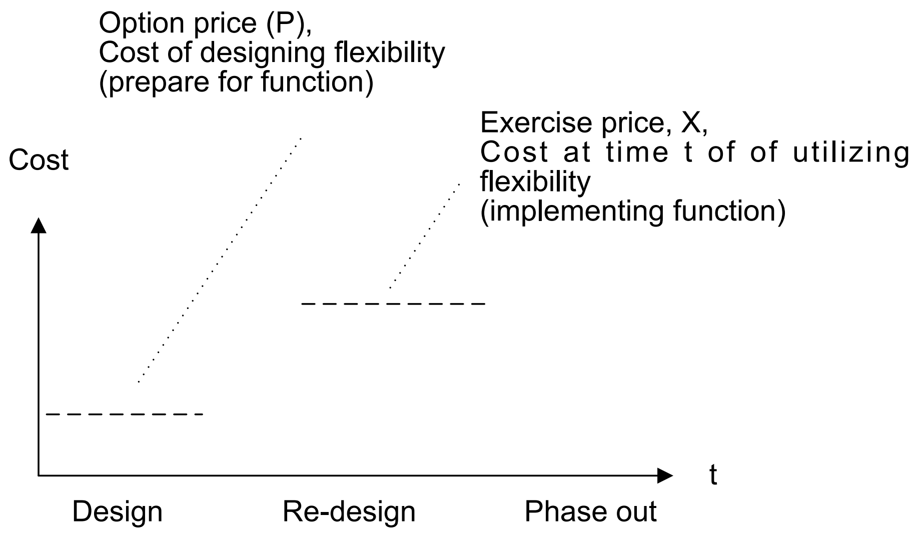

Figure 5 The price and exercise price of the option.

Step 6 - Perform valuation using Real option

The value of the flexibility can be calculated using real option valuation. The quantitative data needed, shown in Table 1, to perform a real option valuation should be extracted for the design concepts as follows:

- The planned lifetime of the platform needs to be estimated. If the function has not been implemented before the expiration date the value of the real option is considered to be lost.
- The current value of implementing flexibility is the result from Step 4.
- The cost of utilizing the flexibility is given from Step 5.
- The volatility is a measure of the annual up or down movement of the option value and often represents the uncertainty of future customer demands. This can be estimated through historical data or expert assessment.

By using the binomial model the value of the option premium can be calculated.

Table 1 Factors affecting the value of an option.

Option on stock	Real option in embedded systems
Option value (V)	The value of designing flexibility
Option price (C)	Cost of designing for flexibility
Exercise price (X)	Cost of utilizing flexibility
Underlying asset value (S)	Current value of implementing flexibility
Volatility (σ)	Uncertainty of costumer demand
Time to exercise (t)	Time when the option is exercised
Time to expiration (T)	Lifetime of the current system

Step 7 - Compare the alternatives

Real option theory provides an extension to the traditional NPV valuation by adding the value of flexibility. The so called expanded NPV is the sum of the static NPV and the value of the option premium (Trigeorgis 1988):

Expanded NPV = Static NPV + Option premium (6)

The best investment is therefore to choose the design alternative with the highest Expanded NPV.

Step 8 - Make decision

Real Options provide the opportunity to analyze the cost of designing for future growth of a platform, based on the estimated value of the future functionality. It is important to stress that decision are often based on factors that are not valued using the presented evaluation process. Other factors that influence the decision are the choice of supplier, time-to-market, project priority or organization. The last step is therefore to make the decision based on the trade-off between all influencing factors.

Case study: Network usage

To analyze the process and its usefulness it is applied on a real case taken from the automotive industry. The problem is how to integrate a new feature implemented in software into an existing E/E architecture. A key element of the problem is in which ECU the new functionality should be implemented.

Step 1 - Describe the design alternatives

A pre-study has found two alternative ways to provide this feature (Figure 6). Design alternative 1 provides this feature by connecting the external communication link directly to the current cabin gateway ECU through an existing but unused bus interface, and the advantage is a low development cost.

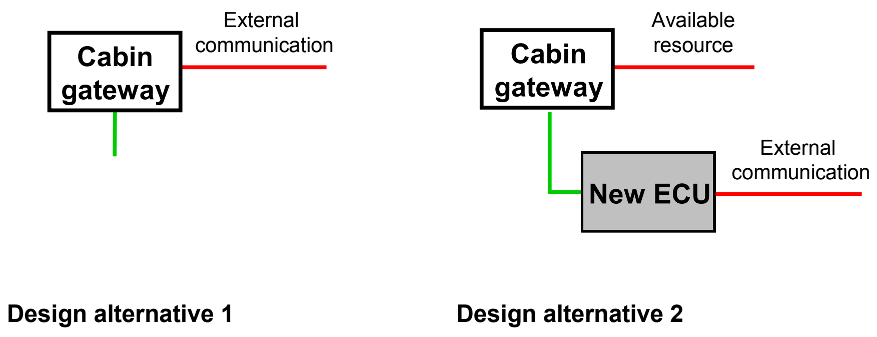

Figure 6 Two design alternatives that differ in their use of communication links to provide the demanded feature.

Design alternative 2 uses a new ECU to create the external communication. The new ECU connects to the cabin gateway using an already implemented internal network. Alternative 2 is more expensive in development cost and component cost, but does not utilize the last available communication link in the cabin gateway.

Step 2 - Perform traditional valuation

The development activities needed for Alternative 1 are very few because an existing ECU is being used. The development cost of Alternative 1 is therefore considered to be zero. For Alternative 2 a new ECU needs to be developed. Using data from previous similar projects the development cost is estimated to be SEK 5 million (Swedish krona) for Alternative 2. The cash flow of alternative 1 is higher due to its low component cost. The results of the calculation are shown in Table 2. The difference in NPV between the two alternatives is SEK 6.9 million given the annual discount rate of 11%. The analysis of the valuation tells us to choose Alternative 1, but this does not take the value of flexibility into account.

Table 2 The calculated NPV of the two design alternatives in million SEK.

		Alternative 1	Alternative 2
Development cost:		0	-5
Cashflow	1st year	15,5	15
	2nd year	15,5	15
	3rd year	15,5	15
	4th year	15,5	15
	5th year	15,5	15
	NPV	57,3	50,4
	Difference:	6,9	

Step 3 - Find sources of flexibility

The communication link is a limited resource which can be of interest to a large number of functionalities, but those functionalities cannot be safely mixed with an external device. Alternative 2 thus gives a higher flexibility for future functionality than Alternative 1.

Step 4 - Estimate value of flexibility

Network communication is a limited resource within the automotive industry. Each network has a predefined maximum capacity and the utilization is also dependent on the physical location of the network cable. There is a growing market demand to monitor and control different vehicle functions through the use of external devices. To meet this requirement one must provide a way to connect external communication devices to the vehicle.

The expected value of the future function (underlying asset, S) is estimated to be SEK 10 million using the simplified model (5).

Step 5 - Estimate the price of flexibility

The exercise price SEK 2.9 million of finally implementing the function is an average of the potential functions found in the product portfolio. The exercise price includes the cost of ECU, sensors, cables, and developing application software.

Step 6 - Perform valuation using Real option

The communication link provides flexibility to the system and its value can be calculated using Real Options valuation. The product portfolio gives us a set of functionalities which could require the use of the communication link. The data needed is provided through an internal pre-study. The planned lifetime of the platform is 5 years.

The minimum goal of the investment in the alternative is to exceed the interest gained from the companies risk free interest rate (5%). The volatility is predicted to be 25% mainly due to the uncertainty of future demand. The up and down factors are given using Equation 2 and 3.

$$u = e^{0.25} = 1.28$$

$$d = e^{-0.25} = 0.78$$

The risk-neutral probability can then be calculated using Equation 4.

$$p = \frac{e^{0.05} - 0.78}{0.5} = 0.542$$

Given the underlying asset value (SEK 10 million) from the previous step the values can be calculated as shown in Figure 3. Inserting the values into Equation 1 calculates the current value of the option to SEK 7.7 million (see Figure 7).

$$A = (p \cdot 10.4 + (1-p) \cdot 5.4)e^{-0.05} = 7.7$$

Step 7- Compare the alternatives

Alternative 2 would be a sound investment if the value of the option premium is higher than the calculated difference (SEK 6.9 million) in Table

1. The option premium was calculated to SEK 7.7 million, which means that adding the flexibility is a good investment compared to the alternative without flexibility.

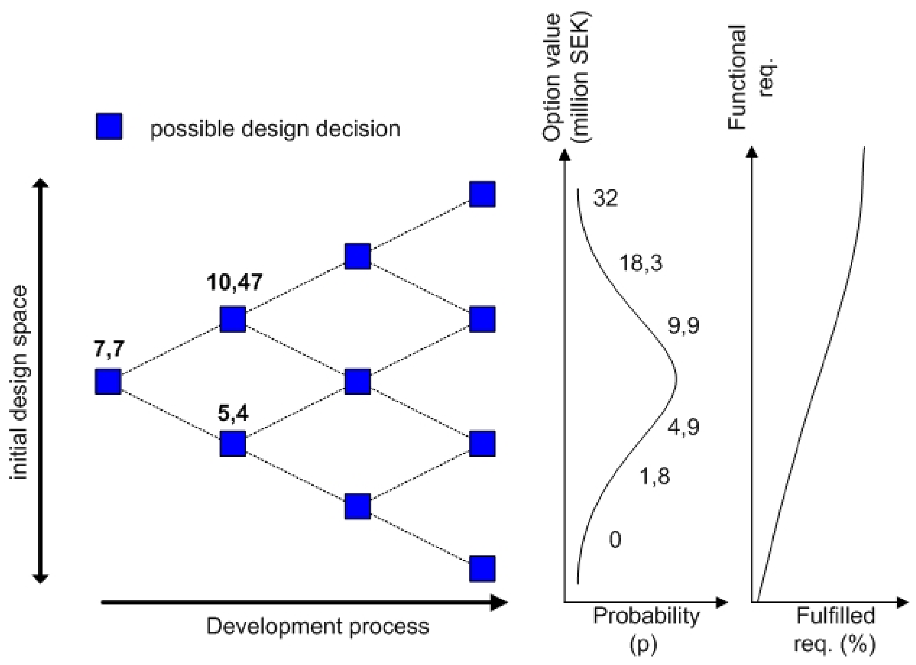

Figure 7 The future option value increases with the number of requirements implemented.

Step 8 - Make decision

The results show that the future option value increases with the number of requirements implemented (Figure 7). If only a low number of requirements will be demanded the value of the option will be lost. It also shows how the risk changes with the probability. This risk could be eliminated by not implementing the possibility to support a certain requirement. This would lead to a limited design space where an improved functionality cannot be implemented without a redesign of the system.

Discussion

The results show that investing in a flexible design would most likely be a sound investment if a large part of the future requirements were implemented during the system life cycle. The diversity of the proposed functionality makes it very uncertain what functionality will be implemented, which also is the reason why flexibility has a value. The prediction of the volatility and the value of the underlying asset are crucial to the results. One of the strengths when using real option valuation is that the uncertainty is taken into account and not left out of the calculation. It also provides a valuation method that can be used to analyze different future scenarios. Similar analyses can be done to estimate the value of future functions by iteration of sales volumes, customer price, etc.

Related work

Real Options is far from being the only method developed for valuing architectures. There are however only few methods that make an economic consideration, CBAM (Kazman et al. 2002) being an exception. Real Options is unique by also considering the flexibility and the architectural evolution over time (Bahsoon and Emmerich 2005). Our literature survey has found three research contributions that involve the usage of real options in system design involving software or hardware. None of them addresses embedded systems or the automotive domain explicitly.

Browning and Engel (2006) extend Real Options ”in” projects to architecture options and present a theoretical example where stakeholder overall value increases with 15% by designing the system for the right amount of adaptability. The framework presented shows a way to implement the optimal degree of flexibility. The initial research proposes using the model of Black and Scholes to calculate the value of the Real Options, but does not present a case. Browning and Engel show that architecture options provide the information to better predict the need for system upgrades and thereby increasing the lifetime value of the system.

Bahsoon and Emmerich (2003) use the concept of ArchOptions to value the stability and scalability of software architectures. ArchOptions are valued using the model of Black and Scholes and a replicating portfolio is therefore needed. The portfolio is valued by the requirements it supports during the operation of the software system.

Banerjee (2004) argues the need for flexibility and presents the solution of flexibility options compared to a fixed design. The value of the flexibility option is calculated using the binomial model that does not need a replicating portfolio and also supports American type options. The work done by Banerjee (2004) seems to be what best meets our prior stated problem definition.

Conclusions

This chapter has presented an evaluation process for practitioners using Real Options theory that enables analysis of both economic and engineering factors. It presents a possibility to put an economic value on system adaptability and could therefore support the design decisions in the early phases. Real Options provide the opportunity to analyze the cost of designing for future growth of a platform, based on the estimated value of the future functionality.

When developing an embedded system using Real Options each function would first buy the right but not the obligation to use the asset at a future date. The real option approach could, when fully developed, provide not only evaluation but also prediction of future needs.

References

Amram, M., and N. Kulatilaka. 1999. *Real options*. Boston, MA: Harvard Business School Press.

Axelsson, J. 2006. Cost models with explicit uncertainties for electronic architecture trade-off and risk analysis. In *Proceedings of the 16th international symposium of the international council on systems engineering*, Orlando, Florida, July.

Bahsoon, R. 2003. Evaluating software architectures for stability: A real options approach. In *Proceedings of the doctoral symposium of the 25th international conference on software engineering*, Portland, Oregon.

Bahsoon, R., and W. Emmerich. 2003. ArchOptions: A real options-based model for predicting the stability of software architecture. In *Proceedings of the 5th ICSE workshop on economics-driven software engineering research* (EDSER 5), Orlando, Florida.

Bahsoon, R., W. Emmerich, and J. Macke. 2005. Using ArchOptions to select stable middleware-induced architectures. In *IEE Proceedings on Software*, IEE Press 152 (4): 176–186.

Banerjee, P. 2004. Describing, assessing and embedding flexibility in system architectures with application to wireless terrestrial networks and handset processors. M.Sc. thesis, Massachusetts Institute of Technology, System Design and Management Program.

Browning, T. R., and A. Engel. 2006. Designing systems for adaptability by means of architecture options. In *Proceedings of the 16th international* *symposium of the international council on systems engineering*, Orlando, Florida.

Copeland, T., and V. Antikarov. 2001. *Real options: A practitioner's guide*. New York: TEXERE Publishing Ltd.

Cunningham, W. 1992. The WyCash portfolio management system. In *Proceedings of the conference on object oriented programming systems languages and applications*, 29–30. New York: ACM Press.

Florentz, B., and M. Huhn. 2007. Architecture potential analysis: A closer look inside architecture evaluation. *Journal of Software* 2 (4): 43–56.

Hoch, D., W. Huhn, U. Naher, and A. Zielke. 2006. The race to master automotive embedded systems development. McKinsey Company, Germany, Automotive and assembly sector business technology office.

Hull, J. C. 1993. *Options, futures, and other derivative securities*, 2nd ed. Englewood Cliffs, NJ: Prentice Hall International Editions.

Kazman, R., J. Asundi, and M. Klein. 2002. Making architecture design decisions: An economic approach. Technical report CMU/SEI-2002-TR-035. Software Engineering Institute, Carnegie Mellon University, Pittsburgh.

Larses, O. 2005. Architecting and modeling automotive embedded systems. PhD thesis, Dept. of Machine Design, KTH, Stockholm.

Leslie, K. J., and M. P. Michaels. 1997. The real power of real options. *The McKinsey Quarterly*, no. 3:4–22 (McKinsey & Company Inc.).

de Neufville, R. 2001. Real options: Dealing with uncertainty in systems planning and design. Paper presented at the 5th international conference on technology policy and innovation, Technical University of Delft, Netherlands.

Sullivan, K. J., P. Chalasani, S. Jha, and V. Sazawal. 1999. Software design as an investment activity: A real options perspective. In *Real options and business strategy: Applications to decision making*, ed. L. Trigeorgis. London: Risk Books.

Trigeorgis, L. 1988. A conceptual options framework for capital budgeting. *Advances in Futures and Options Research* 3:145–167.