Evaluation of a Specification Approach for Vehicle Functions using Activity Diagrams in Requirements Documents

Martin Beckmann, Andreas Vogelsang[1]

Abstract:

Rising complexity of systems has long been a major challenge in requirements engineering. This manifests in more extensive and harder to understand requirements documents. At the Daimler AG, an approach is applied that combines the use of activity diagrams with natural language specifications to specify vehicle functions. The approach starts with an activity diagram that is created to get an early overview. The contained information is then transferred to a textual requirements document, where details are added and the behavior is refined. While the approach aims at reducing efforts needed to understand a function's behavior, the application of the approach itself causes new challenges on its own. By examining existing specifications at Daimler, we identified nine categories of inconsistencies and deviations between activity diagrams and their textual representations. This paper extends a previous case study on the subject by presenting additional data we acquired. Our analysis indicates that a coexistence of textual and graphical representations of models without proper tool support results in inconsistencies and deviations that may cause severe maintenance costs or even provoke faults in subsequent development steps.

Keywords: UML, Activity Diagrams, Vehicle Function Specification

1 Introduction

Complex software systems, which can be found for example in distributed embedded systems in automotive electronics, require model-based and system-oriented development approaches [Br06]. Thus, so-called executable specifications are common practice in the automotive industry. Using graphical models for specification manages complexity and improves reusability and analytical capabilities [AD97]. Although graphical models provide a suitable means to specify and understand dependencies and procedural behavior of a system, in industry, they are usually accompanied by a textual representation. Previous work has shown the need for a continuous systems engineering environment, where referring or constitutive documents are essential to work on complex software systems [Re15]. Also, the combined use of graphical diagrams and textual descriptions is considered beneficial for the requirements management process [STP12]. In addition, for industrial applications, tool support and model exchange for graphical models is still not standardized and, as a

[1] Technische Universität Berlin, Daimler Center for Automotive IT Innovations, Ernst-Reuter-Platz 7, 10587 Berlin, {martin.beckmann, andreas.vogelsang}@tu-berlin.de

result, manufacturer/supplier handover is still performed based on textual documents. This is especially important since these textual documents often serve as the basis for legal considerations between the contractors [STP12]. Also, due to different backgrounds of the stakeholders, not everyone is capable of understanding the graphical models [AEQ98]. Thus, the information contained in a model, needs to be written in words to be appropriately reviewable [Go04].

Daimler applies an approach, where, as a first step, a UML activity diagram [Ob15] is created for each vehicle function to describe the function's activation and deactivation by triggers and conditions. This kind of description is also known in literature to formulate textual natural language requirements [Fi04]. Textual representations of the activity diagrams along with the diagrams themselves are then transferred into a requirements document for everyone to understand and for ongoing development. The transfer of the model into the requirements document is done manually. This is an error-prone task. Besides, the ongoing development using the requirements document might also cause inconsistencies between the activity diagrams and the document if the activity diagram is not kept up-to-date.

We are interested in understanding what types of inconsistencies and quality issues are introduced by this approach and how severe these issues are. If the approach itself introduces more severe issues than expected benefits, this is a strong argument for automation and quality assurance.

For this purpose, we examined 36 vehicle functions of one system at Daimler that was specified by the introduced approach. As a result, a number of inconsistencies between the requirements document and the activity diagram were found.

All of these findings resulted in nine different categories of quality issues. We found occurrences of these categories in all of the examined functions. The categories are introduced in detail as well as the amount of findings in the examined system. Also, we presented the quality issues to different stakeholders of the system, who assessed their severity. We found that 78% of the vehicle functions contained quality issues that were assessed as major issues. This paper complements previous findings on the same system [BVR17] by presenting additional data that was acquired and comparing it with the previous findings.

2 Background

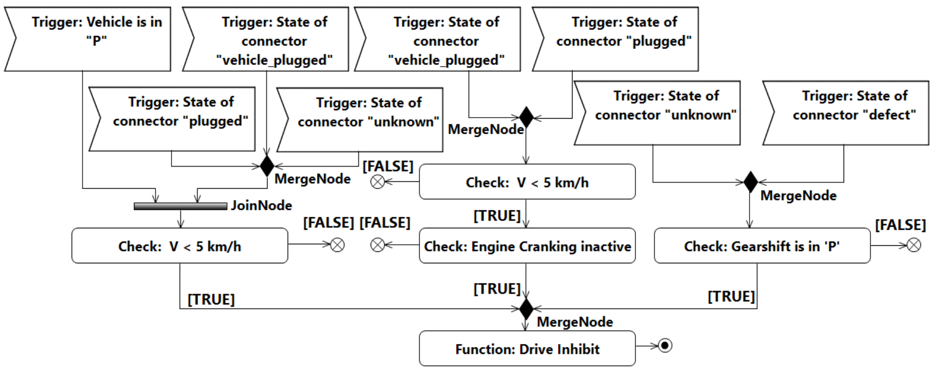
The Daimler approach used to specify functions of a system employs UML activity diagrams. Creating activity diagrams is the first step of specifying a new vehicle function. They are used to get an early overview of the desired function behavior with a special focus on the functions activation, execution conditions, functional paths, and deactivation. The information contained in the activity diagram is then transferred to a textual requirements document. This transfer is necessary since this textual requirements document is the central artifact for further development. Besides, the textual document contains additional and more detailed information as well as statements about its context, which relates this approach to Literate Modeling [AEQ98].

Fig. 1 shows an exemplar specification as we have found it at Daimler. The example consists of an activity diagram and its textual representation in the requirements document. In the following, we will explain the example and also the contained quality issues. In the remainder of this work, an element refers to an entity contained in an activity diagram, whereas an object in the text refers to an entity contained in the requirements document.

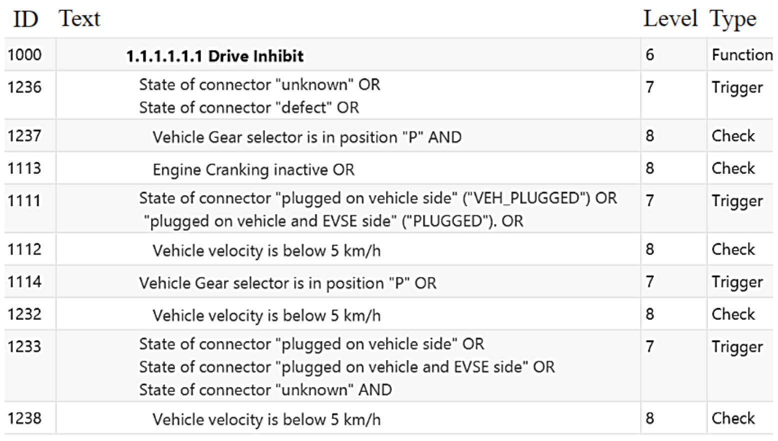
Fig. 1a displays the activity diagram of the function *Drive Inhibit*. The actual behavior of the activated function is described in the *Action* node labeled with *Drive Inhibit* (bottom of the diagram). The function's activation is described by a combination of triggers and checks for conditions. For triggers, the *AcceptEventAction* element is used. The checks are modeled as *Action* elements. If the condition of a check is not fulfilled, the flow ends (*FlowFinal*). The triggers and checks are connected by *ControlNodes* such as *JoinNodes* and *MergeNodes*. *JoinNodes* act as synchronization points and can be interpreted as AND operators in terms of propositional logic. *MergeNodes* represent OR operators. Once the actual functionality of the function is executed, *ActivityFinal* elements designate the end of an activity.

The corresponding chapter in the textual requirements document is displayed in Fig. 1b. Each row in the document represents an object, which is described by a set of attributes (columns). The *ID* attribute contains a unique identifier of the object. The *Text* attribute is a textual description of the object and is supposed to be equal to the text of the corresponding element in the activity diagram. The *Level* is an attribute to structure the document hierarchically. It is derived from the structure of the activity diagram. The *Type* attribute of each text object is supposed to be equal to the type of its corresponding element in the diagram. These attributes are needed to display the relevant information of the activity diagram in the requirements document. Besides the given attributes, the document contains additional attributes used for further development.

There are many possibilities to display different aspects of an activity diagram as text. An exact textual representation as presented in [FMC09] is not desirable, since it lacks proper readability and comprehensibility for those unfamiliar with activity diagrams. Instead, the used textual representation focuses on the propositional logic, readability, and the recognition value of the structure of the activity diagram. This is implemented by copying the text of the elements of the diagram into distinct objects. Propositional logic operators such as OR and AND are used as strings in the *Text* attributes of the objects to represent the logic statements of the activity diagram. The operators at the end of an object's text connect the object with the following object on the same level of the document hierarchy. For instance, in Fig. 1b, the object with *ID 1236* is connected via an OR with the object with *ID 1111* because it is the next object on the same hierarchical level. Besides the propositional logic purposes, the different levels of the documents are used to display the belonging of the elements within the activity diagram. For example, the check *Vehicle Gear selector is in*


Trigger: Vehicle is in "P"
Trigger: State of connector "vehicle_plugged"
Trigger: State of connector "vehicle_plugged"
Trigger: State of connector "plugged"
Trigger: State of connector "plugged"
Trigger: State of connector "unknown"
MergeNode
Trigger: State of connector "unknown"
Trigger: State of connector "defect"
MergeNode
[FALSE]
Check: V < 5 km/h
MergeNode
JoinNode
[FALSE] [FALSE]
[TRUE]
Check: V < 5 km/h
Check: Engine Cranking inactive
Check: Gearshift is in 'P'
[FALSE]
[TRUE]
[TRUE]
[TRUE]
MergeNode
Function: Drive Inhibit

(a) Activity diagram of the Function *Drive Inhibit*



ID	Text	Level	Type
1000	**1.1.1.1.1.1 Drive Inhibit**	6	Function
1236	State of connector "unknown" OR State of connector "defect" OR	7	Trigger
1237	Vehicle Gear selector is in position "P" AND	8	Check
1113	Engine Cranking inactive OR	8	Check
1111	State of connector "plugged on vehicle side" ("VEH_PLUGGED") OR "plugged on vehicle and EVSE side" ("PLUGGED"). OR	7	Trigger
1112	Vehicle velocity is below 5 km/h	8	Check
1114	Vehicle Gear selector is in position "P" OR	7	Trigger
1232	Vehicle velocity is below 5 km/h	8	Check
1233	State of connector "plugged on vehicle side" OR State of connector "plugged on vehicle and EVSE side" OR State of connector "unknown" AND	7	Trigger
1238	Vehicle velocity is below 5 km/h	8	Check

(b) Textual specification of the Function *Drive Inhibit*

Fig. 1: Activity diagram and the specification text of a function

position "P" (*ID 1237*) is executed after one of the triggers contained in the object with the *ID 1236* occurred. Hence, it appears one level below. This is important since there might be more than one check associated with a set of triggers, as can be seen in the object in the text with *ID 1113*.

The transfer of information from the activity diagram to the requirements document is a manual process. This might lead to inconsistencies between the activity diagram and the requirements document and other quality issues as can be seen in Fig. 1. Amongst others, these inconsistencies and quality issues are presented in the next chapter.

3 Identified Quality Issues

A preliminary examination of a set of requirements documents at Daimler revealed a number of quality issues. We grouped these quality issues into nine different categories. The categories and their descriptions are listed in Tab. 1. The categories cover the relation between the activity diagrams and the requirements document. Some of them only appear either in the diagram or the text but still have an influence on the respective other artifact. We will explain some of the categories by examining the example in Fig. 1.

Category name	**Description**
Missing Tracing	There is no information to trace an element to its corresponding object in the text or vice versa.
Missing Element/ Object	Either the activity diagram or the requirements document contains entities, which the other does not contain.
Incorrect Logic	The propositional logic of the activity diagram deviates from the requirements document or the logic connections are not clear.
Textual Differences	Elements and their corresponding objects in the text exhibit textual differences.
Redundant Element	The activity diagram contains multiple elements, which have the same type and the same text.
Non Atomic Element/Object	Either an element or an object contains multiple statements. This might appear in both the requirements document or the activity diagram.
Wrong Placement	The placement of an element in the activity diagram does not match with the placement of the corresponding object in the requirements document.
Unnecessary Repetition	There are multiple objects in the requirements document, which are derived from one single element.
Wrong Type	The type of the element does not match the type of the corresponding object.

Tab. 1: Categories

Textual Differences can be found (amongst others) between the triggers in the objects with the *ID 1111, 1233* and their corresponding elements of the diagram. While all elements of the diagram are atomic, the requirements document contains several **Non Atomic Objects** (*ID 1236, 1111, 1233*). These objects incorporate multiple assertions that are connected by propositional logical operators. This is both an issue in the requirements document as well as a deviation between the activity diagram and the requirements document. There are multiple **Redundant Elements** in the diagram such as the checks *V < 5 km/h* and the triggers *State of connector "plugged"*. In this example, the appearance of redundant elements in the diagram can be avoided by inserting additional *ControlNodes* and restructuring the activity diagram [Be17]. The category **Incorrect Logic** is present in the objects in the document with the *ID 1113* and *ID 1233*. Both objects end with an operator, for which it is not clear which object they refer to. Neither of them has a successor on the same *level* below their respective parent object. The object with the *ID 1236* is the parent object of two objects (*ID 1237, 1113*) containing checks. The elements in the diagram, corresponding to the objects with the *ID 1236*, are followed by the diagram element *Check: Gearshift is in 'P'*. In the document the corresponding object (*ID 1236*) is the parent object of an additional check (*ID 1113*). The additional check in the document is elsewhere in the activity diagram. This situation is denoted as the category **Wrong Placement**. The requirements document contains *Check: V < 5 km/h* three times (*ID 1112, 1232, 1238*). But there are only two elements in the activity diagram. Hence, two of the objects in the document refer to one single element in the diagram. This is an instance of the **Unnecessary Repetition** category and can be avoided by grouping the objects accordingly.

4 Previous Results

In a previous study [BVR17], we analyzed a system consisting of 36 activity diagrams and identified several quality issues. The examined subsystem is responsible for charging the high-voltage batteries of Plug-in Hybrid Electric Vehicles and Battery Electric Vehicles. As such the system contains requirements that are relevant for safety as well as for usability. The number of findings for each category are shown in Tab. 2.

Category name	**Findings**	**Number and ratio of functions with findings**	**Average number of findings per function**
Missing Tracing	all[2]	36 (100 %)	-
Missing Element/Object	126	28 (78 %)	3.5
Incorrect Logic	29	22 (61 %)	0.8
Textual Differences	43	20 (56 %)	1.2
Redundant Element	24	15 (42 %)	0.66
Non Atomic Element/Object	18	14 (39 %)	0.5
Wrong Placement	18	10 (28 %)	0.5
Unnecessary Repetition	15	9 (25 %)	0.42
Wrong Type	10	8 (22 %)	0.28

Tab. 2: Occurrences of quality issues per category ordered by frequency of occurrences in functions.

The results show that we found at least 10 occurrences of each category in the 36 examined functions. Moreover, the Missing Tracing occurred in all elements of all functions, which means that we found no trace links to diagram elements at all. In addition, the categories Missing Element/Object, Textual Differences and Incorrect Logic are the most frequent quality issues. All of these categories appear in more than half of the analyzed vehicle functions. The category appearing the least often is Wrong Type with 10 occurrences. In a

[2] In the examined specifications, no tracing links between diagram elements and textual objects were defined.

follow up survey, we presented two existing examples of each category to seven stakeholders of the system to evaluate the severity of the categories. They were asked to answer three questions:

1. Were you aware of the existence of this finding?
2. Do you think this sample is in fact a quality issue?
3. When would you fix this quality issue?

The first two question could be answered with *yes* or *no*. For the third question, we gave the following options: *immediately*, *soon*, *long term*, *never*. The option *immediately* representing the most severe answer and *never* representing the least severe answer.

From the answers, we concluded that our defined categories Wrong Placement, Missing Element/Object, and Wrong Type are in fact quality issues and were rated as the most severe categories. One of the samples of Incorrect Logic is also considered very severe. The perception of the other sample was more diverse. Two stakeholders mentioned that the presented situation does not require addressing the issues, while more than half of the stakeholders would fix the issues at least *soon*. Therefore, we consider Incorrect Logic as an important quality issue. Textual Differences also reached high levels of agreement and no one mentioned that fixing would *never* be needed. A majority perceived Missing Tracing as a quality issue. Nevertheless, two stakeholders mentioned that this does not need fixing, while the rest mentioned it needs to be fixed at least *soon*. The opinions on the categories Redundant Element, Non Atomic Element/Object, and Unnecessary Repetition diverged. This challenges our initial hypothesis that findings of these three categories are indeed quality issues. Still, except for one sample, a majority of stakeholders stated that these findings should be fixed at least *soon*.

5 New Results

Apart from answering our initial survey, four of the stakeholders agreed to complete an additional survey. In that survey, we wanted the stakeholders to rank the severity of the categories. They were asked to rank the categories by conducting a pairwise comparison. For the nine categories this results in 36 necessary pairwise comparisons. For each pair, they had to decide whether two categories are equally severe or whether one is more severe than the other. If one category is rated more severe than the other, the more severe category is attributed with 1 point while the other is attributed with -1 point. In case of equal severity, both categories are attributed with 0 points. The final ranking results from adding up the points of all comparisons for each category. Hence, the category with the most points is the most severe. Besides, the stakeholders had the possibility to add comments as a free-form text at the end of the survey.

Tab. 3 shows the results for each stakeholder. They are ordered by severity starting at the top with the one that is perceived as the most severe. If multiple categories are not separated by a horizontal line, these categories are perceived as equally severe. The column for the combined results was calculated by summing up the points for each category for all three stakeholders.

Stakeholder 1	**Stakeholder 2**	**Stakeholder 3**	**Combined**
Missing Element/ Object	Textual Differences	Missing Element/ Object	Missing Element/ Object
Wrong Type	Missing Tracing	Textual Differences	Textual Differences
Incorrect Logic	Missing Element/ Object	Incorrect Logic	Incorrect Logic
Missing Tracing	Wrong Type	Wrong Type	Wrong Type
Wrong Placement	Incorrect Logic	Wrong Placement	Missing Tracing
Unnecessary Repetition	Wrong Placement	Unnecessary Repetition	Wrong Placement
Non Atomic Element/ Object	Unnecessary Repetition	Non Atomic Element/ Object	Unnecessary Repetition
Redundant Element	Non Atomic Element/ Object	Redundant Element	Non Atomic Element/ Object
Textual Differences	Redundant Element	Missing Tracing	Redundant Element

Tab. 3: Ranking of quality issue categories for each stakeholders in descending order of their assessed severity

One of the four mentioned stakeholders did not fill in the pairwise comparison and just left a comment instead, saying that deviations between an activity diagram and the textual requirements specification are always severe quality issues regardless of their category. The same person also noted that this is especially important for testing activities since it might impact the results of tests.

Stakeholder 2 (see Tab. 3) stated that complete and unambiguous requirements are essential and hence the requirements must be traceable to elements in complementing diagrams with absolute certainty. This comment is in line with the ranking that emphasizes the importance of tracing and the possibility to match model elements with a specification object by their text.

Considering the individual rankings of the stakeholders, it has to be noted, that the category Missing Element/Object is considered to have a more severe impact than the other categories. *Stakeholder 2* did not differentiate specifically between the categories. As a result, there are only four different ranks. Still, we conclude that the three stakeholders put an emphasis on the category Incorrect Logic being among the most severe as well. Besides, neither the category Wrong Type nor Wrong Placement are among the last three positions of any stakeholder.

The most disagreement occurred for the categories Textual Differences and Missing Tracing. The former is ranked as the most severe by two stakeholders and as the least severe by one stakeholder. The situation is similar for the latter. Missing Tracing is considered to be the second and third most severe category by two stakeholders, while it is considered to be the least severe by one stakeholder. Other than that, the categories Redundant Element, Non Atomic Element/Object and Unnecessary Repetition received low rankings by all of the stakeholders.

The combined results confirm the impression of the individual rankings that Missing Element/Object is the most severe category. Textual Differences on the other hand received a high rank because of the emphasis of *Stakeholder 2* on the category. This is in contrast to the individual rankings, where one stakeholder considered the category as the least severe. The other categories in the combined results reflect the perception of the individual rankings since these categories were ranked similarly by the stakeholders.

6 Comparison of the Results

Overall, the ranking is in line with the results of our previous survey. Missing Element/Object is among the categories perceived as the most severe by the ranking and by the previous survey. Wrong Placement on the other hand, which was perceived as a quality issue by all participants and rated as the most severe in our original survey is no longer perceived as the most severe. Still, it is among the top three of the categories for all stakeholders and in the combined results. The findings for Textual Differences are also confirmed as the individual rankings as well as the previous survey showed disagreement on the severity of the category.

Missing Tracing is also perceived differently regarding its severity during the previous survey. Its position in the middle of the combined ranking merely shows this disagreement. The ranking of Incorrect Logic and Wrong Type confirms the original findings. Both categories were perceived as quality issues by almost all of the stakeholders and rated as very severe as the majority stated the instances of the categories need to be fixed *soon*.

Redundant Element, Non Atomic Element/Object, and Unnecessary Repetition remain the categories that are perceived as the least severe. These categories received the lowest agreement on being quality issues in the first place. Hence, with the exception of a few stakeholders, no one saw the need to fix these quality issues *immediately*. Since these categories take the last three places, the ranking also reflects these previous findings.

The new results confirm the findings of the previous survey. Thus, the conclusion that the identified categories have a severe impact on the quality of the specified vehicle functions remains the same. Since the most severe category Missing Element/Object already appears in 78 % of the investigated functions, it is necessary to address the issues.

7 Possible Solutions

To avoid quality issues resulting from deviations between graphical and textual representations, automatic approaches can be used to keep the diagram and the requirements document in sync. The generation of requirements documents from graphical models is an established approach [NT09]. Different approaches were suggested for specific graphical models. For instance, Maiden et al. [Ma05] use i* models to derive requirements and De Landtsheer et al. [DLLVL04] propose a similar approach for the KAOS goal-oriented method. Fockel and Holtmann [FH14] present a model-driven RE approach with tool support that provides synchronization capabilities for the applied RE models and their textual representations. Still, these approaches are specialized to specific techniques during requirements elicitation and management. Other than that, there are also approaches, that derive textual requirements or a structure for parts of requirements documents from different types of UML/SysML diagrams. Robinson-Mallett [RM12] shows how Statecharts and Block Diagrams can be used to automatically create a structure for a requirements document. Berenbach [Be03] introduces an algorithm that derives a structure for a requirements document from use case diagrams. Since all of these approaches do not use activity diagrams, they are not applicable to the presented specification approach. The approach presented by Drusinsky [Dr08] supports activity diagrams, however, only for UML-1. Additionally, only the generation of actual requirements is addressed but not the creation of a requirements document structure. Another possibility is the use of Projectional Editors, which automatically edit different projections of a common underlying model, in this case the activity diagram and its textual representation. However, this possibility may require substantial efforts and experienced developers [Be16]. Hence, a custom-made and lightweight synchronization solution might be well suited to prevent the mentioned quality issues for the used specification approach. A supporting tool would need direct access to the modeling and requirements engineering tool to extract certain aspects (e.g. propositional logic relations, functional paths) from the diagrams for use in the document.

8 Threats to Validity

The rankings were obtained by asking stakeholders to perform a pairwise comparison of the presented quality issues. The stakeholders we asked already participated in the initial case study. Hence, they had prior knowledge of the possible quality issues. The knowledge is necessary to complete the ranking but might also have had an influence on the decisions the stakeholders made.

Besides, only four of the original seven stakeholders participated. This strengthens the threat that the result are highly influenced by individual persons and can only give an impression. Thus, the conclusion, that the findings of the initial case study are confirmed is challenged by the small number of participants. As a result, it cannot be assumed that the findings are generalizable. In order to achieve generalizability, the case study needs to be repeated with additional stakeholders and under consideration of more than one system.

9 Conclusion and Outlook

In this paper, we presented possible quality issues that may arise when using a certain specification approach that is used at Daimler. The approach incorporates UML activity diagrams in requirements documents. Those activity diagrams are accompanied by textual representations of the diagrams. The textual representation is edited and further refined during ongoing development.

In a previous case study on a real system, we assessed the total number of occurrences of possible quality issues in the requirements document of the system. Also, we conducted a survey to find out whether they agree that the quality issues we identified are in fact quality issues and how they rate the severity of preselected samples. In a following survey, we asked the same stakeholders to rank the presented quality issues by severity. The aim was to gain further insights on whether the previous findings are in line with new results. We found that the ranking of the quality issues confirm our previous findings. Differences occur in the perception of the category Wrong Placement, which was considered more severe during our initial case study. Although less stakeholders participated in this case study, the disagreement on a number of quality issues is also reflected in the rankings of the individual stakeholders.

Since there was only one system available, the generalizability is limited. The findings need to be validated by repeating the case study with a different system. An aspect that was out of scope of this work, is the influence of the identified quality issues on following development stages. The case study assessed the number of occurrences in a certain requirements document and the severity of the quality issues but not the resultant consequences. Hence, it needs to be addressed how these quality issues effect the development of the final product and future products, that reuse the existing requirements document.

Some of the problems resulted from the way the textual representations of the activity diagrams are structured. Ongoing research is about which different presentations might be better in terms of avoiding the quality issues as well as maintaining proper readability.

References

[AD97] Apfelbaum, Larry; Doyle, John: Model Based Testing. In: Software Quality Week Conference. 1997.

[AEQ98] Arlow, Jim; Emmerich, Wolfgang; Quinn, John: Literate Modelling Capturing Business Knowledge with the UML. In: International Conference on the Unified Modeling Language. Springer, 1998.

[Be03] Berenbach, Brian: The Automated Extraction of Requirements from UML Models. In: 11th IEEE International Requirements Engineering Conference (RE). 2003.

[Be16] Berger, Thorsten; Völter, Markus; Jensen, Hans Peter; Dangprasert, Taweesap; Siegmund, Janet: Efficiency of Projectional Editing: A Controlled Experiment. In: 24th ACM

SIGSOFT International Symposium on Foundations of Software Engineering (FSE). 2016.

[Be17] Beckmann, Martin; Michalke, Vanessa; Vogelsang, Andreas; Schlutter, Aaron: Removal of Redundant Elements within UML Activity Diagrams. In: ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems. 2017.

[Br06] Broy, Manfred: Challenges in automotive software engineering. In: Proceedings of the 28th international conference on Software engineering. ACM, NY, USA, 2006.

[BVR17] Beckmann, Martin; Vogelsang, Andreas; Reuter, Christian: A Case Study on a Specification Approach using Activity Diagrams in Requirements Documents. 2017. 25th IEEE International Requirements Engineering Conference.

[DLLVL04] De Landtsheer, Renaud; Letier, Emmanuel; Van Lamsweerde, Axel: Deriving tabular event-based specifications from goal-oriented requirements models. Requirements Engineering, 9(2), 2004.

[Dr08] Drusinsky, Doron: From UML activity diagrams to specification requirements. In: IEEE International Conference on System of Systems Engineering (SoSE). 2008.

[FH14] Fockel, Markus; Holtmann, Jörg: A requirements engineering methodology combining models and controlled natural language. In: 4th International Model-Driven Requirements Engineering Workshop (MoDRE). IEEE, 2014.

[Fi04] Firesmith, Donald: Generating Complete, Unambiguous, and Verifiable Requirements from Stories, Scenarios, and Use Cases. Journal of Object Technology, 3(10), 2004.

[FMC09] Flater, David; Martin, Philippe; Crane, Michelle: Rendering UML Activity Diagrams as Human-Readable Text. Technical report, 2009.

[Go04] Goldsmith, Robin F: Discovering Real Business Requirements for Software Project Success. Artech House, 2004.

[Ma05] Maiden, Neil AM; Manning, Sharon; Jones, Sara; Greenwood, John: Generating requirements from systems models using patterns: a case study. Requirements Engineering, 10(4), 2005.

[NT09] Nicolás, Joaquín; Toval, Ambrosio: On the Generation of Requirements Specifications from Software Engineering Models: A Systematic Literature Review. Information and Software Technology, 51(9), 2009.

[Ob15] Object Management Group (OMG): , OMG Unified Modeling Language (OMG UML), Version 2.5. OMG Document Number formal/2015-03-01 (`http://www.omg.org/spec/UML/2.5/`), 2015.

[Re15] Reuter, Christian: Variant Management as a Cross-Sectional Approach for a Continuous Systems Engineering Environment. In: Proceedings of the 8th Grazer Symposium Virtual Vehicle. 2015.

[RM12] Robinson-Mallett, Christopher L: An Approach on Integrating Models and Textual Specifications. In: 2nd International Model-Driven Requirements Engineering Workshop (MoDRE). 2012.

[STP12] Sikora, Ernst; Tenbergen, Bastian; Pohl, Klaus: Industry needs and research directions in requirements engineering for embedded systems. Requirements Engineering, 17(1), 2012.