Distribution of Effort among Software Development Artefacts: An Initial Case Study

Niklas Mellegård and Miroslaw Staron

Department of Applied Information Technology
Chalmers University of Technology, University of Gothenburg
{niklas.mellegard,miroslaw.staron}@ituniv.se

Abstract. Model-driven development aims at increasing productivity by raising the abstraction level of software specifications and introducing automated transformations for replacing lower level specifications. To assess benefits of replacing a legacy development process with a model-driven approach, one needs to establish a baseline of the current process with respect to the effort invested in the development artefacts. In this paper we report on an initial case study in which we investigate the main artefacts in the analysis and design phase with respect to required effort and perceived importance. We studied a non-model driven development of software based automotive functionality and our initial results show that a few artefacts receive the majority of effort, and that the artefacts that receive the most effort are not the most important ones. The initial results indicate that the distribution of effort between models and other artefacts is similar to that of model-driven projects in spite of the project being perceived and characterized as code-centric.

Keywords: Software engineering, Requirements, Analysis, Design, Modelling, Process.

1 Introduction

Model-driven development (MDD) [1-3] has the goal of increasing development productivity and the quality of software based products by raising the level of abstraction at which the development takes place. Several studies have provided evidence showing that the application of MDD in large industrial project has indeed improved productivity and the quality of the products (e.g. [4-7]), while other studies show mixed results and also point out the lack of objective empirical evidence [8].

Industrial software development, however, rarely provides the possibility to go from code-centric to model-driven software development in a single step and to the full extent. One example of software development domains where this is not possible is automotive software development. The domain is characterized by the existence of a lot of legacy software and high interdependence between car manufacturers and suppliers of car components. This high interdependence requires precise specifications that have to provide possibilities for interoperating of software development

I. Bider et al. (Eds.): BPMDS 2010 and EMMSAD 2010, LNBIP 50, pp. 234–246, 2010.

practices at the manufacturer's side and the supplier's side. This, in consequence, means that a number of practices for using models are used in parallel – e.g. UML, SimuLink. Since modelling notations differ in the degree of formality and quite often can be used interchangeably, it is important to optimize the cost of using these notations. The cost has to be balanced with the benefits that these models bring and perhaps not used in those parts of software development where code-centric approach is already very efficient.

In this case study we explore the costs and efforts of using different modelling notations and different abstraction levels for specifying requirements and designing the software in the automotive domain. The case study was conducted at Volvo Car Corporation (VCC), within the department responsible for electronic and software systems in Volvo automobiles. The research question in the case study was:

> *What is the distribution of the effort invested in the artefacts, documents and deliverable products in the main software specification phase?*

Addressing this research question provided the possibility to assess the costs of using different kinds of modelling notations in software development. The data was collected via document analysis and a sequence of interviews with architects and project managers. The main perspective of the results is the project managers'. The results show that there are a few artefacts which require significantly more effort than other artefacts and that the perception of what is important is different from where the effort is spent.

The rest of the paper is structured as follows. Section 2 describes the most related work in the area of studying effort of software development activities. Section 3 presents the fundaments of the case study – description of the de-facto process of software development at VCC. Section 4 outlines the design of the case study, while section 5 presents the results, and section 6 concludes the paper.

2 Related Work

The concepts which we investigated in this case study were based on the map (referred hereon to as *domain model*) of the development phases reported in our previous research [9]. In [9] we have evaluated the correctness of the domain model, discussed its expressive power and assured that it represents the de-facto development process at VCC.

Mohagheghi and Dehlen [8] conducted a review of documented modelling experiences and concluded that there are few reported results of how the MDD scales to large system development. Furthermore, the paper concludes that there often is a lack of company baselines which results in subjective evaluations. Our case study is intended to contribute to such a company baseline intended to be used to compare the development process with other companies.

Method engineering [10-12] recognizes the fact that no development method will fit every development task. Instead, method engineering aims at establishing a framework for adapting existing methods to better fit the development task at hand [11]. However, what is entailed in "better fit" differs greatly between projects. In this case study we examine – from the perspective of project managers – which development artefacts are considered central to the development process and thereby contributing with evidence of which parts of the development process should be in focus if an adaption of the current process be made.

In [7] we contrasted how effort was distributed between the different development phases in model-driven and code-centric projects within a company in the telecommunication domain. The results showed that the overall efficiency of the development process improved by adopting an MDD approach, the evidence for the analysis and design phase were not conclusive. In this case study we examine the analysis and design phase, specifically with respect to the development artefacts created. Furthermore, in [7] we found that the implementation phase had the most improvement potential; however, in the automotive domain the majority of implementation is done by third-party suppliers. This raises the question of how an MDD approach can be used to improve the efficiency of the analysis and design phase. In this case study we have taken the initial step by examining the distribution of effort among and the perceived importance of the artefact developed in the phase.

Bollain and Garbajosa proposed in [13] an extension to the ISO/IEC 24744 [14] meta-methodology standard. Their purpose was to extend the standard in order to better fit a document-centric (referred to as *code-centric* in this paper) development process. Our research has similar goals, but instead focuses on the use of models within a non-MDD development process. The case study reported in this paper provides evidence of which artefacts are the central ones, which in turn indicate where the focus of a modelling meta-model may be.

Broy [15] outlines the challenges in automotive software engineering, and also outlines a structural view on the development process, which he calls a comprehensive architecture. Our case study focuses on one particular part of this architecture, namely what Broy refers to as the *Design level* (which we refer to as the *function definition phase*). Moreover, Broy concludes that although models are used throughout the automotive software development process, their use is fragmented. This means that the benefit of having a coherent model chain – such as automatic artefact generation and traceability – is lost. Our case study contributes with empirical evidence that characterizes the development process – with regard to effort and importance of the constituent artefacts – which we hope will contribute with further evidence of how such an integrated modelling chain can be created in an optimal way.

Heijstek and Chaudron [5] report on an empirical study regarding model size, complexity and effort in a large model driven process, but whereas their study included the investigation of effort distribution among categories of development activities for the whole development process, our study focuses on how effort is distributed among the artefacts produced in one of the development activities, and specifically on the effort distribution among types of models and requirements. Moreover, Heijstek and Chaudron report on the importance and centrality of models in a pure MDD project, whereas our case study is conducted at a company which does not use an MDD approach. We elaborate on this in section 5.

3 Domain Model

The development of software based functions at VCC makes extensive use of models, although it cannot be classified as ‘model-based’ according to the definitions by

Brown [3]. The types of models used are both descriptive and prescriptive, as defined by Ludewig [16]. However the use of modelling techniques is fragmented and does not constitute a coherent chain of models, as model-driven development advocates [9]. This fragmentation is typical and evidence for such a fragmentation in new adopters of model driven engineering has also been reported by Broy [15].

The development of the products at the electrical department is separated into four phases (described in more detail in our previous research [9]) which – as shown in Fig. 1. – are: the *strategic phase* which is focused on selecting a number of high level features; the *function definition phase* in which individual functions are analysed and specified, and finally; the *system* and *node development phases* which are concerned with designing and specifying a software and hardware solution that realize the functions. The strategic phase is concerned with product planning, where a set of high-level features are selected in order to make the vehicle model competitive at the market. The subsequent phases contain the engineering activities which are of main interest in our research, and the activities in the function definition phase are the focus of this case study.

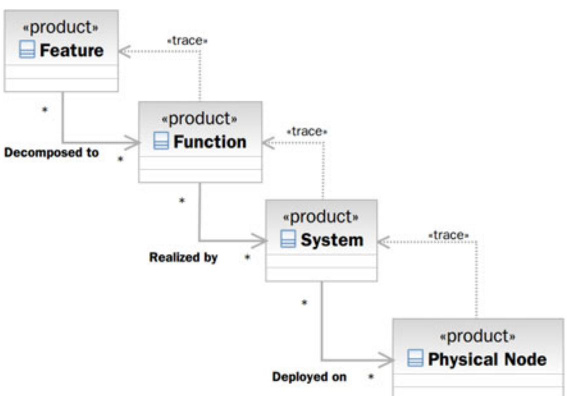

Fig. 1. Product Development Stages

The activities in the function development phase result in design and specification of a function, e.g. airbags or collision warning. In the phase, only the implementation independent properties are under consideration, such as desired behaviour and proposed algorithmic solutions.

In the system development phase the designers specify a system solution that satisfies the specified behaviour of the function on a specific hardware and software platform. In the component development phase the focus is on design and specification of the physical components that are part of the system solution. The components are then implemented, mostly by third party suppliers. Since the implementation is done by the suppliers, the most important design phase is the function design phase, which we study in our research.

Fig. 2. shows the key concepts we have identified in the function development phase [9]. The map in Fig. 2. was created using domain modelling in UML (i.e. using class diagrams with classes as concepts, associations and dependencies). We used three stereotypes to distinguish between types of elements in the domain model:

- *Product*: elements which are results (or in early stages of the project – prospected results) of the software project – e.g. features, functions, components.
- *Artefact*: elements which are used in the development process – e.g. models, requirements, abstract pseudo-code, algorithm descriptions

- *Document*: elements which are official documents prescribed by the software development process at VCC and can be deliverables from the process – e.g. requirement specification

The above three categories show that the three kinds of elements interact and complement each other.

The concepts we have identified in the function development phase include

- The function requirements specification
- Description of the functions behaviour from an end-user perspective, shown in Fig. 2. as *Use Cases*, *Specification Model* and *Requirement*
- Descriptions of algorithmic solutions, shown in Fig. 2. as *Simulation Model* and a textual description of the simulation model, shown as *Design Description*
- An initial design, shown in Fig. 2. as *Logical Design*

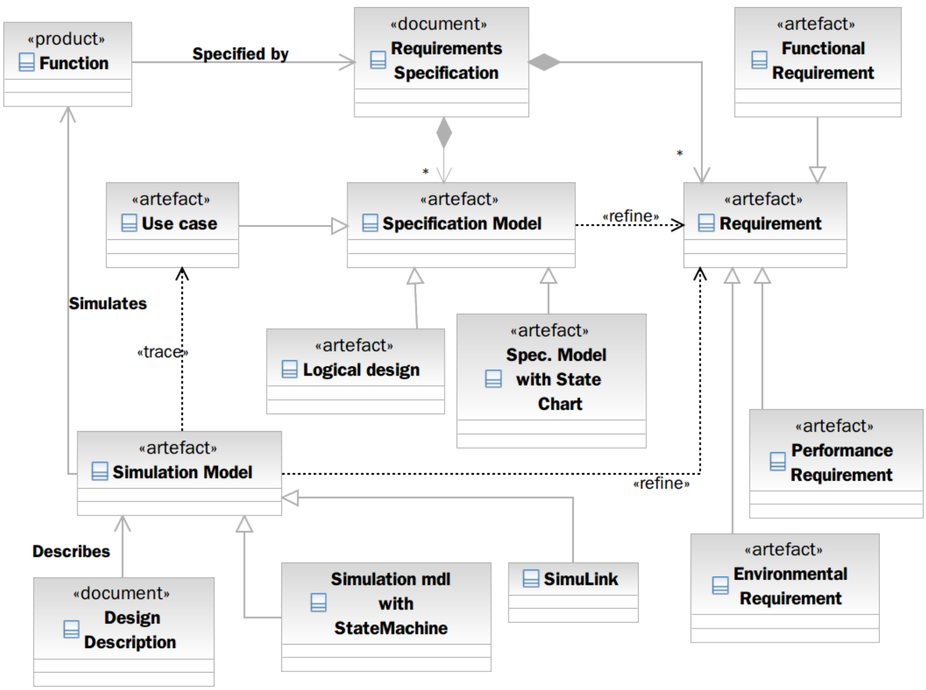

Fig. 2. Artefacts, documents, and software products in the function development phase

4 Design of the Case Study

When designing this case study we followed the case study design proposed by Yin [17]. Initially a case study protocol was set up in which two case study propositions were defined. For each of the two propositions, a hypothesis was formulated and interview questions created that would elicit evidence to support or refute the hypothesis. Using the data collected from the interviews we evaluated the hypotheses and addressed our research question.

In the following subsections we present an outline of the design of the case study: propositions, units of analysis, data collection procedures, and criteria for interpreting the data.

4.1 Proposition 1 – Relative Importance of Artefacts

Based on our previous research [9, 18] we anticipated that there would be large differences between the relative importance of artefacts and the effort put into their development. In other words we anticipated that the most effort-intensive artefacts were probably not the most important ones from the perspective of product development. These differences meant that there could be a 'waste' of effort (in terms of LEAN [19]) and therefore there was a room for improvement or effort optimization.

4.2 Proposition 2 – Effort Distribution among Artefacts

We proposed that there would be a significant difference between the amounts of effort spent on the artefacts identified in the domain model. Furthermore, we proposed that there is a pattern (with respect to artefact type) in how the effort is distributed among the development artefacts.

In order to identify improvement opportunities in the current process, it is important to know which artefacts are the most effort-intensive and therefore which artefacts we should concentrate on in the first place. If the perceived effort and importance do not match, then there is a need for deeper investigation and optimization of the development process at the company.

4.3 Units of Analysis

The main unit of analysis in this case study was the importance of and the relative amounts of effort invested in the development artefacts, specifically requirements, different types of models and documents. However, there was also an embedded unit of analysis in this case, namely the role of the interviewed subjects, as each role has a particular perspective on both the importance of the artefacts in question as well as the amount of effort that is invested in creating them.

4.4 Data Collection Procedures

The data collection consisted of structured interviews in which we collected evidence of the perceived importance and effort spent on the artefacts we identified as being the key concepts in the function development phase (see Fig. 2.).

We first presented and described domain model showing the key concepts in the function development phase (Fig. 2.). The purpose with this presentation was to provide the respondent with the context of the study and also to provide the opportunity to ask about and comment on the content and structure of the domain model (e.g. add concepts that were found important or change relationships among the shown concepts). We conducted this part as a focused interview [17], and in order to ensure construct validity, we noted the comments and incorporated them in the following parts of the interview (in accordance with Yin [17]).

We primarily wanted to interview subjects that had an overview of the entire project in order to avoid introducing bias to specific activities; therefore our sample consisted of:

- **Main study.** The main study sample consisted of three project managers. All subjects had been working at VCC for many years, and within the function development phase between 6 months and several years
- **Pilot study.** The sample used in the pilot study consisted of two architects who had been working as architects for several years.

4.4.1 Units of Analysis for Proposition 1

The unit of analysis in proposition 1 was the perceived importance of the artefacts in the domain model. In order to collect data regarding their importance, we divided the artefacts in the categories models, requirements and documents, and asked the interviewee which they thought was the most important to get right in order to ensure the success of developing the function. Furthermore, in order to contribute to our understanding of why the particular category of artefacts was considered the most important, we asked to interviewee to explain and give examples. These comments were noted for further analysis.

4.4.2 Units of Analysis for Proposition 2

Regarding proposition 2, we collected data by applying a variation of the $100 test technique [20], where we asked the interviewees to place an appropriate amount of money on each of the artefacts in the domain model. Because the artefacts shown in the domain model are constituent parts in the document, we considered effort spent on the documents to be editorial work or tasks not associated with the other artefacts shown in the domain model.

The amount of effort assigned to each artefact was normalised by calculating for each concept the percentage of the total amount distributed. In addition, the sum of effort for each category of artefacts was also calculated (in order to match the categories of artefacts from proposition 1).

We followed up on the result of the $100 test by asking the interviewees whether they thought it was straight-forward to assign the amount to each artefact, and if not, what they felt was the main difficulties in doing so. Furthermore, we asked whether they thought that their distribution of effort would be representative for other roles in their project, as well as for other projects.

4.5 Interpreting the Findings

We intended to analyse the results by means of pattern matching [17]. Based on our earlier research, we anticipated that there could be a difference between which artefacts were considered the most important and the effort that was put into their development. By comparing the results of propositions 1 and 2, we examined whether the category of artefacts considered most important was in fact the same as the one most effort was invested in.

By studying the effort distribution at VCC we could establish a reference point for comparing effort distribution in MDD projects in different companies or domains – e.g. by comparing that to the study conducted at Ericsson [7].

4.6 Validity Evaluation

We have identified and grouped the threats to validity in our study according to recommendations of Yin [17]:

Construct validity. The main instrument of this study is the domain model around which the questions of the interview revolved. In order to ensure construct validity, we began each interview session by presenting the domain model and asking whether the interviewee found any concepts missing. In neither of the interviews did we identify any issues with the domain model; we thus consider the model to be complete and correct.

Internal validity. As any interview study we anticipated some personal bias in the answers from the interviewees. In order to minimize this threat we triangulated the results by conducting document analysis and a pilot study with architects who are not involved in the actual case study.

External validity There is a risk that the results are too specific to Volvo Car Corporation. In order to validate the results we plan to conduct a similar study at other companies which our university collaborates with.

Reliability. As part of the case study design, we have created a case study protocol which ensured that we conducted the study and collected the data in a consistent manner. By using this protocol, we believe that the study can be reliably reproduced.

5 Results

5.1 Pilot Study Results

In the pilot study we interviewed 2 architects. The subjects, however, were not working with detailed design of vehicle functions and were therefore used only to validate the design of our case study.

As a result of the pilot study we found that the domain model was valid, i.e. complete and correct. None of the architects identified issues (such as missing or invalid concepts) with the domain model (nor did such issues arise during the rest of the interviews with project managers). Furthermore, as result of the pilot study we preliminary identified which artefacts might be the most important ones – e.g. Use Cases. This information made us aware when interpreting later findings in the case study.

We have also found that it is probably the models that contain most of the information, but that this is dependent on the role of the subject. As a result of this remark we have checked the list of planned subjects in our study and found no risk of introducing bias by missing important roles.

Finally we have found that specification models and the related requirements are more important than simulation models, although it might vary significantly depending on the product that is under development. As a result of this we have ensured that we introduced another question – which products the interviewee has in mind when answering our questions.

5.2 Case Study Results

In this section we report the initial results of the case study. At this point three project managers have been interviewed; our study, however, is ongoing and results from a larger sample will be reported in a future paper.

5.2.1 Case Study Proposition 1

With regard to the importance of the development artefacts, we found that requirements, and especially performance and functional requirements related to design components such as sensors and actuators, were considered the most important ones. The rationale explaining this was, according to the interviewees, that other models in the design of the functional solution, such as the simulation models, rely on these requirements as assumptions about the available design components and their properties. Moreover, we found that the simulation models were not considered to be that important, as they can easily be reworked if the requirements change.

We had initially anticipated that models would be the most important artefacts, as they are generally more expressive than text. However, we found that it is rather the level of detail and the need to express requirements unambiguously that is considered most important from the project manager's perspective. We found that the models are, from the project manager's perspective, mainly seen as support for the requirements; specification models are used to provide structure to the requirements or to provide them with context. In particular, the simulation models are used to explore possible functional solutions in order to learn to understand what the detailed requirements are.

Furthermore, the detailed requirements created in the function definition phase are provided as input requirements to the subsequent system and component development phases, where they are further refined to form part of the specification that is eventually provided to the third-party supplier. Therefore it was considered highly important that the requirements were fully understood and correct.

5.2.2 Case Study Proposition 2

The models used in the function definition phase, listed in Table 1, consists of

- One or a few use case descriptions, including alternative flows and error states;
- Specification models, such as state machines, refining the use-case into a detailed functional specification;
- Simulation models intended both as validate the function design, as well as to serve as a tool for the function designer to learn to understand the relevant requirements.

The level of formality of the models ranges from informal, such as use cases, to formal ones such as executable simulation models.

From the normalized result of the $100 technique, shown in , we found that about twice the effort is invested in models compared to requirements. This initial finding supported our anticipated result that the artefacts considered most important are not the ones the most effort was spent on. Furthermore, we found that the artefact the most effort is spent on is the simulation models which can be explained by the fact that these models are used to make the requirements more precise.

We also found that a significant amount of effort (approximately the same as is spent on requirements) is spent on editing documents.

Table 1. Models in the Function Definition Phase

Artefact	Model	Notation
Function Description	Use-case	UML/Use-case
	Specification models	
	Logical design	UML/Class Diagram
	State charts	UML/State Machine
	Simulation models	
	Simulink	Proprietary
	Statemate	Proprietary
	Stateflow	Proprietary
	Powerpoint	N/A
	Video sequences	N/A

When showing the results to the interviewees and their managers after the study, their reaction was that this was against their expectations – they expected more effort to be spent on the requirements than it is now. This called for a subsequent (planned for future) study about detailed optimization techniques how to improve the throughput of the process. Possible targets for optimization is the multitude of different modelling techniques and notations used that are compatible to a limited degree. Similar observations and a report on how such challenges was overcome when Motorola successfully introduced MDD in their process is reported in [4].

Table 2. Effort distribution in percent per artefact as estimated by the project managers

Artefact	PM 1	PM 2	PM 3[1]
Models	*(56.4)*	*(50)*	*(73.4)*
Simulation model	39.7	30	26.7
Use cases	7.9	20	20
State Machines / other specification models except UCs	4.0		26.7
Logical Design	4.8		
Requirements	*(23.8)*	*(30)*	*(26.7)*
Requirements (all types)	23.8	30	26.7
Documents	*(19.8)*	*(20)*	
Design Description	11.9		
Requirements Specification	7.9		

5.3 Interpretation of the Results

Interestingly, from the result of the case study propositions we found that the artefacts which were considered as the most important were not the same artefacts that receive the most effort.

Whereas requirements were considered very important, they received about half the effort compared to the simulation models.

Furthermore, creating the documents – mainly considered by the interviewees to be editorial work – required almost as much effort as the requirements do, while not considered as important. We raised an improvement potential here – perhaps the tedious work should be reduced.

We have found that the simulations models are often used during the function definition phase to explore different functional solutions, and thereby assisting the developers in understanding the requirements (conf. [9]). This would explain why such large amount of effort is spent on developing these models. However, this does not explain why the simulation models are not considered as important as the requirements.

Furthermore, the results shown in indicate a strong correlation between level of model formality and amount of effort required. Developing the formal specification model takes according to one subject almost all modelling time – approx. 5 times more than the next one – Use cases model.

As the formal simulation models are not considered as important to the success of the product, it raises the question of why their importance is perceived to be so low. We plan further investigations at the company to explore it to a deeper extent.

The results of case study proposition 2 indicate that documents require a substantial amount of effort and as the majority of their contents are compositions of the other artefacts. Here automating the process of document creation may be a way to improve process efficiency.

Although VCC cannot be considered as using an MDD approach, we have found that the relative amount of effort spent on models compared to other artefacts is similar to what Heijstek and Chaudron found in their paper [5] where they investigate pure MDD projects. In their study, Heijstek and Chaudron found that in a pure MDD project at an IT service provider, 59% of all effort is spent on developing models, whereas in our

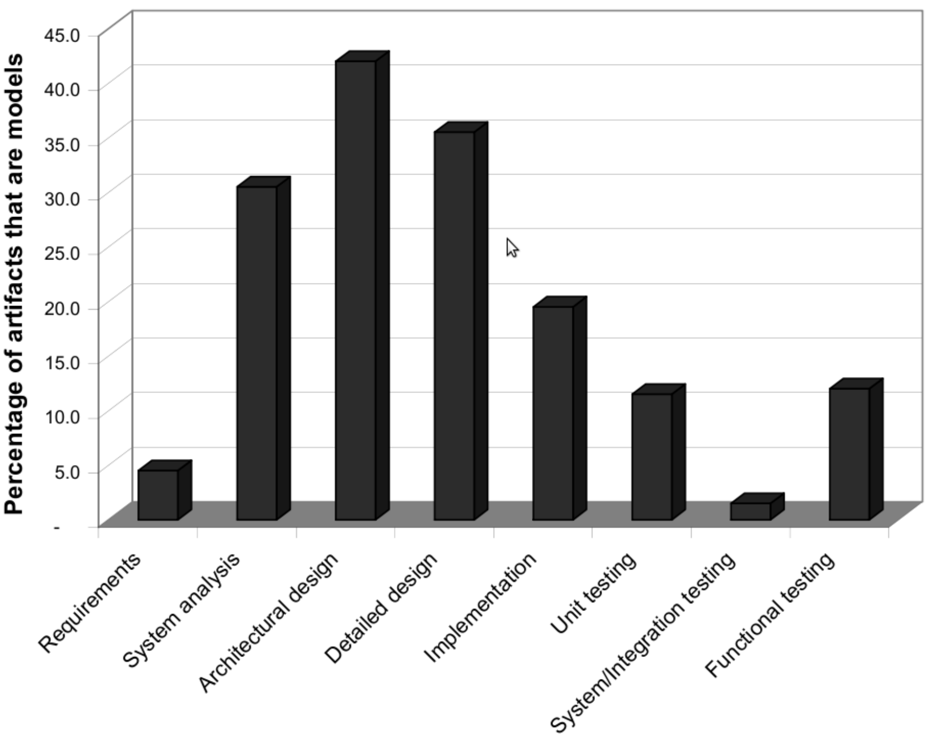

Fig. 3. Percentage of artefacts which contain models per phase from [7]

study we found it to be about 60%. Although the product development process at VCC does not comply to a pure MDD approach – but can rather be described as document-centric [21, 13] – models are frequently used in the requirements specification documents, but they are mainly used to provide context and structure for the requirements.

Furthermore, Heijstek and Chaudron found quantitative evidence that some kinds of models receive more attention than others. In our case study we found similar qualitative results. This finding suggests that a strategy for improving process efficiency is to concentrate such improvement efforts on particular types of models and their associated activities. The results of our study indicate that such effort should be directed at simulation models and their traceability to the requirements that constrains their design.

In the light of our previous research [7] we have investigated how much modelling is present in different phases – the results which are presented in Fig. 3. The results show that modelling is rather a large part of the phases covered by our study – analysis, architectural design and system design – which are all part of the function development phase. Our new findings raise an important question – how much of the modelling in these phases is essentially needed?

6 Conclusion

The objective of this study was to investigate how effort and the perceived importance are distributed among artefacts identified in the function development phase of software based vehicle functions. In this paper we have reported on an initial case study in which we interviewed project managers responsible for a software based safety and security related functions.

Our initial results show that the artefacts that are considered the most important are not the ones that the most effort is invested in. While detailed textual requirements are considered the most important artefact in order to ensure a successful product, it is executable simulation models that receive the majority of effort during development. Furthermore, from the interview we found that the simulation models are not considered very important as they are easy to modify if the requirements should change.

Moreover, we have in this initial study of a non-MDD project found that the amount of the total development effort invested in models is similar to the pure MDD project reported in [5] – about 60% in our case and 59% in the MDD case.

As part of our continuing work we are conducting this case study on a larger scale by interviewing more people in a number of different roles in order to add to the empirical evidence of how effort and importance of development artefacts such as models and requirements are distributed.

Acknowledgements

This research is partially sponsored by The Swedish Governmental Agency for Innovative Systems (VINNOVA) under the Intelligent Vehicle Safety Systems (IVSS) programme.

References

1. Atkinson, C., Kuhne, T.: Model-driven development: a metamodeling foundation. IEEE Sw. 20, 36–41 (2003)
2. Brown, A.: An introduction to Model Driven Architecture - Part I: MDA and today's systems, `http://www.ibm.com/developerworks/rational/library/3100.html`
3. Brown, A.W.: Model driven arch.: Principles and practice. Sw. and Sys. Model 3, 314–327 (2004)
4. Weigert, T., Weil, F., Marth, K., Baker, P., Jervis, C., Dietz, P., Gui, Y., van den Berg, A., Fleer, K., Nelson, D., Wells, M., Mastenbrook, B.: Experiences in Deploying Model-Driven Engineering. In: Gaudin, E., Najm, E., Reed, R. (eds.) SDL 2007. LNCS, vol. 4745, pp. 35–53. Springer, Heidelberg (2007)
5. Heijstek, W., Chaudron, M.: Empirical Investigations of Model Size, Complexity and Effort in a Large Scale, Distributed Model Driven Development Process. In: 35th Euromicro Conference on Software Engineering and Advanced Applications, 2009. SEAA 2009, pp. 113–120 (2009)
6. Baker, P., Loh, S., Weil, F.: Model-Driven Engineering in a Large Industrial Context — Motorola Case Study. In: Briand, L.C., Williams, C. (eds.) MoDELS 2005. LNCS, vol. 3713, pp. 476–491. Springer, Heidelberg (2005)
7. Staron, M.: Transitioning from code-centric to model-driven industrial projects – empirical studies in industry and academia. In: Model Driven Software Development: Integrating Quality Assurance. Information Science Reference, pp. 236–262 (2008)
8. Mohagheghi, P., Dehlen, V.: Where Is the Proof? - A Review of Experiences from Applying MDE in Industry. In: Schieferdecker, I., Hartman, A. (eds.) ECMDA-FA 2008. LNCS, vol. 5095, pp. 432–443. Springer, Heidelberg (2008)
9. Mellegård, N., Staron, M.: Use of Models in Automotive Software Development: A Case Study. Presented at the First Workshop on Model Based Engineering for Embedded Systems Design, Dresden, Germany, March 12 (2010)
10. Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Information and Software Technology 38, 275–280 (1996)
11. Rolland, C.: Method eng.: towards methods as services. Sw. Process: Improvement and Practice 14, 143–164 (2009)
12. Karlsson, F.: Meta - Method for Method Configuration: A Rational Unified Process Case, dissertation, Linköping University of Management and Engineering, VITS-Development of Informations Systems and Work Context (2002)
13. Bollain, M., Garbajosa, J.: A Metamodel for Defining Development Methodologies. Software and Data Technologies, 414–425 (2009)
14. ISO - International Organization for Standardization: ISO/IEC 24744:2007, software Engineering – Metamodel for Development Methodologies (2007)
15. Broy, M.: Challenges in automotive software engineering. In: Proceedings of the 28th international conference on Software engineering, pp. 33–42. ACM, Shanghai (2006)
16. Ludewig, J.: Models in software eng. – an introduction. Sw. and Sys. Modeling 2, 5–14 (2003)
17. Yin, R.K.: Case Study Research: Design and Methods, 3rd edn. Applied Social Research Methods Series, vol. 5. Sage Publications, Inc., Thousand Oaks (2002)
18. Mellegård, N., Staron, M.: A Domain Specific Modelling Language for Specifying and Visualizing Requirements. In: The First Int. Workshop on Domain Engineering. DE@CAiSE, Amsterdam (2009)
19. Poppendieck, M.: Lean Software Development. In: Companion to the proceedings of the 29th International Conference on Software Engineering, pp. 165–166. IEEE Computer Society, Los Alamitos (2007)
20. Leffingwell, D., Widrig, D.: Managing software requirements. Addison-Wesley, Reading (2003)
21. Rausch, A., Bartelt, C., Ternité, T., Kuhrmann, M.: The V-Modell XT Applied–Model-Driven and Document-Centric Development. In: 3rd World Congress for Software Quality (2005)