Agile Development in Automotive Software Development: Challenges and Opportunities

Brian Katumba and Eric Knauss

Department of Computer Science and Engineering
Chalmers| University of Gothenburg
Gothenburg, Sweden
katsbriol@hotmail.com, eric.knauss@cse.gu.se

Abstract. In modern cars, most of the functionalities are controlled by software. The increased significance of software-based functionality has resulted in various challenges for automotive industry, which is slowly transitioning towards being a software centric industry. Challenges include the definition of key competencies, processes, methods, tools, and organization settings to accommodate combined development of software and hardware. Based on qualitative research, this paper aims at understanding the applicability of agile methods to automotive software development. Our explorative case study with one of the development sections at Volvo Car Cooperation identified challenges in their software development process related to process perception and reactive mode, multi-tasking and frequent task switching, individualism and lack of complete knowledge, as well as long communication chains and low cross-function mind set. Moreover it prepares a transition of software development at this multinational automotive company towards agile by relating agile principles and practices to automotive software process challenges.

Keywords: agile software development, automotive software development, software process improvement, embedded systems, challenges.

1 Introduction

Agile software development methods have changed the way software is developed in many domains. They promise better ability to cope with changing requirements, shorter time to market, and faster release cycles [1]. In contrast to earlier assumptions, agile principles have successfully implemented in large-scale software development and it has been reported that advantages of agile methods can be realized even in such environments [2–4].

In the domain of automotive software development, introduction of agile methods is hindered by the fact that software development has to be in sync with hardware development, which has been used as a strong argument to use a plan-driven approach. Yet, increased complexity and interdependency of automotive software challenges the ability to create an accurate plan upfront. In addition, automotive companies face an increased pressure to shorten their time to market and their release cycles. In this context, we performed a qualitative study with the following objective.

© Springer International Publishing Switzerland 2014

Research Objective: This paper aims at i) understanding to what extent agile methods are applicable to the software development at Volvo Car Cooperation (VCC) and ii) preparing the transition of software development at this multinational automotive company towards agile by relating agile principles and practices to automotive software process challenges.

We approach this objective based on a qualitative explorative case study with one development section in the powertrain department of VCC. The development section under investigation is characterized by doing part of the software development in house. We performed 29 semi-structured interviews with members of the development section and triangulated this data with internal documents.

Our main finding is that today, complexity of software, need to shorten release cycles, and pressure to cut development costs has led to a situation where plan-driven development starts to fail. Our interviewees mention critical challenges they encounter today, including process perception and reactive mode, multi-tasking and frequent task switching, individualism and lack of complete knowledge, and long communication chains and low cross function mind set. All of these challenges can be considered waste from the perspective of Lean software Development [5–7].

2 Agile Software Development and Related Work

2.1 Agile Software Development

Agile software development approaches emerged in the mid-1990s as a new solution to well-known problems experienced by traditional software development methods, including exceeding budgets, poor code quality and exceeded development schedules [6], [8], [9]. Agile methods intend to solve the persistent problems of traditional development by taking an iterative and incremental approach to software development. All agile methods have in common that requirements and solutions evolve through collaboration between self-organizing and cross sectional teams. Agility focuses on new ways of running business and casting off old ways of doing things, and the concept involves exploration, opportunity exploitation [10], acquiring new competencies, developing new product lines, and opening up new markets [11].

Agile development is guided by the agile manifesto, which declares the main values and purpose of agile software development in the form of the agile principles [12]. Although these principles are suitable and have been widely used for smaller software development organizations, evidence indicates that they can be adapted to large software-intensive organizations operating in complex global development environments [2], [4], [13]. Therefore these organizations are indeed in the process of deploying agile methods as part of their de-facto approach to software development, for example telecommunication companies, automotive industries, medical industries. Hence, in representing the values mentioned above, agile methods may very well pave the way for the future of software development also within large multi-national software organizations [14]. This paper aims at identifying how i) to check whether agile methods are applicable to the software development at Volvo Car cooperation and ii) to use these principles to transition software development at this multinational automotive company towards agile.

Among the agile software development methods, we discuss Scrum [15], eXtreme Programming (XP) [16], Lean Software Development [6].

2.2 Related Work

The study of agile software development is currently persuasive in software organizations [2], however its application and context for several years has often been to smaller organizations where development teams are involved in the product from development to product release [17]. While seemingly incompatible in embedded software development industry [18], introducing agile software development and agile practices is a challenge undertaken by many automotive industries producing embedded software. In this study we report the challenges of introducing agile software development practices in large automotive software development industry, where development involves the production of both hardware and software at the same time some of the requirements are realized by the suppliers.

Paasivaara et al. report the successful use of agile practices in large software projects [17]. Their case study is carried out on a 40-person development organization that is geographically distributed between Norway and Malaysia. They base their results on qualitative interviews to come up with a description on how Scrum practices were successfully applied by for example the use of teleconference and web cameras for daily meetings, frequent visits, unofficial distributed meetings, and annual gathering among others.

Abrahamsson explains the need of having a cost efficient development process in embedded software development [19]. He points out the increase of software in embedded devices that has resulted into challenges in the European industry. His ITEA agile project for application of agile processes in complex embedded systems concludes that the application of agile practices in embedded software development methods and process can reduce lead time and cost by 70% throughout the different industry sector [19]. Abrahamsson also explores the actual use and usefulness of Extreme Programming and Scrum in complex embedded software development [20].

The increasing use of embedded software in a wide range of products is causing a considerable impact to the society. Some of these products deal directly with human lives such as in automobiles, because of this their production need to go through rigorous process. Albuquerque presents in a systematic review [18] how agile methods have been used in the production of embedded systems. In addition they describe their benefits, challenges, and limitations to different industries including automotive industries.

Ulrik and Bosch present a set of factors that should be considered when implementing agile software development in mass produced automotive embedded software where development is governed by a stage gate process for hardware and product development as a whole is driven by a plan-driven process [2]. In addition they list agile measures to be considered by the original equipment manufacturer (OEM).

Besides applying agile in the actual automotive software development, agile has also been used to enhance supply chain management between suppliers and in-house development in the automotive industry [21].

It is worth noting that agile introduction comes with different perceptions in an organization. Dybå and Dingsoyr present an empirical study of agile software development [22]. As part of their results, they present the perception of agile development from a customer, developer and an organization perspective [22].

3 Background and Research Methodology

3.1 Research Site: Volvo Car Cooperation (VCC)

Volvo Car Cooperation is a Swedish automotive company operating on the global market. For this particular study, one development section was studied. This section is part of the department of Complete Powertrain Engineering within R&D. As part of the Volvo Car Corporation organization, the studied section contributes to system development of complete powertrains, developing and optimizing of the powertrain control system and assessing the powertrain attributes in VCC's vehicles.

In specific, the studied section delivers SW and electronic HW to VCC production line, after market, and to different departments at VCC for development purposes. The section develops parts of the software in house (focus in this paper) and specifies software requirements for other parts that are then developed by a variety of suppliers. At the development section, most of the software is developed in MATLAB's Simulink in addition to following MISRA-C/C++ guidelines. To confirm that a supplier delivers what the specification stated, VCC does the validation and verification of the software. The software developed is embedded in nature and is massively produced.

Because of the complexity of the system development within the car industry, this section has seen a need to investigate more effective and reliable methods of system development. In every new car model, manufacturers are introducing new advanced functionality due to customer demands and competition from the market. Often, suppliers delegate development tasks to sub-suppliers that are specialized on certain features. This considerably adds to the complexity of integration, making software development in the automotive industry even more complex in terms of development, time and cost.

By transitioning to agile, the development section in our case study aims at further improving the end-to-end process flow. Specifically the goal is to cut lead times, to be more customers focused, and to make better use of limited resources. At the time of initiation of this study one of four groups at the section had unsuccessfully tried using Scrum for in-house software development. Motivated by this experience, we conducted our case study to systematically explore the possibilities of adopting agile practices at the section.

3.2 General Overview of Current Process

The development section under research consists of about 60 engineers who are divided in four groups. The first group is responsible for the architecture &

non-propulsive control, while the remaining three groups are responsible for the engine control modules and transmission. For software development, there are various roles involved at the section. Intentionally, the specific names of the roles are left out but rather the roles are categorized in seven categories as shown in Table 1; managers, software responsible, test engineers, software coordinators, trouble shooter (not part of this study), system responsible/ architects, system developers/designers, Hardware responsible (not part of the study), and internal software responsible.

Table 1. Roles and Responsibilities

Roles	*Experi-ence(Years)*	*Tasks and Responsibilities*
Manager	> 10	Spearheading group activities and project related communication
System Responsible/ Architects	>10	Architects have the key role to control architectural part of the model. The architect is responsible for all elements in the architecture and communication between the elements. This involves both hardware and software architecture
Software responsible	2 <= 7	Responsible for both in-house and supplier software production, software deliveries, Technical questions of the project, Project Planning
SPM(Software Plugin Module) Developer	2 <= 8	Responsible for SPM(Software Plugin Modules) development(in-house software development)
Software Coordinator	5 <= 9	Responsible for controlling and coordinating software versions and releases, Compiling software together as well as managing change orders, uploading software to DTECS- Development Tool for Embedded Control Systems, and handling special deliveries internally
Tester	2 <= 7	Verify and validate software on both component and system level, software calibration

The development section follows a traditional stage-gate process (V-Model), where the gates are driven by decisions and investment in the manufacturing of the product, i.e. driven by the hardware. The gates progress as the project grows and their progression corresponds to software artefacts such as user requirements, system requirements, software architecture, component requirements, and software implementation. At this development section, the big part of software requirements comes from the electrical department as a result of system engineering work while the other requirements come from other departments, e.g. system safety, engine and transmission department, and legal department. Requirements are collected and documented in a special purpose tool (Elektra), which acts as a requirement repository. System architects refine these requirements and break them down to support the decision on which requirements are developed in house and which requirements are sent to the suppliers or subcontractors.

There are three different development paths a requirement can take: i) hardware developed by a supplier, ii) software developed by a supplier, and iii) software developed in house. In case i) and ii), the supplier decides which development process to follow, but for software process improvement, the supplier is required to use one of the following software process improvement models: ISO/IEC15504, Automotive SPICE, and CMM/CMMI. In case iii), a traditional V-model approach is followed according to the architects.

In this study various roles at the development section where interviewed (see table 1), and based on the results from the interviews we came up with different development challenges.

3.3 Research Method

We investigated the applicability of agile methods based on an exploratory qualitative case study [23]. In this exploratory research, the researchers studied the current development method at VCC development section and then sought new insights and generated new ideas and hypotheses for the study.

By definition, a case study is "an investigation of a contemporary phenomenon in depth and within a real life context, where the boundaries between the phenomenon and the context are unclear" [24]. With the aforementioned definition, the study is based on real life experiences of the researchers at Volvo Cars' Power Train section. It was unclear if agile methods could be applied at this section; therefore it was the purpose of the research to bring out the clarity of the phenomenon and context for the section in question.

A case study method has been chosen because identifying a suitable development process requires an in-depth investigation of the current process from the beginning to the end to understand the underlying principles and the problem that may be involved. In addition, Yin suggests that case studies allow investigators to retain the holistic and meaningful characteristics of real life events such as individual life cycles, group behaviors, organizational and managerial processes which can be augmented to fit the domain of this research [24]. Similarly, Andersson and Runeson argue that case studies in the software engineering discipline often take an improvement approach, similar to action research [25]. The purpose of this study is to improve the working process at the section under question.

According to [24], there are six main sources of data: documentation, archival records, interviews, direct observations, participant observations and physical artifacts. For this particular study, semi-structured interviews were the primary source of data and complemented by company presentations, company documentations and literature reviews as the secondary data source to allow triangulation of results. Interviews where transcribed and coded by highlighting and labeling important parts. We then grouped resulting codes into themes and derived opportunities and challenges in workshops. While we used all collected data to derive our conclusions, we can only partly disclose it to protect the company's sensitive data.

3.4 Research Setting

During the study, and as a significant starting-point for acquiring an insider's view of the research phenomenon, one of the authors of this paper spent three or four days every week during a four-month period at Volvo cars' site involved in this study. In supporting engagement between researchers and research subjects, this in-depth study as well as the observational studies that were carried out as part of it, were important impetus for developing an understanding of the research setting. While observational studies, and the documentation of these, were the main activity during this time, data sources such as meeting minutes and organizational documents were also used to get an enhanced understanding of the development teams and the development unit in which they operate.

As a starting point, 2 project managers introduced a vague project topic to the researchers that showed a need to become agile at the development section. The researcher had to narrow down the topic to make it feasible in the short time that was available. Never the less, it was not so easy since many things had to be put in consideration i.e. the time frame available, the nature of the organization and resources at the section.

The researchers took observational studies at the premises and, 29 semi-structured qualitative interviews[1] were conducted as the primary data source. During these interviews, questions focused around areas such as; 'requirements', 'Roles and competence', 'communication', 'process and phases', agile understanding, 'quality improvement' and 'co-location'.

In focusing the interviews around areas that had been identified as important by the researchers, there were reasons to believe that the research would attract attention and that the managers were even more motivated to participate actively, and on a continuous basis, in the study. The aim of the interviews was to reach an understanding of the current process at the section, to check for the suitability of agile development, and to initiate a new working method based on agile principles.

As in exploratory research, the findings generated emerged as an iterative process between theoretical conceptions and empirical data [26]. In accordance to this, collection and analysis of empirical data was undertaken as a concurrent activity, with an important part of the analysis conducted also after the empirical work. The initial conceptual apparatus – encompassing certain assumptions, beliefs, and rationale – transformed over time. Thus, our notions, our empirical data and the transformations of our interpretations of this worked as entangled elements in the process of analyzing the case.

4 Results: Process-Related Challenges in Relation to Agile

In this section, we present the results from our qualitative study and relate the finding to the research objectives: Firstly, to check whether agile methods are applicable to

[1] See interview guide at `https://dl.dropboxusercontent.com/u/13255493/Katumba-Agile_in_Automotive-Profes14-Interview_guide.pdf`

the software development at Volvo Car cooperation and secondly to use these principles to prepare for the transition of software development towards agile. Specifically, we collected opportunities for agile methods, e.g. challenges that can be addressed or overcome by an agile method work in this context, and challenges for agile methods, i.e. characteristics of automotive software development that make it hard to introduce agile methods. We categorized these opportunities and challenges in five categories: *Process ability*, *workload management*, *domain specifics and supplier network*, *working context*, and *culture of sharing information and knowledge*. For each category, we include the perspective of different roles at the development section.

4.1 Process Ability

As discussed in our research background, development is supposed to follow the V-Model, as clearly indicated by introductory interviews from two managers and internal training documents. However, our interviewees indicated a lack of structure when it comes to developing software in house. Tasks are started as they hit the engineer's desktops and development seems to be driven by sudden urgencies rather than by a long term plan. One interviewee explicitly noted this *reactive mode* approach by saying:

> "[For developing software in house,] *There is no working process at the section, it may be there but I have not used it nor do I know how it looks like. What I know are the milestones and when I am supposed to deliver*"
> "*We had a process but we stopped using it since it required a lot of resources and hence it lacked the practicability of the project*",

This *perceived lack of an accepted process* by our interviewees is related to the fact that each member may have more than three or four roles. Not having a clear methodology to follow causes confusion and is seen as a challenge by our interviewees:

> *"…we are in a confusing situation and nothing […] works in this confusing state of working. If this agile thing works out, you would have [helped us a lot]."*

We assume that the perceived lack of structure is the consequence of VCC aiming at increasing their flexibility with in-house software development. Agile methods might be helpful in this situation by adding just enough structure while still offering flexibility.

4.2 Workload Management

Heavy workloads is one of the main challenges faced at the VCC section, as supported by all interviewees irrespective of their roles. For example, the role of the *software responsible* was merged with another role in order to improve development efficiency. Initially, this role was to work on project planning activities, cost estimation, and supplier communication. Today, this role is also responsible for answering technical questions, prioritizing requirements for both internal and supplier software, breaking requirements down, as well as ensuring accuracy of test and calibration results. This *multitasking* is seen as challenging by our interviewees and it affects other roles as well. For example in internal software development, one engineer is responsible for all phases of development, including requirements, design and testing. This leads to a loss of focus of developers and ultimately can affect the quality of the feature.

A related challenge is *task switching*: people are participating in many projects at the same time and often need to jump from one task to the next, which might be in the context of a completely different project. A resulting problem mentioned in our interviews is the lack of time for continuous learning. Specifically, the testers at the section claimed to serve many software responsible with a ratio of 1:3 i.e. one tester can test software from three software responsible with each software responsible working on more than three projects. This results in *unbalanced workloads* and hinders the performance of the group members. In addition, *schedule synchronization* becomes a problem, because engineers are working on many projects, on multiple tasks, which they frequently switch. Synchronizing schedules of team members in this context becomes challenging, as indicated by the following quote from our interviews:

"*.. you may be in one meeting yet at the same time you needed at another meeting….*"

As agile methods rely to a large extent to oral face-to-face communication, such context switching might be problematic when introducing agile methods.

4.3 Domain Specifics and Supplier Network

The domain of automotive software development involves both in-house and supplier software development and VCC is no exception to this setting. At VCC, the big part of software requirements come from the electrical department as a result of system engineering work while the other requirements come from other departments; system safety, engine and transmission department and legal department. They usually collect and gather requirements in a tool called Elektra, which acts as a requirement repository. These requirements are further broken down at the section, of which some are developed in house while other requirements are sent to the suppliers or subcontractors. It is however important to note that these requirements may be hardware, internal software or supplier software requirements. For supplier requirements, the supplier decides which development process to follow, and is required to produce working software fully tested and integrated with the in house software. For software produced internally, according to the architects it has some elements of following a traditional V-model where specifications are done first, before design, integration, and validation. The domain of the network of in-house, supplier and hardware results into *inventory and motion*. In this, all requirements sent to the supplier are fully managed by the supplier till integration. This means if anything is missing or not done even if the in house team can work on it, the software is sent back to the supplier, the interviewees mentioned this *rework* could result in bottlenecks caused by *process dependencies*, if the software is needed for further activities to take place. One of the software responsible mentioned:

> "*We keep sending back the software modules to the supply in case there is something missing, even if it is something small which we can fix internally.*"

While this is a perfectly normal workflow in traditional automotive development, it seriously impacts the effectivity of (agile) in house development.

4.4 Working Context

In this category, we discuss challenges in the context of knowing what a function or feature should do. The interviewees claimed that at times they do not know the *context* of the functions or features they are developing, mainly due to the fact that they do not participate in the requirement elicitation phase of the project. This is a consequence of the multitasking challenge above, as one interviewee indicates:

> "*There are many projects running at the same time, meaning that each person is participating in more than one project*".

An example, one system responsible claimed to be participating in four projects at the same time. In this situation, engineers tend to focus on the projects close to deadline instead of working in projects that are in early stages. Also, this leads to a lack of competence and knowledge. For example, a system architect might only have a helicopter view of a function, whereas a tester is limited to knowledge about testing activities. This *separation of concern* and *lack of end-to-end knowledge* was perceived as problematic in the context of agile in house software development.

Challenges around working context are intensified when *requirements are vague* and not easy to understand. An example given during an interview was "*the engine should run fast and smooth*", leaving it to the developer to find out what smoothness is and at what level of smoothness needs to be reached.

Finally, the *ramp-up* challenge relates to bringing new recruits up to speed. Because of the complexity and lack of structure of the current development process, new recruits take long time to get to know their ways. The high workload means that they get limited support from the old 'group' members, who are always busy working on their tasks. This further reinforces individualism, where instead work should aim to achieve a common goal. Also, problems like lack of continuity and low knowledge about the features and their functions are a consequence of this.

4.5 Culture of Sharing Information and Knowledge

The different roles at VCC section under consideration are challenged with sharing information and knowledge. This challenge is mainly caused by their working culture that encompasses on individuals to achieve a common goal. Although each role is placed in a group, when it comes to actual work, the group influence is minimal. In addition, the chain of communication at times is long since information has to pass through different channels to reach a person who is really going to use it. For example, if a tester wants to clarify unclear requirements, information has to go through the software responsible, to the architects, to the electrical department and so on. This in the end leads to low continuity in development since by the time the information comes back to the person who initiated it, it may be late as that person may be already engaged with other activities in a different project. This again explains the problem of participating in more than one project as already mentioned.

5 Discussion

In this section we discuss the themes we found in the qualitative interview study and outline agile practices that can be adopted in automotive. For this, we map each theme to a set of relevant agile methods, principles, and practices. From our interviews, it is clear that the division under investigation is facing challenges that prevent them to fully leverage the potential of developing software in-house. The opportunities and challenges we found were grouped into themes: process related challenges, workload management, domain specifics, working context as well as information and knowledge sharing. In Table 2 we summarize opportunities and challenges for agile methods at the company and suggest agile practices that can be adopted.

The process related challenges seem to be caused by the nature of the organizational structure, which focuses more on the finished product than on the way the product is developed. By this all interviewees were less concerned about the process but instead more concerned on what they have to deliver.

Moreover, the organizational setting and the available competences are characterized by low agile knowledge as well as low general software development knowledge in comparison to the excellent knowledge in hardware development. We triangulated these results with responses to the agile questionnaire from interviewees and additional managers that showed that, although the teams know the term “agile”, they lack the full context of it. This can be explained that the automotive industry has traditionally been characterized more by hardware production than software [2]. Relating this challenge to the agile methodology, it can be mitigated by having a flexible, holistic product development strategy as for example in Scrum [27]. With Scrum, there is a defined product strategy and structure, which accommodate changes at higher level at the same time leaving room for flexibility and innovativeness. Besides, the process ability challenges can also be explained by the domain specifics and supplier network. Automotive companies produce cars in-house by integrating their suppliers’ deliveries of hardware and software. This means there should be a strategy that can accommodate both the in-house process and the supplier process to reduce the inventories and motions involved. From a lean perspective, the unnecessary motions are referred to as waste and so lean calls for an absolute elimination of waste in the production process [6], [7].

Table 2. Automotive challenges and opportunities for agile in-house software development

Theme	*Opportunities for agile in automotive*	*Challenges for agile in automotive*	*Agile Practices*
Process ability	Perceived lack of software development methodology and structure	Low agile competence, Reactive mode	Flexible, holistic product development strategy[1],
Workload management	Heavy workloads, Unbalanced workloads	Multi-tasking, Task switching, Schedule synchronization	Task boards[1,2] Sprint planning[1], Commitment phase[2] Defer Commitment[3] Frequent releases, short development cycles[2]
Domain specifics and supplier network	Inventory and motion, Rework	Process dependencies	Eliminate waste[3]
Working context	Vague requirements, Lack of end-to-end knowledge	Feature context, Separation of concerns, Ramp-up	Requirement Prioritization[1,2], Emergent Design and Metaphor[2] Product Backlog[1] User stories/ Product vision[1,2] Sprint Planning[1] System metaphor[2] Relying on a product owner[1]
Culture of sharing information and knowledge	Low knowledge sharing, Individualism, No team work	Long communication chains, Low cross function mind set	Retrospective[1,2,3] Cross function Teams[1] Self-organizing teams by encouraging colocation of all team members[1,2] Daily stand up[1], Pair programing[2] Continuous learning[3]

NOTE: 1 – Scrum, 2- Extreme Programming, 3- Lean

The heavy workloads mentioned in the interviews by almost all interviewees have resulted in multi-tasking, task switching, and poor schedule synchronization. These challenges indeed affect production since they are bottlenecks to the development chain. Management also confirmed these effects: the nature of the organization is set that way. A similar case is also discussed and experienced in another multinational cooperation in the telecommunication sector where task switching was one of the bottlenecks at their development section [4]. Looking at some of the agile practices that can be adopted to solve this issue were; using task boards to show the work to be done [28], sprint planning to solve product planning issues [29] and having short iterations that can result into frequent releases [20].

The working and feature context showed that teams sometimes lack the domain knowledge of the products they are working on. This was however explained by the vague requirements they get from the requirement engineers who mainly have hardware and electronics knowledge than software knowledge. In addition the separation of concerns where for example architects are doing only architectural work and testers do only the testing, is also seen as a cause of low domain understanding. Ramp up was also mentioned and refers to the fact that new comers find it difficult to find their way around development. This can be related to the process ability challenges as well. Working context challenges are suggested to be solved by having a product backlog for requirements [27]. The requirements in the backlog should be prioritized and each requirement should have a user story that explains what the requirement should do [28]. Moreover in the development, the use of design metaphors are encourages and relaying a competent product owner to share the knowledge of requirements [2].

The culture of sharing information and knowledge is something that is vital in software development[30]. It is in this that teams learn new tools, know each other, and to improve the process as well as innovativeness [4], [3]. However at the development section it was affirmed that there is low knowledge sharing, high level of individualism and not much teamwork. This results in long communication chains where people take long to communicate or provide feedback to those who need it. The lack of teamwork can be explained by teams having a low cross-functional mind-set and can result into individualism where everyone is concerned about him/herself to do the work. We suggest retrospectives, a self-organized cross-functional team setting, team colocation, working in pairs, and daily stand-ups to cub these challenges.

All the challenges mentioned were validated by the management at the section and are seen to overlap each other. The process related challenges could be the cause of low domain understanding at the same time can be argued to be the cause of workload related challenges. On the other hand the domain specific and supplier network challenge can be the cause of process related challenges, which can explain the working context challenges. And also having a complex development structure can explain why there is low knowledge sharing and no team interactions. In other words, these challenges are intertwined and one can result into the other.

6 Conclusion and Outlook

In this paper we presented our findings from a qualitative study with one of the development sections at Volvo Car Cooperation. Challenged by its transition towards being a software-centric company, which shows in the increased need for in-house software development. The department is looking for ways to improve their software development processes and decrease their time to market. Based on semi-structured interviews, we discovered specific challenges with the current way of developing their software. We discuss the applicability of agile methods to these challenges. For VCC, this study can serve as a first step towards transition to agile methods. Future work should focus on quantifying and measuring the current challenges so that potential improvements from switching to agile methods can be proven. We hope that others find our insights useful for understanding challenges that arise from software and software development pervading more and more products.

References

1. Cockburn, A.: Agile software development: The cooperative game, vol. 113, pp. 2000–2001. Addison-Wesley (2001)
2. Eklund, U., Bosch, J.: Applying Agile Development in Mass-Produced Embedded Systems. In: Wohlin, C. (ed.) XP 2012. LNBIP, vol. 111, pp. 31–46. Springer, Heidelberg (2012)
3. Holmström, H.O.: Acting Agile in Streamline Development. Inf. Syst. Res. Semin. Scand. (2009)
4. Katumba, B., Antanovich, A.: Bottlenecks in the Development Life Cycle of a Feature- A case study conducted at Ericsson AB. In: 7th Annual International Conference on Computing and ICT Research, pp. 472–490 (2011)
5. Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison Wesley, Boston (2003)
6. Shalloway, A., Beaver, G., Trott, J.R.: Lean-Agile Software Development, Achieving Enterprise Agility. Addison Wesley, Upper Saddle River (2010)
7. Womack, J.P., Jones, D.T., Roos, D.: The Machine that Changed the World: The Story of Lean Production, pp. 1–11. Harper Collins, New York (1990)
8. Hafterson, T.: Incorporating Agile Methods into the Development of Large-Scale Systems. In: UMM CSsci Senior Conference, Moris, MN
9. Salo, O., Abrahamsson, P.: Agile methods in European embedded software development organisations: A survey on the actual use and usefulness of Extreme Programming and Scrum. IET Software 2(1), 58 (2008)
10. Yusuf, Y.Y., Sarhadi, M., Gunasekaran, A.: Agile manufacturing: The drivers, concepts and attributes. Int. J. Prod. Econ. 62(1–2), 33–43 (1999)
11. Dismukes, J.P., Uppal, M., Vonderembse, M.A., Huang, S.H.: Designing supply chains: Towards theory development. International Journal of Production Economics 100(2), 223–238 (2006)
12. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D.: Manifesto for Agile Software Development. The Agile Alliance (2001), http://agilemanifesto.org/ (accessed: May 30, 2014)
13. Harrison, R., West, A., Lee, L.: Lifecycle Engineering of Future Automation Systems in the Automotive Powertrain Sector. In: 2006 IEEE Int. Conf. Ind. Informatics, pp. 305–310 (August 2006)
14. Abrahamsson, P., Warsta, J., Siponen, M.T., Ronkainen, J.: New directions on agile methods: A comparative analysis. In: Proceedings of the 25th International Conference on Software Engineering, pp. 244–254 (2003)
15. Schwaber, K., Beedle, M.: Agile Software Development with Scrum, vol. 18(9), p. 158. Prentice-Hall (2001)
16. Beck, K.: Extreme Programming Explained: Embrace Change, p. 224. IEEE (1999)

17. Paasivaara, M., Durasiewicz, S., Lassenius, C.: Distributed Agile Development: Using Scrum in a Large Project. In: 2008 IEEE Int. Conf. Glob. Softw. Eng., pp. 87–95 (August 2008)
18. Albuquerque, C.O., Antonino, P.O., Nakagawa, E.Y.: An investigation into agile methods in embedded systems development. In: Murgante, B., Gervasi, O., Misra, S., Nedjah, N., Rocha, A.M.A.C., Taniar, D. O., Apduhan, B.O. (eds.) ICCSA 2012, Part III. LNCS, vol. 7335, pp. 576–591. Springer, Heidelberg (2012)
19. Abrahamsson, P.: Speeding up embedded software development. ITEA Innov. Rep. (2007)
20. Salo, O., Abrahamsson, P.: Agile methods in European embedded software development organisations: A survey on the actual use and usefulness of Extreme Programming and Scrum. IET Software 2(1), 58 (2008)
21. Tarokh, M.J., Ghahremanloo, H., Karami, M.: Agility in Auto Dealers SCM. In: IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI 2007, August 27-29, pp. 1–6 (2007)
22. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: A systematic review. Inf. Softw. Technol. 50(9–10), 833–859 (2008)
23. Robson, C.: Real world research: A resource for social scientists and practitioner-researchers, vol. 2, p. 624. Blackwell (2002)
24. Yin, R.K.: Case Study Research: Design and Methods, vol. 5(5), p. 219. Sage Publications (2009)
25. Andersson, C., Runeson, P.: A spiral process model for case studies on software quality monitoring method and metrics. Softw. Process Improv. Pract. 12(2), 125–140 (2007)
26. Klein, H.K., Myers, M.D.: A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. MIS Q. -Spec. Issue Intensive Res. Inf. Syst. 23(1), 67 (1999)
27. Julian, B.M.: Scrum Master Activities: Process Tailoring in Large Enterprise Projects. In: 2014 IEEE 9th International Conference on Global Software Engineering (ICGSE), August 18-21, pp. 6–15 (2014)
28. Guang-yong, H.: Study and practice of import Scrum agile software development. In: 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN), May 27-29, pp. 217–220 (2011)
29. Schwaber, K., Sutherland, J.: The scrum guide (October 2011)
30. Sekitoleko, N., Evbota, F., Knauss, E., Sandberg, A., Chaudron, M., Olsson, H.H.: Technical Dependency Challenges in Large-Scale Agile Software Development. In: Cantone, G., Marchesi, M. (eds.) XP 2014. LNBIP, vol. 179, pp. 46–61. Springer, Heidelberg (2014)