Utilizing Modeling Approach Patterns in the Embedded Software Industry

Deniz Akdur
ASELSAN Inc.,
Ankara, Turkey
denizakdur@aselsan.com.tr

Onur Demirörs
Department of Computer Engineering
İzmir Institute of Technology
İzmir, Turkey
onurdemirors@iyte.edu.tr

Abstract—**To cope with the growing complexity of software-intensive embedded system development, modeling is a widely used approach. The modeling approaches in the embedded software industry vary depending on many modeling characteristics (e.g., purpose, modeling rigor, medium type used, modeling stakeholder profile, etc.). In the light of our previous studies, we have identified modeling approach patterns of embedded software development projects and constructed a characterization model. This model not only identifies and defines modeling approach patterns of the stakeholder in embedded software development projects, but also gives recommendations for commonsense modeling practices. In this article, one of the application results of this characterization model, which was performed in Defense & Aerospace sector is presented. The model was successfully applied to the case, in which a series of both structured and semi-structured interviews with 17 embedded software professionals were conducted. The results show that identification of individual patterns provide insight for improvement both for individuals as well as larger units of operations such as projects and organizations.**

Keywords-embedded software; modeling; sketching; model-based; model-driven; modeling patterns and cultures; modeling characteristics; characterization model, case study

I. INTRODUCTION

Software modeling helps engineers to work at higher levels of abstraction, facilitates communication and automates the generation of software development life cycle (SLDC) artifacts (e.g., code, documentation, test case) to manage the complexity of software-intensive embedded systems [1]. The modeling approaches in the embedded software industry vary since modeling characteristics differ among systems and sectors, e.g., consumer electronics, defense or automotive [2]. At one extreme, some stakeholders (e.g., systems engineers) use modeling informally, where diagrams are sketched on a paper in order to communicate with colleagues. At the other extreme, for other stakeholders (e.g., software developers), modeling turns into programming with automated generation of code [3].

In order to decide how much modeling rigor is necessary for the embedded software development (e.g., as a sketch without modeling language formality or by automating software artifact generation as in model driven engineering (MDE)), the identification of the relations between modeling characteristics (e.g., rigor, purpose, medium type, SDLC phase) is very important while identifying and defining "*modeling patterns*".

In the literature, there aren't many research studies related to the modeling patterns and categories [4, 5]. Note that none of these studies are also directly related with embedded software development. Moreover, there is not any approach, which provides a modeling guidance for different stakeholders in the embedded software development projects by creating and increasing the awareness of what modeling stakeholders do with different needs and project characteristics. Therefore, we have focused to bridge a gap in the existing literature by constructing a characterization model, called MAPforES standing for "*Modeling Approach Patterns for Embedded Software*". This model not only identifies and defines a stakeholder's pattern and culture but also presents what the other stakeholders, who have similar profiles in the embedded domain is doing while modeling. The goal of this paper is to report one of the application results of MAPforES and to increase the awareness of different modeling stakeholders by suggesting various modeling practices. The empirical study reported here is based on case study, which included a series of both structured and semi-structured interviews in a Defense &Aerospace company. The case study took over two months with 17 embedded software professionals from different software engineering (SE) roles (e.g., from developer to tester and systems engineer to project manager).

The remainder of this paper is structured as follows. Section 2 gives an overview of MAPforES. Section 3 presents the research process, the findings and the potential validity threats. Finally, Section 4 concludes this study.

II. OVERVIEW OF THE CHARACTERIZATION MODEL: MAPFORES

MAPforES defines software modeling characteristics in an embedded software development project and assists modeling stakeholders to realize an effective modeling approach with respect to these characteristics by also giving an opportunity to the stakeholder to compare and contrast what embedded software professionals with the similar profiles (e.g., role, industrial sector) are doing while modeling [6].

We have constructed MAPforES via two iterations. As a first step, in order to understand the state-of-the-practices in modeling practices in the embedded software industry, we conducted a global survey. Participants were from 27 countries working in different subsectors of embedded software industry and SE roles [7]. The survey showed that the embedded software professionals use modeling approaches in varying degrees with different needs. All of the usages (e.g., informal sketching, model based engineering (MBE), or MDE) could be effective depending on the characteristics of modeling. Based on the results of our findings, we investigated the relations between modeling characteristics [8] and created the preliminary version of the modeling approach pattern set. This set was based on these significant modeling characteristics: "*purpose*", "*medium type*", "*archivability*", "*modeling language, if any*" and "*SDLC phase*" [9]. However, there might be some hidden patterns [6] (e.g., the patterns, who do not know exactly their software modeling characteristics, especially their modeling rigor and hence they are unaware as

978-1-7281-1740-9/19/$31.00 ©2019 IEEE

to whether they use software modeling or MDE), which could not be found out from the analysis of survey data. Therefore, there was a need to validate and improve this preliminary version. As a second step, we improved these quantitative results with a deeper and more qualitative strategy via in-depth interviewing over eight months with 53 embedded software professionals across a variety of target industrial sectors and roles [10]. As a result, the set of patterns were extended to 12 patterns, where the final set is given in Table 1 [10].

TABLE 1. MODELING APPROACH PATTERNS

Main Pattern	Modeling Approach Patterns			
model-driven	3.3	With DSL-like*	Purpose of the modeling includes "Code generation" or "Test case generation (e.g., Model Based Testing (MBT))"	With "any DSL-like" usage
	3.2	Without DSL-like		Without any "DSL-like" usage
	3.1	Limited	Only with "Documentation generation", "Model simulation" or "Model to model transformation" purpose	
	3.x	Unaware of MDE	Purpose of the modeling includes any MDE-specific purpose, but the stakeholder is unaware of MDE usage	
model-based	2.2	Prescriptive	SDLC phase where modeling is used includes "implementation" or "testing"	
	2.1	Descriptive	SDLC phase does not include "implementation or testing"	
sketching	1.3	Archived	Purpose of the modeling includes "Documenting Analysis & Design" Media type used: "Analog media usage" such as paper/whiteboard is more than "Digital media usage"	
	1.2	Selective	Casually & informally with some formalized modeling language (most probably, UML elements) Modeling Language set includes sketch & any formalized modeling language (e.g., UML or DSL-like)	
	1.1	Ad-hoc	Purpose of the modeling includes only "Understanding" or "Communication" Only pen & paper / free format (e.g., without any formalized modeling language, e.g., UML) Medium type used while modeling is only analog (paper or whiteboard)	
	1.x	Unaware of modeling	The stakeholder is unaware of modeling although there is –some kind of- modeling (especially sketching)	
none	0.1	Bad experienced**	Not using any modeling approach.	
	0.0	Not experienced		

* "*With DSL-like*" means that the modeling language set of the stakeholder includes any kind of Domain Specific Language (DSL) (e.g., any DSL [provided by tool provider or their own design] such as AUTOSAR, AADL, EAST-ADL or, any UML profiles, which provides a generic extension mechanism for customizing UML diagrams such as MARTE, SysML, SoaML, any BPML or MATLAB Modeling Utilities etc.).
** As terminology, "*bad experienced*" pattern indicates the embedded software professionals, who don't use any kind of modeling due to disappointing and insufficient experiences of software modeling.

After defining modeling patterns, six modeling cultures in embedded software projects are identified: *None, Performed, Formalized, Archived, Prescripted and Auto-generated* [6]. Accordingly, a modeling culture (as a particular group of modeling approach patterns) consists of different characteristics of software modeling. In this categorization, a "higher" pattern/culture can use the characteristics of the "lower" patterns/cultures and the modeling stakeholder might apply the stakeholders' lower level patterns' modeling practices, if necessary; but not vice versa. For example, a modeling stakeholder, who is at pattern 3.3, can also use analog medium type (e.g., paper) besides digital ones (e.g., modeling tools in PC), i.e., sketching without any modeling rigor as if being at pattern 1.1. Therefore, a "higher" culture does not necessarily entail a more "correct" or "mature" use of modeling with respect to job/task requirements of the stakeholder although a change into a "higher" culture might allow the stakeholder to better use software modeling with possibly some extra costs and challenges. The cultures derived are detailed in [6].

After identifying the patterns and cultures, we started to create MAPforES. During this process, we firstly derived a decision tree mechanism, except for "hidden patterns" (See [6] for detailed derivation of the tree). Feedback from 14 software professionals was taken via expert opinion strategy before finalizing the decision tree, which is one of the 'view's of MAPforES. Accordingly, MAPforES, firstly, takes the modeling characteristics of the stakeholder as input. Depending on these characteristics, the modeling pattern and culture is found. Moreover, based on these characteristics, MAPforES presents what the other stakeholders, who have similar profiles is doing while modeling as a set of commonsense industrial practices. (via the database constructed with survey data and semi-structured interviews' findings [11]). By querying the similar profiles in the database with the stakeholder's input, the stakeholder learns as suggestions what their competitors do in the same context such as the necessary modeling approaches, languages, tools, etc. [6].

The application of the characterization model, which is the main objective of this study will be presented next to detail the research methodology and process by discussing the findings and the potential validity threats.

III. APPLICATION OF MAPFORES

In order to determine the benefits of MAPforES in practice, we performed case studies, which included a series of both structured and semi-structured interviews.

A. Research Methodology and Process

The goal is to apply and observe the usefulness and improvement opportunities of the MAPforES by identifying stakeholder's modeling processes. Based on this goal, the following research questions (RQs) are raised:

***RQ1**: In what ways does MAPforES reflect or fail to reflect stakeholder's current modeling pattern and culture?*

***RQ2**: How do the stakeholder evaluate MAPforES's usefulness and conceptual insightfulness?*

To address these RQs, an evaluation form (See Appendix of [6]) is used to evaluate the result of the model with respect to validation criteria [12]. For this study, due to space constraints, we report only the results of RQ1 (Please refer to [6] for the analysis of the evaluation of MAPforES, which were perceived by the participants; i.e., for RQ2).

We use interviews as a main source of evidence for two reasons: (1) to observe stakeholder's demographics and modeling practices to understand the modeling characteristics (i.e., the structured part, first round); (2) to understand personal experiences to confirm their responses via face-to-face in-depth analysis besides direct observations (i.e., the semi-structured part, second round). To prevent misinterpretations during data collection, a presentation on MAPforES was designed to be given on the site as the first step including brief information about the study, the model and the terminology used.

Before the company visits, the data to be gathered is outlined as a questionnaire (see Appendix of [6]) and there are also "evaluator notes" parts in it, which are filled out after the first round of the interview when the evaluator takes notes on all given responses. By this way, the interview results have both closed-ended and open-ended answers. During first round of the interview, the questionnaire provides all necessary inputs to MAPforES by taking necessary modeling characteristics such as purpose, medium type, modeling language, SDLC phase and stakeholder profile (e.g., university degree, SE role, target sector of the product) [6]. This first part is answered individually by the participant without any interaction of the evaluator. During the second part, which is conducted face-to-face, the responses of each participants are checked for as to whether there is any misunderstanding or any missing critical information in the questionnaire (e.g., wrong data for modeling practices). For further details about data collection, refer to [6].

After data collection process, all answers are analyzed and MAPforES is applied to the participant's modeling characteristics. After this stage, the evaluator sends two forms to the participants via email. The first form summarizes the interview results and the second form is used to evaluate the model usefulness by the participants. For further details about evaluation of the results, refer to [6].

Due to space constraints, in this paper, we report only one case study, which was conducted in a Defense &Aerospace company. This company is a global provider of advanced radar systems serving both military and civilian markets. The number of employees working in R&D engineering roles in this company is more than 3000. Having a CMMI-3 certification, the company is specialized in developing products with high-end software development techniques like agile programming, software product lines and reusable components. The size of a typical software development team, which includes different SE roles is 15-25 people. Specifically, a radar software project was chosen as a reported case study, which included 17 interviewees covering all SE roles in this project.

B. *Results and Discussion*

The results showed that there is a difference on modeling approach patterns for different project characteristics and SE roles (which was expected in line with previous literature [7, 10]); however the results also showed that this difference is also related to the tasks and responsibilities of modeling stakeholder besides her/his formal education (e.g., university degree).

We present different profile details from different modeling approach patterns observed in the case study to support this argument as shown in Table 2 (Note that this table is arranged and sorted according to the modeling patterns and cultures column; however online versions of these tables where other attributes can be chosen as sorting/filtering criteria to observe distributions of the rest are also available [13]). Please refer to Table 3 for the abbreviation used in Table 2.

TABLE 2. CASE STUDY RESULTS, DEFENSE & AEROSPACE SECTOR, RADAR SOFTWARE PROJECT

Participant#	Position	Degree		Experience (in years)		PL & HW closeness	Modeling Language(s)	Modeling Pattern		Modeling Culture	
		Academic	University	Work	Modeling			Observation / Interview	According to Model	Observation / Interview	According to Model
6	Dev, Desg, Arch	MSc	CS, IS	10+	10+	C, C++, Java Medium	Sketch, UML, UML profiles, DSL	3.3 **With DSL-like**	3.3 **With DSL-like**	Auto-generated	Auto-generated
11	Tstr	MSc	CENG	10+	10+	Java, C++ Low	Sketch, UML, DSL	3.3 **With DSL-like**	3.3 **With DSL-like**	Auto-generated	Auto-generated
3	Dev, Desg, Arch	MSs	EE, CENG	10+	10+	C, C++ Medium	Sketch, UML	3.2 **Without DSL-like**	3.2 **Without DSL-like**	Auto-generated	Auto-generated
9	Dev	MSc	EE	10+	6-10	C, MATLAB High	Sketch, UML, MATLAB	3.1 **Limited**	3.1 **Limited**	Auto-generated	Auto-generated
14	Sys	MSc	EE, SE	6-10	6-10	MATLAB Low	Sketch, UML, MATLAB, SysML	3.1 **Limited**	3.1 **Limited**	Auto-generated	Auto-generated
8	Dev, Desg	MSc	EE, CENG	10+	10+	C, C++ Medium	Sketch, UML, DSL*	3.x **Unaware of MDE**	3.3 **With DSL-like**	Auto-generated	Auto-generated
1	Dev, Desg	MSc	EE, IS	10+	10+	C++ Medium	Sketch, UML	2.2 **Prescriptive**	2.2 **Prescriptive**	Prescripted	Prescripted
7	Dev	BSc	EE	10+	10+	C, C++ Medium	Sketch, UML	2.2 **Prescriptive**	2.2 **Prescriptive**	Prescripted	Prescripted
10	Dev	BSc	EE	6-10	6-10	C High	Sketch, UML	2.2 **Prescriptive**	2.2 **Prescriptive**	Prescripted	Prescripted
12	Tstr	MSc	EE, CENG	10+	10+	C++ Medium	Sketch, UML	2.2 **Prescriptive**	2.2 **Prescriptive**	Prescripted	Prescripted
17	QA	MSc	CENG	10+	10+	BPEL Not applicable	Sketch, UML, BPML	2.1 **Descriptive**	2.1 **Descriptive**	Archived	Archived
2	Dev	BSc	EE	6-10	2-5	C High	Sketch, UML	1.3 **Archived**	1.3 **Archived**	Archived	Archived
16	PM	MSc	ME	10+	10+	Not applicable	Sketch, UML	1.3 **Archived**	1.3 **Archived**	Archived	Archived
4	Dev	BSc	EE	2-5	2-5	C High	Sketch, UML	1.2 **Selective**	1.2 **Selective**	Formalized	Formalized
13	Tstr	BSc	EE	6-10	2-5	Not applicable	Sketch, UML	1.2 **Selective**	1.2 **Selective**	Formalized	Formalized
15	Sys	BSc	EE	10+	6-10	Not applicable	Sketch	1.1 **Ad-hoc**	1.1 **Ad-hoc**	Performed	Performed
5	Dev	MSc	EE	10+	10+	C Very high	-	0.1 **Bad experienced**	0.1 **Bad experienced**	None	None

Totally, 204 years of software development experience.
**: this information was obtained during the interview or direct observation after the participant's completion of the questionnaire*

TABLE 3. ABBREVIATIONS USED IN TABLE 2

Dev: Software Developer/Programmer	**Arch:** Software Architect	**Desg**: Software Designer
Sys: Systems Engineer	**QA:** Quality Assurance Engineer	**Tstr:** Software Tester
CS: Computer Science	**CENG:** Computer Engineering	**SE:** Software Engineering
EE: Electrical/Electronics Engineering	**IS**: Information Systems	**ME**: Mechanical Engineering

Participant#11 tests UI (User Interface) application modules of the radar software project and mainly writes UI test simulators in Java or C++. He described the simulators he developed as "low" in terms of hardware closeness (Note that "*Hardware closeness*" is taken as the fact that firmware or DSP software is closer to hardware than UI or middleware software). He has also used their own MDE tool (which is based on their own DSL design) to generate test cases as model based testing (MBT). Therefore, he benefits from both UML diagrams and DSL-like diagrams during analysis, design and test phases of SDLC. Participant#12 tests the communication protocol parts and message interfaces between middleware and digital signal processing (DSP) modules of the radar software, which are deployed in the main processor card (not in PC). She described the simulators she developed as "medium" in terms of hardware closeness. She does not use any model-driven techniques although she took some modeling languages courses during her MSc in CENG. She benefits from sequence diagrams, use case diagrams and communication diagrams during analysis and test phases of SDLC. On the other hand, participant#13, whose academic background is different from other testers (i.e., he is an EE graduate and did not take any SE courses on modeling) tests DSP algorithms and he does not use any programming language directly related with modeling. Besides, he mentioned that he never uses any digital medium (e.g., PC) while modeling although he limitedly uses some use case or sequence diagrams just to communicate with other colleagues without archiving them (e.g., lifespan of these diagrams are shorter since they are soon discarded after conversation)As seen, although participant#11, participant#12 and participant#13 are in the same project with the same SE role, since their task/responsibilities are different (e.g., testing different modules of the same software), their modeling characteristics, hence their modeling patterns are different. Similar situations happened for the same SE roles (e.g., developers or systems engineers), which shows that the difference on modeling patterns might be related not only to project characteristics or roles in the project but also the tasks and responsibilities of that participant in that role besides formal education (e.g., university degree) of the stakeholder.

The results have organizational implications since it is beneficial to identify common techniques for different modeling purposes while pinpointing the potential challenges. For example, the results showed that even stakeholders in the same SE roles within the same projects might have different modeling practices (e.g., participant#11, participant#12 and participant#13 as being software tester). On the other hand, we also observed that different modeling stakeholders (e.g., systems engineers, test engineers and software engineers) might have common modeling approaches and they might be in the same pattern (as participant#1, participant#7, participant#10 and participant#12 are in "Descriptive" pattern although their SE roles are different). Such findings show that the need for further training in modeling or implementing practices of effective use of modeling in an organizational unit might change from case to case. In other words, MAPforES can be utilized to decide the best standardization approaches to identify the common techniques for different modeling characteristics such as purpose, medium type (e.g., modeling environment/tool), SDLC phases or modeling languages. For example, those stakeholders, who find UML too general or vague for their purposes [10] might actually benefit from DSL-like approaches in their choices of modeling languages to carry out effective MDE. The detection of such a need and further training can be made via MAPforES characteristics [13].

Moreover, during the first round of the interview, which is based on the questionnaire, participant#8 is at pattern 3.3 (i.e., "With DSL-like" pattern, see Table 1), but during the second round of the interview, face-to-face conversation revealed that he is one of the modeling stakeholders, who were "unaware" of MDE. The professionals, who were "unaware" of MDE filled the questionnaire as if they have benefitted from automatic code generation or documentation generation with sketch and UML usage. However, it was observed that they actually used DSL-like modeling languages, which categorizes them as pattern 3.x. For further details of participants' responses, see [14]. According to our results (e.g., including all other case studies), this "unawareness" related to software modeling is mainly based on stakeholder's profile (e.g., university degree, modeling experience, etc.). In other words, without a background and common terminology in modeling, the stakeholder may not be aware that she is not actually using modeling in SDLC. An organization might use such outputs of MAPforES to fulfill any training need to create and increase the awareness of what their modeling stakeholders do.

C. *Threats to Validity*

In this research, multiple sources of evidences were used with case study strategy. All evidences were collected in questionnaires, written notes after interviews and direct observations; and then were kept in a technical report [14]. During the second round of the interview, the evaluator confirmed what the interviewee gave as responses in the questionnaire to ensure the validity of the collected data. By this way, cross-checking of what the questionnaire gave in the first round and what the evaluator observed during the second round, provided more robust conclusions.

In order to mitigate internal validity threat, we focused on the study design and checked whether the results are consistent with the data. During the first part, all participants filled out the questionnaire individually and separately so that the interviewer prevented answers of a participant to be influenced by others [15]. By this way, the interviewer avoided any information sharing between interviewees.

The generalizability of the results is focused to mitigate external validity threat [16]. The participants were selected intentionally with variation points (e.g., position, academic background, experience, hardware closeness, etc). It cannot be stated that the selection is representative of other embedded

software development projects. However, all case studies (see [6]) have similar results and by applying the model in more case studies and projects, the generalizability shall be improved.

This study has a case study protocol and database, which were documented and archived systematically so that the replicability and repeatability of the operation of the case study have been ensured. Note that both questionnaires evaluation form of participants were saved in case study database as a paper repository and then were digitized during the analysis by taking the photo of each page [14].

IV. CONCLUSIONS

All qualitative and quantitative data gathered through evaluation forms (see Appendix of [6]) have shown that the model has been useful in:

- creating and increasing the awareness of what modeling stakeholders do,
- giving an opportunity to the stakeholder to compare and contrast what the similar profiles are doing while modeling,
- in suggesting on software modeling practices [14].

After this study, we observed that the industrial context reflects what we presented for modeling patterns, which focus on significant characteristics (e.g., not only “modeling rigor” but also “purpose”, “medium type used”, “stakeholder profile”, etc.) and fills the gap of what constitutes “software modeling” (e.g., including DSLs and other formal languages beyond UML). We found out that organizations may need different modeling patterns for different projects or even for different individual SE roles within projects. MAPforES provides an approach to provide feedback to modeling stakeholders thereby creating insight for individuals.

ACKNOWLEDGMENT

The authors would like to thank all embedded software professionals, who contributed to this study.

REFERENCES

[1] D. Akdur and V. Garousi, "Model-Driven Engineering in Support of Development, Test and Maintenance of Communication Middleware: An Industrial Case-Study," in *International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, France, 2015.

[2] D. Akdur, V. Garousi, and O. Demirörs, "Cross-factor analysis of software modeling practices versus practitioner demographics in the embedded software industry," in *6th Mediterranean Conference on Embedded Computing (MECO)*, Montenegro, 2017.

[3] R. Heldal, P. Pelliccione, U. Eliasson, J. Lantz, J. Derehag, and J. Whittle, "Descriptive vs prescriptive models in industry," in *ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems*, France, 2016.

[4] A. G. Kleppe, J. Warmer, and W. Bast, *MDA Explained: The Model Driven Architecture: Practice and Promise*: Addison-Wesley Longman Publishing Co., Inc., 2003.

[5] M. Petre, "UML in practice," in *35th International Conference on Software Engineering (ICSE)*, 2013, pp. 722-731.

[6] D. Akdur, "Modeling Patterns and Cultures of Embedded Software Development Projects," Thesis, Doctor of Philosophy (PhD), Information Systems, Middle East Technical University (METU), www.researchgate.net/publication/322701453_Modeling_Patterns_and_Cultures_of_Embedded_Software_Development_Projects, 2018.

[7] D. Akdur, V. Garousi, and O. Demirörs, "A survey on modeling and model-driven engineering practices in the embedded software industry," *Journal of Systems Architecture* vol. 91, pp. 62-82, 2018.

[8] D. Akdur, O. Demirörs, and V. Garousi, "Characterizing the development and usage of diagrams in embedded software systems," in *43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Vienna, Austria, 2017.

[9] D. Akdur and O. Demirors, "Modeling Patterns and Cultures of Embedded Software Development Projects: Towards Preliminary Characterization Model," in *12nd Turkish National Software Engineering Symposium (In Turkish: Ulusal Yazılım Mühendisliği Sempozyumu (UYMS))*, İstanbul, Turkey, 2018.

[10] D. Akdur, O. Demirörs, and B. Say, "Towards Modeling Patterns for Embedded Software Industry: Feedback from the Field," in *44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Prag, Czech Republic, 2018.

[11] D. Akdur, V. Garousi, and O. Demirörs, "MDE in embedded SW industry-Raw survey data," *https://dx.doi.org/10.6084/m9.figshare.4262972,* 2015, Last accessed: Nov. 27, 2016.

[12] B. A. Kitchenham, S. Linkman, and D. Law, "DESMET: A Methodology for Evaluating Software Engineering Methods and Tools," *Computing and Control Engineering Journal,* 1997.

[13] D. Akdur. (2017, Last accessed: September 16, 2018). *Online Dataset: Multiple Case Study Results*. Available: https://www.researchgate.net/publication/323810535_Multiple_Case_Study_Results_-_Applications_of_the_Modeling_Patterns_for_Embedded_Software_Development

[14] D. Akdur and O. Demirörs, "Multiple Case Studies to Validate Modeling Patterns and Cultures of Embedded Software Development Projects, Technical Report," METU, METU/II-TR-2017-90, 2017.

[15] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. Hoaglin, K. El Emam, *et al.*, "Preliminary guidelines for empirical research in software engineering," presented at the IEEE Transactions on Software Engineering, 2002.

[16] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*: Springer Berlin Heidelberg, 2012.