Barriers and Enablers for Shortening Software Development Lead-Time in Mechatronics Organizations: A Case Study

Mahshad M.Mahally
Department of Powertrain
Software Engineering Division
Volvo Car Corporation, Sweden
+46-31-325 6902
mahshad.mahally@volvocars.com

Miroslaw Staron
Computer Science and Engineering
Chalmers | University of Gothenburg
Gothenburg, Sweden
+46-31-772 1081
miroslaw.staron@cse.gu.se

Jan Bosch
Computer Science and Engineering
Chalmers | University of Gothenburg
Gothenburg, Sweden
+46-31-772 5716
jan.bosch@chalmers.se

ABSTRACT
The automotive industry adopts various approaches to reduce the production lead time in order to be competitive on the market. Due to the increasing amount of in-house software development, this industry gets new opportunities to decrease the software development lead-time. This can have a significant impact on decreasing time to market and fewer resources spent in projects. In this paper we present a study of software development areas where we perceived barriers for fast development and where we have identified enablers to overcome these barriers. We conducted a case study at one of the vehicle manufacturers in Sweden using structured interviews. Our results show that there are 21 barriers and 21 corresponding enablers spread over almost all phases of software development.

Categories and Subject Descriptors
D.2.9 [Software Engineering]: Management, D.2.13 [Software Engineering]: Reusable Software

General Terms
Performance, Design, Standardization, Verification

Keywords
Lead-time, V-model, automotive software

1. INTRODUCTION
The increasing number of electronic control units in cars leads to the growing complexity of software components. This, in turn, leads to long development lead times. The software used in components is developed by both supplier and car vendors, often in parallel. However, a modern trend in automotive software development is to develop software in-house (i.e. by the car manufacturers themselves) or at least increase the number of ECUs for which the software is developed in-house. This allows for full control over the development, to keep specifications confidential and to have the same behavior of common functionality in all projects despite different suppliers. In order to reduce the software lead time that is necessary to consider the distinctive structure of mechatronics organizations which does not match completely the software structure. There are different approaches used in software companies to improve the software development process, but in order to find the method which is applicable for a mechatronics organization we need to first find the specific barriers that prevent us from achieving high development speed. Once these are identified we can identify the enablers (usually modern software development techniques) which can overcome these barriers. Based on the above we explore the following research question in this paper:

What are the main barriers and enablers for shortening the software development lead-time in a mechatronics organization?

In this research project, we explore software development process considering the upper layer of production including design and verification of functions, systems and components. The contribution of this paper is a list of barriers and related enablers which could be added to the current development process. We also outline related levels for each barrier in the current software development V-model.

2. ORANIZATIONAL CONTEXT
Case Descriptions: *Volvo Cars* is a manufacturer of cars and has a long-term tradition of developing integrated mechanical, electrical/electronic and software systems. Volvo Cars is a Swedish car original equipment manufacturer (OEM), based in Gothenburg. Volvo Cars was developing software and hardware in a distributed software development environment. For a number of Electronic Control Units (ECUs) both in-house software development teams develop the software and external suppliers who design, implement and test the functionality based on specifications from Volvo Cars [1].

3. RESEARCH METHOD AND STUDY DESIGN
In this research we performed a set of individual interviews with 12 people. Interviewees were from Volvo Cars with at least 5 years' experience from automotive industry and various roles involving commodity purchaser, technical leader, technical expert, manager, system architect, quality assurance, integrator, system developer and system designer. Interviews were conducted in English and Swedish and lasted for approximately 1 hour. In addition to the interviews, we used documentation review and field notes as complementary data collection methods. The documents included software development documents and project

Table 1. Barriers at the Function Definition Level

	Barriers (B)	Enablers (E)
1	**Project-based development strategy:** misalignment between product and project structure makes the visualization of product development progress difficult	**Product-project alignment**: structure projects based on the developed products
2	**Inadequate early specifications:** incomplete specifications lead to late changes, which can result cost-overrun	**Early providing specification**: have a quality review of the specification prior to start of function design and involve development team in the review process
3	**Long response time from supplier**: time-consuming and inefficient process between requesting an offer from supplier and setup the final agreement	**Start initial communication at high level:** establishing a link to the supplier at higher management level to emphasize the importance of starting the project
4	**Running research-development projects at strategy phase:** extend the time of concept phase and postpone the start of industrialization	**Running research-development projects parallel with concept & industrialization phases:** shortening the time of concept phase and reducing the lead time

management documents. Identified enablers in this paper are based on the interviews and own experience and we will work in the future to find more enablers.

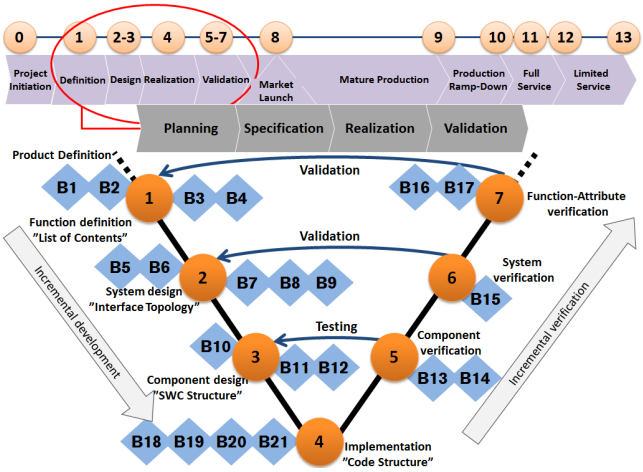

Figure 1. Barriers and Enablers in the V-model

Findings & Analysis: In this section, we outline the identified barriers in product life cycles that has an effect on software lead-time. Moreover we propose corresponding actions (enablers) that could be adopted by current process for resolving them. The results from this research are outlined in figure 1 where an overview of production life cycles (levels 0 to 13) and barriers (blue diamonds) for shortening software lead times are illustrated individually for each of the stages in V-model. In order to gain a better understanding we classify identified barriers for each individual level in separate tables.

3.1 Barriers at the Function Definition Level

The barriers are listed in table 1 and related to the processes of definition functions and how the projects are structured. The main enablers to decrease the lead-time for this phase include the alignment between project and product structure as well as the ways in which communication is structured.

3.2 Barriers at the System Design Level

The logical view of entire system irrespective to hardware is provided at this level. The main barriers at this stage are listed in table 2. They are allied to inadequate requirements provided at function definition level, incomplete interface between nodes and the requirements that are generated without considering the software architecture. The main enablers include improving

Table 2. Barriers at the System Design Level

	Barriers (B)	Enablers (E)
5	**Undefined interfaces between system nodes:** wrong sub-function implementation get provided at lower levels in the process due to incomplete specification provided at system design	**Enhancing the communication between node owners:** review the requirement and provide complete software component description
6	**Large and growing complexity at system level:** incomplete system design and time consuming process for handling the requirement at implementation level	**Reduce complexity/Separation of concern:** apply the axiomatic design theory (ADT) [2], decomposing of complex functions to sub-functions, increasing modularity and reuse.
7	**Unclear and incomplete order specification:** extra cost and effort at later stages to correct provided requirements provided at function level	**Review of function requirement:** frequently review process to secure that the system architecture has the right interpretation of the function list.
8	**Varying procedure at system level between groups:** wrong software requirement and immature software specification increase overall lead-time and decrease quality of the system	**Improve the quality of requirement:** using requirement management framework [3] to deal with the tailored requirement engineering process [4].
9	**Shortage in non-functional requirement:** difficulties to implement non-functional requirement (i.e. security or efficiency) which does not fulfill subsystem framework	**Using Object Oriented analysis model:** contribute to provide a clear picture of system and reduce the number of downstream errors in maintains.

Table 3. Barriers at the Component Design Level

	Barriers (B)	Enablers (E)
10	**Complex/Unstandardized specifications for supplier:** entails additional cost and risk for exceeding the delivery deadline from the supplier	**Review unstandardized specifications:** individual evaluation for each case to confirm the needs, providing alternative solution which requires less reconstruction of existing components
11	**Several software tracks for various projects:** difficulties to design the components due to inappropriate structure of Sub-systems	**Using common baseline for software, reuse strategies:** identify the functions with common behaviors and classify them in one sub-function in order to enhance the system architect
12	**Incomplete and incompatible specification of sub-system provided at system level:** difficulties to provide a complete software and hardware requirement when the software component structure does not fulfill the interfaces defined at system	**Apply Ontology based knowledge management system:** consider the component restriction by designing the sub-functions [5] provide a clear picture of sub-systems and enhance transparency of detailed structure, thus improving system architect

communication levels between node owners, applying axiomatic design theory (ADT) to reduce the complexity [2], decomposing of a complex function to sub-functions which can be maintained independently [6] and perform the separation of concern principles at architect level to increase modularity. This is achieved by deviation of the system into distinct features with minimization of interaction points [7]. Another enabler is improving the quality at requirement level through using approaches such as requirement management framework [3].

3.3 Barriers at the Component Design Level

As listed in table 3, the key barriers of this level are related to complex structure of sub-systems, software implementation and component structure which does not fulfill the interfaces defined at system level. The main enablers are improving the communication between purchasing with technical group at OEM and supplier and enhancing the system architect.

3.4 Barriers at the Component Verification

The main barriers at this level are associated to supplier and hardware issues and listed in table 4. The key enablers are improvement of relationship to supplier and choosing the right supplier for component at beginning of project, evaluating the products delivered from the suppliers and providing information required to prepare accurate cost estimation for the whole project.

3.5 Barriers at the System Verification Level

Significant barrier at this level is mentioned in table 5. It is linked to misalignment of release plan between various software development teams. It entails to delay in verification and high risk to find critical bugs due to the short testing period. Related enabler is having synchronized time plan between electrical and software development for harmonizing the release plans and reducing deviation in verification plan. All parties should approve this plan before starting the main development.

3.6 Barriers at the Function Verification

Function verification includes testing of the final implemented solution in a car and approving both the software and hardware that meet the customers expectation according to the functional list provided at the beginning of project. Table 6 shows a list of barriers which are related to sourcing plan and late verification. The number of resources allocated in project and the amount of implementation and verification increased as we come closer to the conclusion of the project. This results in finding number of issues in the ending phase of the project. Enablers related to this level are performing review on specification at each engineering step and applying a new resource planning management.

Table 4. Barriers at the Component Verification Level

	Barriers (B)	Enablers (E)
13	**Repetitive and time-consuming design and product verification:** deviation in project time plan due to finding issues which require hardware's modification	**Review approval for specification and agile verification process:** request more frequent delivery from supplier to perform early hardware verification and accurate review of requirements between supplier and OEM
14	**Late evaluation of supplier solution:** difficult to predict issues related to supplier until the project reaches late stages such as component verification phase, risk for delay in project time plan	**Multiple suppliers strategy and using supplier management techniques :** start the project parallel with at least two suppliers for a short time of period to encourage competition between suppliers and raise the level of performance [8], performing iterative development in a supplier development approach

Table 5. Barriers at the System Verification Level

	Barriers (B)	Enablers (E)
15	**Dependency to Signal Database (SDB) releases to verify software:** delay in verification and high risk to find critical bugs due to the short testing period	**Clear communication between departments to have a synchronized release plan:** provide a set of iterative releases to reduce deviation between implementation and verification, lead time and risk for unexpected cost

Table 6. Barriers at the Function Verification Level

	Barriers (B)	Enablers (E)
16	**Detecting critical difference between implemented solution and initial function request at late phases:** high risk for undesired changes of function specification after going through component, system and implementation gates due to wrong understanding of provided requirement from previous design level	**Review approach:** increase the number of review on specification and improve the level of communication between team members
17	**Finding issues late in project:** delay in planning of software releases	**More resource at early phases:** identify product change requests at primary phase by allocating more resources at early stages and conducting more verification loops

3.7 Barriers at the Implementation Level

Table 7 shows a list of barriers at this level. Barriers are related to dependency of SW to HW, working process at software development and limited verification possibility. Related enablers are decoupling software and hardware, introducing verification process which does not require having hardware for perform verification and improving software structure.

4. CONCLUSIONS

In this study, we addressed 21 barriers preventing shortening of the software development lead-time in the automotive industry. As we can see in Figure 1 most barriers are allocated at the left site of the V-model where design and implementation is conducted. We identified that inadequate or unclear requirement is the common barrier for all implementation steps. In future work, we aim to explore further identified barriers and determine the effect of individual barrier in software lead-time. The company is moving towards more iterative development and therefore more research will follow in this area.

5. REFERENCES

[1] Eklund, U., Jonsson, N., Eriksson, A., & Bosch, J. (2012). A reference architecture template for software-intensive embedded systems. Proceedings of the WICSA/ECSA 2012 Companion Volume (pp. 104-111). ACM.

[2] Suh, N. (1998). Axiomatic design theory for systems. Research in engineering design, 10(4), 189-209.

[3] Kumar, S., & Kumar, T. (2011). Study the Impact of equirements Management Characteristics in Global Software Development Project: An Ontology Based Approach. International Journal of Software Engineering and Application, 2(4).

[4] Broy, M., Kruger, I., Pretschner, A., & Salzmann, C. (2007). Engineering automotive software. Proceedings of the IEEE, 95(2), 356-373.

[5] Bhatt, G. (2001). Knowledge management in organizations: examining the interaction between technologies, techniques, and people. Journal of knowledge management, 5(1), 68-75.

[6] Gershenson, J., Prasad, G., & Zhang, Y. (2003). Product modularity: Definitions and benefits. Journal of Engineering Design, 14(3), 295-313.

[7] Kendall, E. (2000). Reengineering for separation of concerns. Proceedings of the Workshop on Multi-Dimensional Separation of Concerns in Software Engineering at ICSE'00.

[8] Richardson, J. (1993, Juli). Parallel sourcing and supplier performance in the Japanese automobile industry. Strategic Management Journal, 14(5), 339.

[9] Dersten, S., Froberg, J., Axelsson, J., & Land, R. (2010). Analysis of the business effects of software architecture refactoring in an automotive development organization. EUROMICRO Conference (pp. 269-278). IEEE

Table 7. Barriers at the Implementation Level

	Barriers (B)	Enablers (E)
18	**Lack of a complete software verification process:** Frequently releases of internal developed software at late stage	**Establish a proper verification process:** verification at all steps including model, generated code and software itself contributes to reduce number of errors in software
19	**Refactoring hindered by incomplete claims and incomplete modularity:** complexity to interact with the function requirements introduced during the implementation	**Review software architecture:** using component-based or product line strategy [9] to avoid product changes by facing new function requirement, review the software architecture
20	**Inappropriate working process at software development:** high maintenance cost and unnecessary administrative job	**Clarify benefit of aligned software development process:** provide a common verification procedure having same working process and tools used by all units
21	**Unable to perform verification of the software due to lack of hardware component:** dependency between software and hardware	**Decoupling software from hardware, using virtual verification process:** using AUTOSAR or a virtual verification tool which simulate the same test scenario