
HAZARD RELATION DIAGRAMS

DEFINITION AND EVALUATION

Bastian Tenbergen

HAZARD RELATION DIAGRAMS

DEFINITION AND EVALUATION

DISSERTATION

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften
(Dr. rer. Nat.)

durch die Fakultät für Wirtschaftswissenschaften der
Universität Duisburg-Essen
Campus Essen

vorgelegt von

Bastian Tenbergen

aus Hamminkeln, geboren in Wesel (Deutschland)

Essen (2017)

Tag der mündlichen Prüfung: 29.08.2017
Erstgutachter: Prof. Dr. Klaus Pohl
Zweitgutachter: Prof. Dr. Dr. h.c. Stefan Jähnichen

»*Quos Hamminceln iunxit...*«

Abstract

The development process of safety-critical, software-intensive embedded systems is characterized by the need to identify hazards during safety assessment in early stages of development. During operation, such hazards may lead to harm to come to humans and external systems in the form of death, injury, damage, or destruction, respectively. In order to improve the safety of the system during operation, mitigations are conceived for each hazard, and documented during requirements engineering by means of hazard-mitigating requirements. These hazard-mitigating requirements must be adequate in the sense that they must specify the functionality required by the stakeholders and must render the system sufficiently safe during operation with regard to the identified hazards.

The adequacy of hazard-mitigating requirements is determined during requirements validation. Yet, the validation of the adequacy of hazard-mitigating requirements is burdened by the fact that hazards and contextual information about hazards are a work product of safety assessment and hazard-mitigating requirements are a work product of requirements engineering. These work products are poorly integrated such that the information needed to determine the adequacy of hazard-mitigating requirements are not available to stakeholders during validation. In consequence, there is the risk that inadequate hazard-mitigating requirements remain covert and the system is falsely considered sufficiently safe.

In this dissertation, an approach was developed, which visualizes hazards, contextual information about hazards, hazard-mitigating requirements, as well as their specific dependencies in graphical models. The approach hence renders these information accessible to stakeholders during validation. In addition, an approach to create these graphical models was developed and prototypically implemented. Moreover, the benefits of using these graphical models during validation of hazard-mitigating requirements was investigated and established by means of four detailed empirical experiments.

The dissertation at hand hence provides a contribution towards the integration of the work products of safety assessment and requirements engineering with the purpose to support the validation of the adequacy of hazard-mitigating requirements.

Zusammenfassung

Der Entwicklungsprozess sicherheitskritischer, software-intensiver eingebetteter Systeme wird im Besonderen durch die Notwendigkeit charakterisiert, zu einem frühestmöglichem Zeitpunkt im Rahmen des Safety Assessments sogenannte Hazards aufzudecken, welche im Betrieb zu Schaden in Form von Tod oder Verletzung von Menschen sowie zu Beschädigung oder Zerstörung externer Systeme führen können. Um die Sicherheit des Systems im Betrieb zu fördern, werden für jeden Hazard sogenannte Mitigationen entwickelt, welche durch hazard-mitigierende Anforderungen im Rahmen des Requirements Engineering dokumentiert werden. Hazard-mitigierende Anforderungen müssen in dem Sinne adäquat sein, dass sie zum einen die von Stakeholdern gewünschte Systemfunktionalität spezifizieren und zum anderen die Wahrscheinlichkeit von Schaden durch Hazards im Betrieb minimieren.

Die Adäquatheit von hazard-mitigierenden Anforderungen wird im Entwicklungsprozess im Rahmen der Anforderungvalidierung bestimmt. Die Validierung von hazard-mitigierenden Anforderungen wird allerdings dadurch erschwert, dass Hazards sowie Kontextinformationen über Hazards ein Arbeitsprodukt des Safety Assessments darstellen und die hazard-mitigierenden Anforderungen ein Arbeitsprodukt des Requirements Engineering sind. Diese beiden Arbeitsprodukte sind in der Regel nicht schlecht integriert, sodass den Stakeholdern bei der Validierung nicht alle Informationen zur Verfügung stehen, die zur Bestimmung der Adäquatheit der hazard-mitigierenden Anforderungen notwendig sind. In Folge könnte es dazu kommen, dass Inadäquatheit in hazard-mitigierenden Anforderungen nicht aufgedeckt wird und das System fälschlicherweise als ausreichend sicher betrachtet wird.

Im Rahmen dieses Dissertationsvorhabens wurde ein Ansatz entwickelt, welcher Hazards, Kontextinformationen zu Hazards, hazard-mitigierende Anforderungen sowie die spezifischen Abhängigkeiten in einem graphischen Modell visualisiert und somit für die Validierung zugänglich macht. Zudem wird ein automatisierter Ansatz zur Generierung der graphischen Modelle vorgestellt und prototypisch implementiert. Darüber hinaus wird anhand von vier detaillierten empirischen Experimenten der Nutzen der graphischen Modelle für die Validierung hazard-mitigierender Anforderungen nachgewiesen.

Die vorliegende Arbeit leistet somit einen Beitrag zur Integration der Arbeitsergebnisse des Safety Assessments und des Requirements Engineering mit dem Ziel die Validierung der Adäquatheit hazard-mitigierender Anforderungen zu unterstützen.

Acknowledgements

For many years, I have enjoyed the emotional and scientific help of many individuals, to whom, undeniably, I am indebted. Foremost, I would like to thank Klaus Pohl for conversations and critical feedback, which helped me improve my work. I also thank Stefan Jähnichen, who was kind, able, and willing to serve as the second reviewer of my work given the particular constraints at the time. I also thank Michael Goedicke for chairing the examination committee.

Years of devotion were invested by my colleague and ex-supervisor Thorsten Weyer. His outstanding guidance, scientific advisement, and wisdom have been and still are a source of unique insights and improvement to me. This work in large parts is the result of his many, many pages of comments, thought-provoking debates, and critical reflection. Thank you, Thorsten.

Key individuals, whom I am deeply indebted to include “the other half of my brain,” my long-time friend and colleague Marian Daun. Marian and I shared more than an office. We were allowed to share triumph, defeats, and a love for what we do. Together, we started new exciting undertakings, reviewed each other’s work, and were a source of ideas to each other. He and I share a mutual understanding and trust that I will always cherish.

During my time in Essen, I was lucky to work with a “lighthouse” full of talented individuals: Moritz, Torsten, Jennifer, Christiane, Philipp B., Johanna, Andreas F., Andreas G., Nazila, Heike, André, Frederike, Tim, Tobias, Kevin, Andreas M., Ovis, Mark, Andrea, Philipp S., Eric, Heiko, Vanessa, Nelufar, and many more. Thank you for all the things—small and big.

I will always be deeply grateful for my colleagues and mentors Doug Lea and David Vampola of SUNY Oswego, who facilitated my scientific career in the first place.

Every day, I can count on my “brothers” Sebastian Michelbrink and Christian Lübbers, their understanding, their help, their trust, and their friendship. Without them, I am nothing. I am also indebted to those that bring together, what time cannot separate: Vanessa and Matthias, Melanie, Jennifer, Daniela and Raimo, Astrid and Achim, Thomas and Enya.

Undying love and gratitude is reserved for my family, especially my parents Christa and Wolfgang Tenbergen. You have enabled me in so many ways, with so few words, but through so many actions, such that I may do so many things. Simply thanking you will never be enough.

Words fail to express the appreciation and gratefulness I feel for my loving wife Gilian. Your companionship and sacrifice is beyond what I deserve. You are there when I need you. You shall always have my spirit.

Bastian Tenbergen

Hamminkeln, Germany, and Oswego, NY, in Summer 2017

Table of Content

Abstract	I
Zusammenfassung	III
Acknowledgements	V
Table of Content	VI
List of Abbreviations	VIII
List of Figures	IX
List of Tables	X
Part I: Introduction	1
1. Motivation and Goal	3
1.1. The Early Development Stages of Safety-Critical Systems	3
1.2. Challenges in Hazard-Mitigating Requirements Adequacy Validation	5
1.3. Assumptions	7
1.4. Goal and Solution Idea	9
1.5. Overview of the Approach	10
1.6. Outline	12
2. Fundamentals	13
2.1. Running Example – The Adaptive Cruise Control System	13
2.2. Terminology	15
3. State of the Art	21
3.1. Evaluation Framework for the State of the Art	21
3.2. Safety Requirements Elicitation	23
3.3. Requirements Quality Assurance	29
3.4. Traceability of Safety Requirements	33
3.5. Safety Argumentation	39
3.6. Model-Based Safety Assessment	41
3.7. Evaluation Summary of the State of the Art	47
Part II: Hazard Relation Diagrams	51
4. Overview	53
4.1. Modeling Concepts of Hazard Relation Diagrams	53
4.2. Creation Approach for Hazard Relation Diagrams	54
4.3. Tool Support for Hazard Relation Diagrams	54
5. Modeling Concepts of Hazard Relation Diagrams	57
5.1. Ontological Foundations	57
5.2. Integration with UML Activity Diagrams	58
5.3. Visual Notation for the Modeling Concepts	59
5.4. Relationship Types between Hazards and Conceptual Mitigations	64
5.5. Well-Formedness Rules	69
6. Creation Approach for Hazard Relation Diagrams	75
6.1. Overview	75
6.2. Canonical Artifact Type Formalization	77
6.3. Artifact Creation	92
6.4. Summary of the Satisfaction of Well-Formedness Rules	110

7.	Tool Support for Hazard Relation Diagrams.....	113
7.1.	Purpose and Rationales of Technology Choices	113
7.2.	Hazard Relation Diagram Profile.....	114
7.3.	Tool Prototype to Create Hazard Relation Diagrams.....	117
8.	Discussion of the Solution Approach.....	123
Part III: Empirical Evaluation.....		125
9.	Overview	127
9.1.	Experimental Design.....	127
9.2.	Experimental Results	127
9.3.	Threats to Validity	127
9.4.	Discussion of the Results	127
10.	Experimental Design.....	129
10.1.	Research Questions	129
10.2.	Evaluation Strategy.....	130
10.3.	Experimental Stimuli	131
10.4.	Hypotheses and General Metrics	131
10.5.	Experiment 1: Individual Reviews (Between-Subjects)	135
10.6.	Experiment 2: Fagan Inspections (Between-Subjects).....	139
10.7.	Experiment 3: Individual Reviews (Within-Subjects)	144
10.8.	Experiment 4: Fagan Inspections (Within-Subjects)	148
10.9.	Data Preparation	153
10.10.	Participant Experience Levels across Experiments.....	155
11.	Experimental Results.....	159
11.1.	Hypothesis H1: Rationale Objectivity	160
11.2.	Hypothesis H2: Effectiveness	165
11.3.	Hypothesis H3: Efficiency	169
11.4.	Hypothesis H4: Self-Reported Confidence.....	173
12.	Discussion of the Results of the Empirical Evaluation	179
12.1.	Hypothesis H1: Rationales Objectivity	179
12.2.	Hypothesis H2: Effectiveness	180
12.3.	Hypothesis H3: Efficiency	180
12.4.	Hypothesis H4: Self-Reported Confidence.....	181
13.	Threats to Validity	183
Part IV: Conclusion and Outlook		187
14.	Contribution, Limitations, and Future Work	189
14.1.	Contribution of this Dissertation.....	189
14.2.	Discussion and Limitations.....	190
14.3.	Future Work.....	191
References		193

List of Abbreviations

AADL	Architecture Analysis and Design Language
ACC	Adaptive Cruise Control
ESP	Electronic Stability Program
FHA	Functional Hazard Analysis
FMEA	Failure Mode and Effect Analysis
FTA	Fault Tree Analysis
GSN	Goal Structuring Notation
H+R	Hazard and Risk Assessment Analysis
MOF	Meta Object Facility
QVT	Query/View/Transformation
QVTo	QVT Operational Language
STPA	Systems-Theoretic Process Analysis
SUD	System under Development
SysML	Systems Modeling Language
UML	Unified Modeling Language

List of Figures

Figure 1-1	Current Situation –Lack of Explicit Dependencies between Independent Artifact Types.....	6
Figure 1-2	Solution Approach – Definition of a Diagrammatic Representation to Visualize Dependencies...	10
Figure 2-1	Excerpt of the Functional Requirements of the Example ACC.	13
Figure 2-2	Relationship between Defined Concepts.	19
Figure 4-1	Example for the Definition of Ontological Relationships between Artifact Types.	53
Figure 5-1	Ontological Foundation of Hazard Relation Diagrams.....	59
Figure 5-2	Example of a Hazard Relation Diagram featuring Hazard H1 from Table 2-1.	63
Figure 5-3	Example of a Hazard Relation Diagram featuring Multiple Mitigation Partitions.	69
Figure 6-1	Exemplary Overview over the Creation of Hazard Relation Diagrams.....	76
Figure 6-2	Excerpt of the Functional Requirements of the Example ACC from Figure 2-1.....	79
Figure 6-3	Trigger Condition of Hazard H1 from Table 6-1 as a Binary Tree.....	83
Figure 6-4	Hazard Relation Diagram from Figure 5-2 overlaid with the formal elements from (6.48).	90
Figure 6-5	Simplified Example of Transformation Steps to Create Hazard-Mitigating Requirements.....	93
Figure 6-6	Result of Applying the Partial Mitigation from Table 6-3 by Means of Listing 2.....	97
Figure 6-7	Result of Applying the Partial Mitigation from Table 6-3 by Means of Listing 2 and Listing 3....	98
Figure 6-8	Preliminary Hazard Relation Diagram without Appended Hazard Associations.	109
Figure 7-1	Conceptual Metamodel of the Hazard Relation Diagram Profile.	115
Figure 7-2	Structure and Technical Interplay of the Tool Prototype Components.....	118
Figure 7-3	Functional Interplay of “GenerateMitigationImpl ”from Figure 7-2.	120
Figure 7-4	Functional Interplay of “GenerateHRDIImpl” from Figure 7-2.	120
Figure 10-1	Experimental Procedure for Experiment 1.	138
Figure 10-2	Experimental Procedure for Experiment 2.	142
Figure 10-3	Experimental Procedure for Experiment 3.	147
Figure 10-4	Experimental Procedure for Experiment 4.	151
Figure 11-1	Distributions over all Ordinates of the Retained Quality Foci for H4.3.	176
Figure 11-2	Distributions over all Ordinates of the Retained Quality Foci for H4.4.	177

List of Tables

Table 2-1	Functional Hazard Analysis of the Function “Compute Brake Force” from Figure 2-1.....	14
Table 3-1	Evaluative Criteria for the Related Approaches in the State of the Art.	23
Table 3-2	Classification Scheme of Model-Based Safety Approaches according to [Lisagor et al. 2011].	41
Table 3-3	Evaluation Summary of Related Works.	48
Table 5-1	UML Activity Diagram Elements Used to Depict Functional Requirements in Hazard Relation Diagrams.....	60
Table 5-2	Modeling Concepts of Hazard Relation Diagrams and their Visual Notation.	61
Table 5-3	Types of Relationships between Hazards and Conceptual Mitigations.	68
Table 6-1	Excerpt of the Functional Hazard Analysis from Table 2-1 featuring Hazard H1.....	82
Table 6-2	Mitigation Template to Specify Changes to Activity Diagrams Containing Functional Requirements in order to Create Hazard-Mitigating Requirements.....	84
Table 6-3	Example of a Specified Mitigation Template to Mitigate Hazard H1 from Table 2-1 according to Mitigation 1 from Section 2.1.....	87
Table 6-4	Summary of Well-Formedness Rules and their Satisfaction.	111
Table 10-1	Summary of the Experimental Configuration in the Four Experiments.	130
Table 10-2	TAM3 and TTF Item Selection using a Template based on [van Solingen and Berhout 1999]. ..	134
Table 10-3	General Metrics for each Hypothesis.....	135
Table 10-4	Specific Metrics for Hypothesis H2.3 in Experiment 2.	140
Table 10-5	Specific Metrics for Hypotheses H3.2 in Experiment 3.	145
Table 10-6	Specific Metrics for Hypotheses H3.4 in Experiment 3.	146
Table 10-7	Specific Metrics for Hypotheses H4.2 in Experiment 4.	149
Table 10-8	Specific Metrics for Hypotheses H4.3 in Experiment 4.	149
Table 10-9	Specific Metrics for Hypotheses H4.4 in Experiment 4.	150
Table 10-10	Cronbach’s α for all quality foci from Table 10-2 and all experiments.....	155
Table 10-11	Relative Levels of Experience in Participants for all Experiments.....	156
Table 10-12	Means, Std. Dev., T-Test, and Power for Participants’ Levels of Experience in Exp. 1 & 3.	157
Table 10-13	Means, Std. Dev., T-Test, and Power for Participants’ Levels of Experience in Exp. 2 & 4.	158
Table 11-1	Means, Std. Deviations, T-Test, and Power for H1.1.	160
Table 11-2	Rationales Categorization for each Inspected Conceptual Mitigation for H1.2.....	161

Table 11-3	Means, Std. Deviations, T-Test, and Power for H1.3	162
Table 11-4	Rationales Categorization for each Inspected Conceptual Mitigation for H1.4.....	164
Table 11-5	Means, Std. Deviations, T-Test, and Power for H2.1.	165
Table 11-6	Judgement Effectiveness for H2.2.	166
Table 11-7	Means, Std. Deviations, T-Test, and Power for H2.3.	167
Table 11-8	Judgement Effectiveness for H2.4.	168
Table 11-9	Means, Std. Deviations, T-Test, and Power for H3.1.	169
Table 11-10.	Judgement Efficiency for H3.2.	170
Table 11-11	Means, Std. Deviations, T-Test, and Power for H3.3.	171
Table 11-12	Judgement Efficiency for H3.4.	172
Table 11-13	Means, Std. Deviations, T-Test, and Power for H4.1.	174
Table 11-14	Means, Std. Deviations, T-Test, and Power for H4.2.	175
Table 11-15	Means and Std. Deviations H4.3	176
Table 11-16	Means and Std. Deviations H4.4.	177
Table 12-1	Summary of Experimental Results, Hypotheses and Research Questions.....	179

Part I: Introduction

1. Motivation and Goal

This chapter serves as the introduction of this dissertation. Section 1.1 introduces the development process of safety-critical systems in early phases by illustrating the interplay between early safety assessment activities and requirements engineering. In Section 1.2, challenges regarding the validation of the adequacy of requirements intended to mitigate hazards are discussed. Assumptions for a solution approach are discussed in Section 1.3. The solution idea is presented in Section 1.4 followed by an overview of the solution presented in this thesis in Section 1.5. The structure of this thesis is outlined in Section 1.6.

1.1. The Early Development Stages of Safety-Critical Systems

The development process of safety-critical systems shall make sure that the system to be developed is sufficiently safe. “*Safe*” in this sense means that no harm can come to human users or external systems (see [Ericson 2005], p. 478). In early safety assessment¹, the system’s functional requirements are subjected to hazard analyses (e.g., Functional Hazard Analysis, FHA, see [Ericson 2005], p. 271ff). According to [Leveson 2011a], hazards are operational situations that – given disadvantageous trigger conditions from the operational context – can cause or contribute to harm (cf. [Leveson 2011a], p. 467). During hazard analyses, hazards are identified based on the functional requirements that induce them. The functional requirements are defined during requirements engineering. They specify the functionality needed to fulfill the system’s operational purpose [IREB e.V. 2016]. Some of the functional requirements may be the reason of hazards to occur during operation. In the following, we will use the term “*hazard-inducing requirements*” for such functional requirements (cf. [Firesmith 2004], see Section 2.1 for the definitions of the terminology used in this work). Based on the identified hazards, safety goals are derived and mitigations are conceived to fulfill the safety goals. In requirements engineering [IREB e.V. 2016], the term “*goal*” refers to a top-level desired property or objective of the system under development. For safety-critical embedded systems, this means that the safety goals derived during hazard analyses must be refined into mitigations that fulfill the safety goals [Leveson 2011b]. Mitigations thus comprise specific functionalities, which avoid, reduce, or control the occurrence of the hazard’s trigger conditions, once these functionalities have been implemented into the system entirely and without error ([Leveson 1995], p. 401). This can be

¹ In this dissertation, we use the term “safety assessment” to refer to the activities carried out during development of a system, while we use the term “safety engineering” for the academic discipline.

achieved by pursuing one of the following strategies (see [Leveson 1995], pp. 154, 177, and 401, for a detailed discussion):

- **Hazard Prevention.** A hazard is mitigated by preventing the hazard’s trigger conditions from occurring during operation.
- **Hazard Reduction.** A hazard is mitigated by reducing the likelihood of the hazard to occur, e.g., by reducing the likelihood of the trigger conditions to occur.
- **Accident Prevention.** If a hazard cannot be prevented or sufficiently reduced, a mitigation can be concerned with preventing an accident due to the hazard.
- **Damage Control.** If a hazard can neither be prevented, nor sufficiently reduced, nor can damage can be prevented, a mitigation can aim to protect human users from injury or external systems from damage or reduce the severity of injury or damage. This could be achieved, for example, by means of additional functionality intended specifically for damage reduction.

In the simplest case, the hazard-inducing requirement can simply be omitted in order to mitigate a hazard. Yet, typically, the requirement documents functionality that is necessary for the system’s operational purpose. Therefore, regardless of the chosen mitigation strategy, in order to incorporate the mitigation into the system, it is first necessary to amend the functional requirements specification of the system under development by means of hazard-mitigating requirements [Berry 1998]. Hazard-mitigating requirements are a specific type of functional requirements (cf. [Firesmith 2004]) which, assuming correct and complete consideration of the hazard’s trigger conditions, fulfill the safety goals [Wilson et al. 1997; Bishop et al. 2004] in order to make harm sufficiently unlikely or sufficiently tolerable (see [Berry 1998; Leveson 1995; US NASA 2011]).

One of the challenges of the development of safety-critical systems according to [Hatcliff et al. 2014] is to validate that the hazard-mitigating requirements are *adequate to mitigate* the hazards (see [Bohm 1984; Leveson 1995 (p. 177); Hatcliff et al. 2014]). This means that the right hazard-mitigating requirements must be defined to adequately prevent or reduce the hazard, prevent an accident, or control the occurring damage. We use the term “*adequate*” in the sense outlined in [Glinz 2000] in order to distinguish between provable correctness of requirements, which is ascertained through formal methods, and the subjective suitability of requirements to document stakeholder needs (see [Glinz and Fricker 2015]), which must be validated manually. Validating adequacy of hazard-mitigating requirements means that stakeholders must check whether the hazard-mitigating requirements are suitable to mitigate a hazard. This

is typically done using techniques, such as reviews [Flynn and Warhurst 1994; Wiegers 2002] or inspection-like techniques [Aurum et al. 2002].

1.2. Challenges in Hazard-Mitigating Requirements Adequacy Validation

The challenge in validating the adequacy of hazard-mitigating requirements is due to the dependencies between hazard-mitigating requirements the hazards intended to be mitigated. Typically, these dependencies are not trivial one-to-one relationships, where one hazard-mitigating requirement mitigates exactly one hazard. The following multiplicities can exist (see [US DOD 2010], p. 101):

1. **One-to-One: One hazard-mitigating requirement exists for one hazard.** This is the simplest case, in which the adequacy of exactly one hazard-mitigating requirement depends on its ability to prevent or reduce exactly one hazard, prevent an accident due to the hazard, or control the damage due to a hazards.
2. **One-to-Many: One hazard-mitigating requirements exists for multiple hazards.** In this case, the adequacy of exactly one hazard-mitigating requirement must be judged with regard to all hazards it is intended to mitigate. This means that the hazard-mitigating requirement may adequately mitigate one hazard, but potentially not all other hazards.
3. **Many-to-One: A number of hazard-mitigating requirements exist to mitigate one hazard.** In this case, a set of hazard-mitigating requirements has been specified. All of those hazard-mitigating requirements must adequately mitigate exactly one hazard.
4. **Many-to-Many: A number of hazard-mitigating requirements exist to mitigate multiple hazards.** This case is a combination of case 2 and case 3. The adequacy of a set of hazard-mitigating requirements depends on their ability to mitigate every single hazard they are intended to mitigate. A set of hazard-mitigating requirements may adequately mitigate one hazard, but not another hazard. Furthermore, there might be overlap within the set of hazard-mitigating requirements such that an individual hazard-mitigating requirement contributes to mitigating one hazard, but impairs the adequate mitigation of another hazard.

These different types of multiplicities make validating the adequacy of hazard-mitigating requirements difficult. Among others, the stakeholders have to identify the type of multiplicity before validation can commence. In addition, hazard-mitigating requirements and hazard analyses results exists in two independent types of documents: hazards, trigger conditions, and safety goals are typically documented by means of hazard analysis worksheets (e.g., [Ericson 2005], p. 276). They are work products of safety assessment. The hazard-inducing requirements

as well as the hazard-mitigating requirements are part of the functional requirements specification. The functional requirements specification is a work product of the requirements engineering process. Typically, dependencies between safety assessment artifacts and requirements engineering artifacts are not explicitly documented ([Cleland-Huang 2005; Cleland-Huang et al. 2012], cf. [Hatcliff et al. 2014]). Figure 1-1 illustrates this situation.

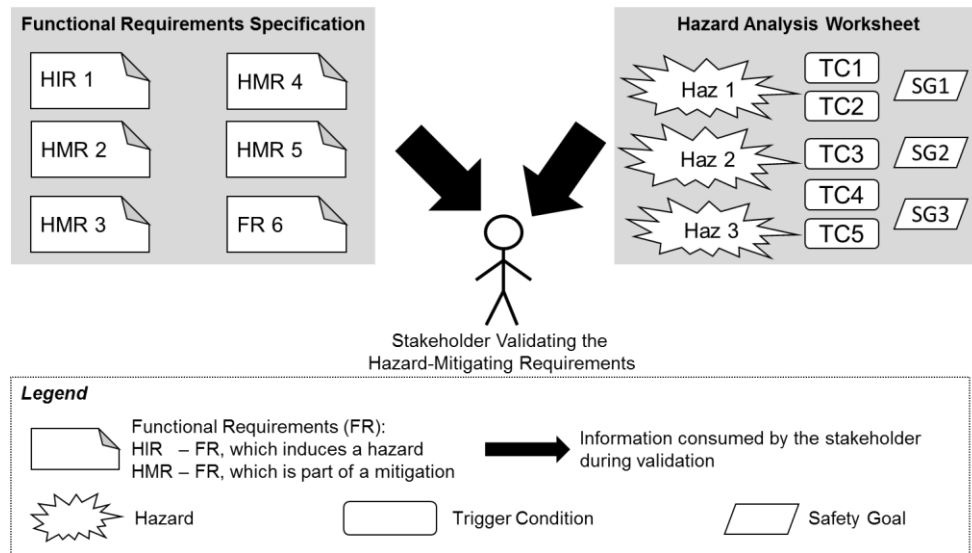


Figure 1-1 Current Situation –Lack of Explicit Dependencies between Independent Artifact Types.

In Figure 1-1, the functional requirements specification contains all of the system’s requirements and the hazard analysis worksheets contain all hazards, the hazards’ trigger conditions, and safety goals. The stakeholders must consider the information within the requirements specification as well as the information within the hazard analysis worksheet (black arrows in Figure 1-1) and manually differentiate hazard-mitigating requirements from other functional requirements. Due to the lack of explicitly documented dependencies between the hazards and their mitigation, the stakeholders must manually ascertain which hazard-mitigating requirements are intended to mitigate which hazards. Only if stakeholders know which hazard-mitigating requirements pertain to which hazard, the adequacy of the mitigation can be validated. This leads to the following challenges for the stakeholders during validation:

Challenge 1: Stakeholders have no indication how hazards have been mitigated.

As mentioned above, hazard analysis worksheets contain “recommended actions” in the form of hazard negations. If there are no explicitly documented dependencies between the hazard-mitigating requirements and the hazard, stakeholders have difficulty ascertaining which hazard-mitigating requirements were defined to mitigate which hazard, and whether these hazard-mitigating requirements adequately fulfill the safety goal (see Section 4.1 in [Heimdahl 2007]).

Challenge 2: Stakeholders have no indication how trigger conditions are influenced by the hazard-mitigating requirements.

Hazard analysis worksheets typically list causal factors of a hazard (see [Ericson 2005], p. 276). However, hazard analysis worksheets typically do not define how the occurrence of the trigger conditions during operation can be avoided in order to adequately mitigate the hazard [Leveson 1991; Hatcliff et al. 2014]. Whether or not alternative conditions to trigger a hazard exist, or how these trigger conditions influence one another is also not documented (cf. [Lutz 2000]). Hazard-mitigating requirements might adequately mitigate the hazard for one trigger condition. Yet, if they also prevent the occurrence of a trigger condition for another hazard remains covert.

Challenge 3: Stakeholders have no indication which hazard a specific hazard-mitigating requirement is related to.

The requirements specification typically does not differentiate between hazard-mitigating requirements and other functional requirements with no impact on safety. A lack of differentiation increases the effort for stakeholders to identify the relevant hazard-mitigating requirements intended to mitigate a specific hazard. Moreover, stakeholders cannot easily distinguish hazard-mitigating requirements pertaining to the same hazard mitigation from hazard-mitigating requirements intended to mitigate other hazards. In consequence, their ability to validate the hazard-mitigating requirements' adequacy to mitigate the hazard is impaired.

1.3. Assumptions

In order to develop an approach to address the challenges outlined above, we make the following assumptions.

Assumption 1: Focus on software-intensive safety-critical embedded systems.

The approach will be specific to and evaluated with regard to its application to software-intensive safety-critical embedded systems. For example, a discussion of the differences of hazards for mechanical and software-intensive systems is given in [Leveson 1986] and the differences between hazards for embedded and cyber-physical systems are discussed in [Tenbergen et al. 2014].

Assumption 2: Focus on “early” hazards identified based on functional requirements.

The approach focuses on hazards identified during early phases of development. Hazards identified during late development phases of safety assessment, e.g., through Failure Mode and Effect Analysis (FMEA, see [Ericson 2005], p. 235) are typically dealt with on implementation-level. Such hazards are defined as constraints and formal quality assurance methods are applied to check those constraints.

Assumption 3: Completeness of functional requirements specification and hazard analysis results.

A functional requirements specification is complete and available such that hazard analyses can be conducted (see Section 1.2). Hazard analyses must be conducted based on the functional requirements in order to determine which of the functional requirements induce hazards, to identify the hazards that are induced, to identify the hazard’s trigger conditions, and to elicit possible safety goals. Based on these hazard analysis results (see Assumption 5), mitigations can be conceived (see Assumption 6). For this work, it is assumed that these activities have been carried out completely and yielded complete results.

Assumption 4: Depiction of functional requirements in UML activity diagrams.

The same information that is input for hazard analyses are available in a graphical representation, i.e. UML activities and other activity diagram modeling elements (see [OMG 2015b]). For these modeling elements, an underlying ontology exists called the meta-object facility (MOF, see [OMG 2015a] for details). These modeling elements can be hence be ontologically integrated with hazard analysis results (see Assumption 5). Specifying functional requirements using diagrammatical representations in this manner is common industry practice [Cziharz et al. 2016].

Assumption 5: Documentation of hazard analyses in worksheets.

Hazard analyses have been conducted based on the functional requirements (see Assumption 3) and the results are documented and available to stakeholders. The hazard analysis worksheets suggested in [Ericson 2005] (p. 276) are a table-based format, which can be ontologically related to the modeling elements representing the functional requirements of the system (see Assumption 4).

Assumption 6: Completeness of mitigations.

The engineering effort necessary to conceive at least one mitigation for each hazard has been undertaken by the stakeholders. This means that the approach is not concerned with finding a mitigation, but instead with documenting the dependencies between the mitigation and the hazard analysis worksheets.

1.4. Goal and Solution Idea

Based on the assumptions from Section 1.3, the goal of this dissertation can be formulated as follows:

GOAL OF THIS DISSERTATION

»Support stakeholders in the validation of hazard-mitigating requirements through an integrated diagrammatic representation which visualizes the dependencies between hazard-mitigating requirements, hazards, the hazard's trigger conditions, and the hazard's safety goal.«

This dissertation proposes an integrated diagrammatic representation of hazard-mitigating requirements, hazards, the hazards' trigger conditions, and the safety goal in order to explicitly visualize the dependencies between them. The integrated diagrammatic representation is called “*Hazard Relation Diagram.*” Hazard Relation Diagrams contain the information from hazard analysis worksheets as well as the hazard-mitigating requirements defined to mitigate a specific hazard. The stakeholders make use of Hazard Relation Diagrams during validation in order to ascertain the adequacy of the hazard-mitigating requirements to mitigate hazard. Hazard Relation Diagrams address the challenges outlined in Section 1.2 in the following way:

- By making explicit the dependencies between hazard-mitigating requirements and hazard analysis results, Hazard Relation Diagrams allow visualizing how hazards have been mitigated, i.e. which specific hazard-mitigating requirements pertain to which hazard. This addresses Challenge 1.
- Hazard Relation Diagrams visualize mitigations and hazards together with the hazard's trigger conditions, in one diagram, per hazard mitigation. This allows stakeholders to focus validation on whether or not the hazard-mitigating requirements sufficiently mitigate the hazard. This addresses Challenge 2.
- Hazard Relation Diagrams preserve the relationship between the functional requirements that were part of a diagram before the mitigation was included, the hazard-inducing re-

quirements which gave rise to some hazard, and the hazard-mitigating requirements intended to mitigate the same hazard. This allows Hazard Relation Diagrams to visually differentiate the modeling elements that constitute the hazard-mitigating requirements from those modeling elements in the diagram that remained unchanged. This eases validation of the adequacy of the mitigation and addresses Challenge 3.

One Hazard Relation Diagram is created for each hazard mitigation. A Hazard Relation Diagram contains all hazard-mitigating requirements that are part of the mitigation for the same hazard. During validation, each Hazard Relation Diagram is reviewed individually, thereby allowing for each hazard mitigation to be validated.

1.5. Overview of the Approach

The specific solution approach is discussed in Part II. Figure 1-2 shows an overview over the solution approach.

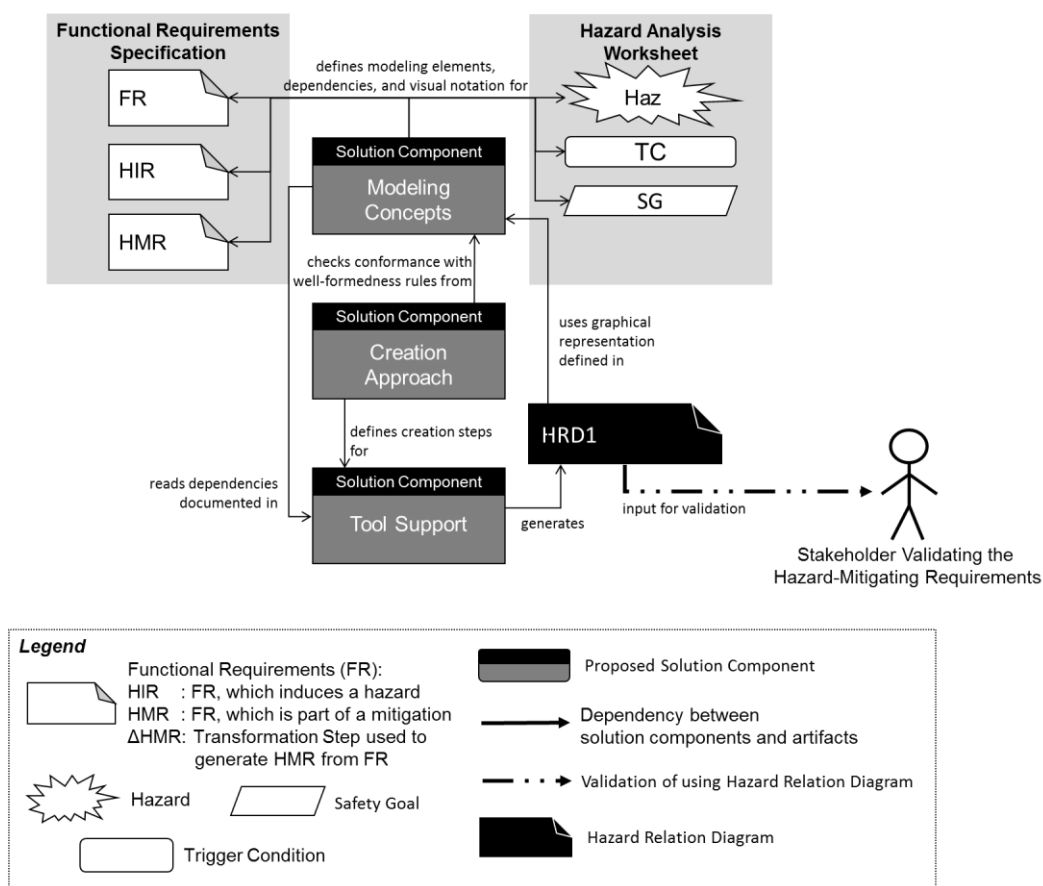


Figure 1-2 Solution Approach – Definition of a Diagrammatic Representation to Visualize Dependencies.

As can be see, the solution approach consists of the following components and the following interplay of the components:

- 1. Modeling Concepts of Hazard Relation Diagrams.** The modeling concepts of Hazard Relation Diagrams is the central solution component, which introduces the modeling elements to *visualize the dependencies* between functional requirements, hazard-inducing requirements, hazard-mitigating requirements, hazards, trigger conditions, and safety goals. To do so, an *ontology* is used to derive syntactic *well-formedness rules*, which are based on the semantic relationship defined in the ontology between the modeling concepts depicted in the Hazard Relation Diagram. A *visual notation* for each concept type and relationship type of the ontology defines the visual language of Hazard Relation Diagrams. This language is used to represent concrete hazard-mitigating requirements, hazards, trigger conditions, and relationships in a Hazard Relation Diagram. This solution component is described in Chapter 5.
- 2. Creation Approach for Hazard Relation Diagrams.** The creation approach is used to *create Hazard Relation Diagrams*. It uses the transformation steps as well as the dependencies documented in *mitigation templates*. Mitigation templates document the dependencies between the mitigated hazard from the hazard analysis worksheet, its trigger conditions and safety goal, as well as to the hazard-inducing requirements that induced the hazard. The mitigation takes the form of *transformation steps* to allow for the creation of Hazard Relation Diagrams from activity diagrams containing functional requirements. The approach executes the transformation steps and observes the well-formedness rules defined on the basis of the ontology and makes use of the visual notation to depict modeling elements. This solution component is described in Chapter 5.5.
- 3. Tool Support for Hazard Relation Diagrams.** The tool makes use of the mitigation templates from the creation approach to specify the mitigation for a certain hazard. Using a *technical UML profile* that adheres to the ontology, well-formedness rules and visual notation, the tool allows creating Hazard Relation Diagrams *manually or automatically*. This solution component is described in Chapter 7.

In addition, we conduct an **empirical evaluation** of Hazard Relation Diagrams. The purpose of the evaluation is to investigate the impact of Hazard Relation Diagrams. The results of the empirical evaluation show that using Hazard Relation Diagrams during validation improves stakeholders' ability to ascertain how hazards have been mitigated (Challenge 1). The evaluation furthermore provides evidence that Hazard Relation Diagrams improves stakeholders' ability to ascertain how hazard-mitigating requirements avoid, control, or reduce the occurrence of the hazard's trigger conditions (Challenge 2). Our results indicate that Hazard Relation Diagrams successfully allow stakeholders to differentiate hazard-mitigating requirements from other

functional requirements with no impact on safety (Challenge 3). The empirical evaluation is described in Part III.

1.6. Outline

This dissertation is structured as follows:

- In **Part I “Introduction,”** the specific challenges and goal of this dissertation are illustrated. Furthermore, this part discusses the fundamentals necessary for the remainder of this dissertation consisting of the terminology adopted in the dissertation, and approaches this dissertation builds upon. Finally, this part discusses and evaluates the state of the art of relevant related state of the art.
- In **Part II “Hazard Relation Diagrams,”** the solution to the challenges developed in this dissertation is outlined. Ontological foundations and formal underpinnings of the solution components are illustrated and the approach developed in this dissertation is presented. Furthermore, tool support for the approach is discussed consisting of a UML profile as well as a tool-prototype.
- In **Part III “Empirical Evaluation,”** the experiments conducted to evaluate the integrated diagrammatic representation are presented. The motivation and design of the four experiments is described, including both, between-subjects and within-subjects designs. The results of the experiments are discussed in detail in this section as well.
- In **Part IV “Conclusion and Outlook,”** we summarize and critically discuss the results of this dissertation. Limitations of Hazard Relation Diagrams and their underlying approach are discussed. Furthermore, an outlook on future work to continue this research trajectory is given.

2. Fundamentals

In this chapter, we introduce a running example and define the adopted terminology.

2.1. Running Example – The Adaptive Cruise Control System

In this section, the running example is introduced. This running example is used to illustrate the solution approach in Part II as well as the empirical evaluation in Part III.

The example system under development (SUD) is a simplified adaptive cruise control system (ACC) from the automotive domain. An ACC is a driver assistance system of modern automobiles that maintains both a desired speed set by the driver and a minimal safe distance to a vehicle driving ahead. An excerpt of the functional requirements from such a simplified ACC are shown in the UML activity diagram in Figure 2-1.

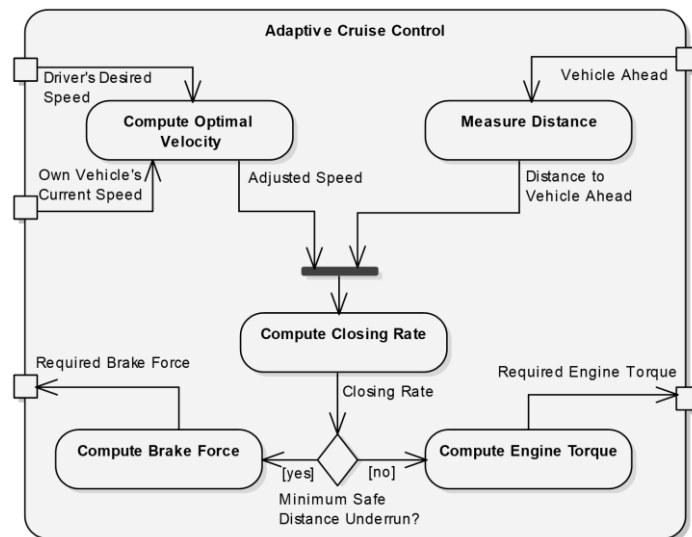


Figure 2-1 Excerpt of the Functional Requirements of the Example ACC.

As can be seen, the ACC in this example continuously monitors the *Driver's Desired Speed* and the *Own Vehicle's Current Speed* and computes the *Adjusted Speed* through the function “Compute Optimal Velocity”. Doing so allows the ACC to gradually reach the desired speed through smooth acceleration and deceleration, rather than using maximum engine power or brake force. Concurrently, the function “Measure Distance” continuously monitors some *Vehicle Ahead* that has been detected, e.g., by means of RADAR or LIDAR sensors, and computes the *Distance to Vehicle Ahead*. Based on the adjusted speed and the distance to the vehicle ahead, the function “Compute Closing Rate” determines whether or not the own vehicle is closing in on the vehicle driving ahead. The term “*Closing Rate*” hence refers to the rate of approximation of the own vehicle to the vehicle driving ahead. If it is determined that the own vehicle

underruns the minimal safe distance, the ACC computes the *Required Brake Force* to maintain the minimal distance. Else, the *Required Engine Torque* is computed to achieve the *Adjusted Speed*. More details on ACCs can be found in [Reif 2010; Mao and Chen 2012; Caramihai and Dumitrache 2013].

Given that ACCs are driver assistance system from the automotive domain, their development is governed by safety standards, particularly [ISO26262 2011]. This means that during early phases of safety assessment, these functional requirements are subjected to systematic hazard analyses. It is to note, that depending on the industry, different types of hazard analyses may be common. For example, while the avionics industry mandatorily conducts Functional Hazard Analysis (FHA, see [ARP4761 1996]) for the purpose of certification, the automotive industry typically conducts Hazard and Risk Assessment Analysis (H+R, see [ISO26262 2011]). As outlined in [Ericson 2005], early-stage hazard analyses overlap with respect to the identification of potential hazards, their harmful effects, possible trigger conditions under which the hazards are induced, and possible safety goals in response to the hazard (see Section 2.1 for definitions of these terms). In this example, the results of hazard analyses have been documented using the hazard analysis worksheet suggested in [Ericson 2005] (p. 276). The hazard analysis worksheet has been truncated by removing risk index fields, as these are beyond the scope of this research. In Table 2-1, an excerpt of a hazard analysis for the functional requirement “Compute Brake Force” from Figure 2-1 is shown.

Table 2-1 Functional Hazard Analysis of the Function “Compute Brake Force” from Figure 2-1.

System	Adaptive Cruise Control		Functional Hazard Analysis			
Hazard-Inducing Requirement	Hazard		Effect	Trigger Condition	Safety Goal	
	ID	Description			ID	Description
Compute Brake Force	H1	Yaw Momentum Causes Oversteering	Loss of Control, causing Crash	Driving through Curve and (High Vehicle Velocity or Low Road Surface Friction)	SG1	Prevent Loss of Control in Curves
	H2	Understeering due to Skidding		Driving through curve or Low Road Surface Friction	SG2	Use Anti-Lock Brakes to Prevent Wheel Skid
	H3	Brake Force Too Low due to Skidding	Rear-End Collision with Vehicle Driving Ahead	Low Road Surface Friction	SG3	Adjust Brake Force Independently for All Four Wheels

As can be seen from Table 2-1, “Compute Brake Force” induced three hazards. It is hence not only a functional requirement, but also a hazard-inducing requirement (see Section 1.1). One hazard that could occur during operation is that a rapid deceleration command is issued by the ACC whilst the car is driving through a curve, which may cause the driver to lose control over the vehicle and crash (hazard H1 in Table 2-1). There are two possible mitigations for hazard H1 to fulfill the safety goal “Prevent Loss of Control in Curves” (SG1 in Table 2-1):

Mitigation 1: Query Yaw Rate and Delimit Maximum Brake Force.

Safety goal SG1 in Table 2-1 could be satisfied by querying a yaw rate sensor in order to determine whether or not the vehicle is currently driving through a curve, and hence limit the maximum brake force. Naïvely, this may be an adequate way to mitigate the hazard. However, doing so might cause the vehicle to not brake sufficiently, thereby underrunning the minimal safe distance, or possibly even rear-ending the vehicle driving ahead.

Mitigation 2: Query Yaw Momentum and Brake Individual Wheels.

More experienced automotive engineers might instead choose to query the current yaw momentum (i.e. the lateral force exerted onto the car during yaw) from the electronic stability program (ESP) in addition to querying the yaw rate sensor. In addition, this mitigation computes the necessary brake force for each wheel and submit the result to the ESP, thereby maximizing brake efficiency and maintaining a controlled yaw, while at the same time, keeping a safe distance to the vehicle driving ahead.

2.2. Terminology

Anecdotal evidence [Heimdahl 2007] suggests that different interpretations between practitioners and scholars in the disciplines of safety engineering and software engineering may be the culprit. To lay a common foundation for the following sections, this section defines the basic concepts, terms, and relationships that are relevant for this dissertation.

The term "hazard" is used differently across industrial safety standards (e.g., [ARP4761 1996; ISO26262 2011]) and scholars in the field of safety engineering. For example, while both [ARP4761 1996] as well as [ISO26262 2011] use the term *hazard* to denote some unsafe effect of the system onto its operational context, [ARP4761 1996] adopts a somewhat broader definition. In [ISO26262 2011], a hazard merely arises from random hardware failures (i.e. *endogenous* hazards according to [Leveson 1995]), whereas in [ARP4761 1996], a hazard may also be triggered by external events such as faulty user input (i.e. *exogenous* hazards according to [Leveson 1995]). Consequently, the conditions triggering a hazard are considered *failure conditions* in [ISO26262 2011] and *hazardous events* in [ARP4761 1996]. Moreover, hazards can be differentiated by means of other characteristics (see [Tenbergen et al. 2014] for a detailed overview and a taxonomy). For example, hazards can arise due to defects or failures in design or implementation or due to the basic functionality of the system (see [Berry 1998]). While the former are design-specific hazards, the latter are design-agnostic (cf. [Pumfrey 1999]). Such

design-agnostic hazards are identified based functional requirements during early stages of development as described in Section 1.1. This is done by guiding hazard analyses through the use of key words indicating erroneous behavior (e.g., “fails to operate,” “operates inadvertently,” “produces wrong output,” see [Ericson 2005], p. 372f). In Assumption 2 (see Section 1.3), this approach was limited to the latter kind, i.e. design-agnostic “early” hazards. We therefore adopt the following generic definition of the term “hazard,” based on [Leveson 2011a] (p. 467) and [Stoneburner 2006]:

Definition 1: Hazard.

A hazard is an operational situation that – given disadvantageous triggering conditions in the operational context of the system – could lead or contribute to harm to come to humans or systems.

In this sense, a hazard does by itself not cause harm during operation, but creates the potential for harm, if no mitigation is in place. As outlined in Section 1.1, hazards are triggered by disadvantageous trigger conditions. We define the term trigger condition as follows in accordance with [Leveson 2011a] (p. 184):

Definition 2: Trigger Condition.

A trigger condition is an operational or environmental condition, which may occur during operation such that a hazard is caused.

This means that a trigger condition, possibly together with other trigger conditions can cause one or more hazards during operation and hence render the system potentially unsafe. Trigger conditions must hence be avoided or rendered sufficiently unlikely to occur during operation in order for the hazard to be mitigated. Safety goals are conceived in response to the hazard and its trigger conditions. For the term “safety goal,” we adopt the following definition based on [Leveson 2011b]:

Definition 3: Safety Goal.

A safety goal is a statement about the system’s safety or about a specific safety property that the system possesses or shall possess.

Many authors point out that safety goal are often simple objectives, which must be fulfilled by the system’s specification (see also, e.g., [Wilson et al. 1997; Bishop et al. 2004; Kelly and Weaver 2004; US FAA 2009a; US FAA 2009b]). In the most trivial cases, safety goals simply

state the negation of an identified hazard (see, e.g., [Johnson and Holloway 2006; Kelly 2007; US FAA 2009b; ISO26262 2011]), but just like in goal-oriented requirements engineering, these goals have to be refined into concrete functional safety requirements to mitigate a hazard.

The term “functional safety requirement” entails two distinct, but related meanings. On the one hand, safety engineering literature understands requirements to be safety-critical when they give rise to a hazard (see, e.g., [Knight 2002; Johnson and Holloway 2006]). On the other hand, especially requirements engineering literature often considers safety requirements a type of quality requirement (e.g., [Kotonya and Sommerville 1998]), which is in place to achieve a certain level of safety. However, safety can only be achieved when concrete functional safety requirements in the sense of [Firesmith 2004] are in place, i.e. concrete conditions and capabilities that, when implemented entirely and without error, mitigate the hazard. To honor this dual role of requirements (see also [Berry 1998]) with regard to safety and to emphasize the functional nature of requirements documenting hazard mitigations, we hence adopt the following definitions inspired by [Firesmith 2004]:

Definition 4: Hazard-Inducing Requirement.

A hazard-inducing requirement is a functional safety requirement in the sense of [Firesmith 2004], which is origin of a hazard during operation, given the occurrence of trigger conditions from the operational context of the system.

Definition 5: Hazard-Mitigating Requirement

A hazard-mitigating requirement is a functional safety requirement in the sense of [Firesmith 2004], which mitigates a hazard (possibly together with other hazard-mitigating requirements).

Hazard-mitigating requirements may themselves cause hazards. Therefore, safety assessment must entail iterative hazard identification and hazard mitigation (see, e.g., [ARP4761 1996; Pumfrey 1999; ISO26262 2011]), i.e. hazard-mitigating requirements themselves must be subjected to hazard analyses and it might be necessary to conceive hazard-mitigating requirements for hazards that arise due to a mitigation of another hazard.

Definition 5 suggests that minimally, the mitigation of a hazard consists of one hazard-mitigating requirement. In practice, this is rarely the case and typically, multiple hazard-mitigating requirements must be defined in order to adequately mitigate a hazard. In this sense, completeness of hazard-mitigating requirements may impair adequacy: if one or more hazard-

mitigating requirements are missing, the mitigation is likely to be inadequate. Assessing the adequacy of hazard-mitigating requirements is part of the validation process.

In the safety engineering literature, the term “*mitigation*” is used rather abstractly. For the technical aspects of Hazard Relation Diagrams (see Sections 5.4 and 5.5), a differentiation of this term is necessary. This is because in model-based specifications, diagrams are structured according to, e.g., organizational needs [Conway 1967], use cases, or the appropriate level of detail needed for the specific development situation [Finkelstein et al. 1992]. Requirements diagrams are hence rarely organized according the hazards identified during hazard analyses. In consequence, it is necessary to differentiate the hazard-mitigating requirements that pertain to some mitigation from other model elements. We therefore define the following two terms:

Definition 6: Partial Mitigation.

A partial mitigation consists of a set of hazard-mitigating requirements that are intended to mitigate a specific hazard.

Definition 7: Conceptual Mitigation

A conceptual mitigation consists of at least one partial mitigation, which refines a hazard’s safety goal into concrete implementable measures to avoid the hazard or reduce its harmful effects.

The term “*conceptual mitigation*” hence refers to the chosen strategy of how to mitigate a hazard, while the term “*partial mitigation*” refers to the concrete hazard-mitigating requirements defined to implement the strategy. The concept “*conceptual mitigation*” therefore serves as a bridge between the abstract mitigation strategy (see Section 1.1 as well as [Leveson 1995]) and its technical realization (see Section 6.3). If two or more conceptual mitigations for the same hazard exist, these can be thought of as alternative strategies to mitigate the same hazard. In the following Section 5.4, the distinction between conceptual mitigations and partial mitigations is discussed in more detail.

The relationship between these the terms and concepts is visualized in Figure 2-2 by means of the running example from Section 2.1. As can be seen, the functional requirements of the ACC from Figure 2-1 were subjected to hazard analyses. One hazard that could be *triggered* when driving through a curve at high velocity is that the yaw momentum might exceed a critical level, thereby causing oversteering such that the driver to loses control. This might potentially result in a crash (see hazard H1 in Table 2-1). In this case, “Compute Brake Force” from Figure

2-1 is a *hazard-inducing requirement* for the hazard “Yaw Momentum Causes Oversteering.” A possible *conceptual mitigation* for this would be to add requirements to achieve the *safety goal* “Prevent Loss of Control in Curves” (SG1 in Table 2-1). Following the strategy indicated in the conceptual mitigation, one *partial mitigation* was added, containing *hazard-mitigating requirements* to determine the current yaw rate (“The ACC shall query the yaw sensor to determine the current yaw rate”) and limiting the brake force (“If the yaw rate exceeds 5°/sec, the maximum brake force shall be reduced by 15% per 5°/s of yaw”).

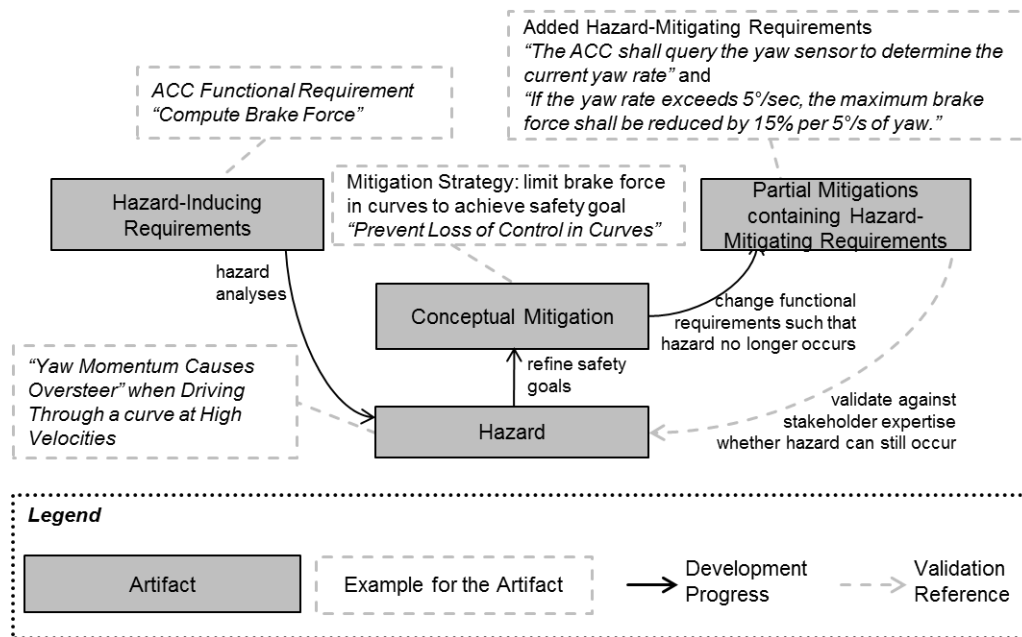


Figure 2-2 Relationship between Defined Concepts.

3. State of the Art

In the following subsections, the relevant state of the art is reviewed and evaluated. Subject of this evaluation are approaches, which contribute to addressing the challenges discussed in Section 1.2. For this purpose, an evaluation framework is derived and discussed in Section 3.1. The following Sections 3.2 through 3.6 present and evaluate the related approaches before Section 3.7 summarizes the findings of the evaluation.

3.1. Evaluation Framework for the State of the Art

The evaluation framework for the state of the art is composed of two components: a set of *categories* of related approaches and a set of *evaluation criteria* for the related approaches in the respective categories.

3.1.1. Categories of Related Work

Existing work in the state of the art is considered *related*, if the work potentially contributes to a solution of the challenges discussed in Section 1.2. We differentiate between the following categories:

- 1. Elicitation of safety requirements.** This category comprises constructive approaches intended to obtain safety-related requirements after hazard analyses have been conducted. Such approaches typically aim at providing “constructively correct” requirements that make up the conceptual mitigation² for a hazard (see Assumption 4 in Section 1.3). If constructively correct conceptual mitigations consisting of hazard-mitigating requirements in the sense of Definition 5 can be obtained using these approaches and their dependency to a hazard (in the sense of Definition 1) and the hazard’s trigger conditions (in the sense of Definition 2), is explicit, the challenges from Section 1.2 can be addressed. This state of the art is described in Section 3.2.
- 2. Requirements quality assurance.** This category comprises analytical approaches intended to assess the quality of safety-related requirements. Such approaches are potentially capable of identifying inadequacy in hazard-mitigating requirements (in the sense of Definition 5) with regard to their ability to mitigate a hazard (in the sense of Definition

² Since in the related work, the distinction between the hazard’s conceptual mitigation and the partial mitigation subsuming the hazard-mitigating requirements is not common, we will use the terminology proposed by the respective authors throughout Chapter 3 and relate these terms to the terminology outlined in Section 2.2 as well as the challenges from Section 1.2.

- 1). Such approaches therefore potentially contribute to addressing the challenges from Section 1.2 and are reviewed in Section 3.3.
- 3. Traceability of requirements.** This category comprises approaches concerned with establishing trace dependencies safety-related requirements or other requirements to other development artifacts. Such approaches can possibly be adapted to establishing dependencies between hazard-mitigating requirements (in the sense of Definition 5) and hazard analysis worksheets for the purpose of assisting validation. These approaches hence potentially contribute to addressing the challenges outlined in Section 1.2 and are reviewed in Section 3.4.
- 4. Safety argumentation.** This category comprises approaches focused on demonstrating that the system under development is certifiably safe. Safety arguments are typically concerned with establishing inferences between evidence for the system’s safety and some top-level safety claim. Such inferences can be understood as dependencies between a safety goal (in the sense of Definition 3) and a conceptual mitigation (in the sense of Definition 7). Hence, safety argumentation may harbor potential solutions for the challenges outlined in Section 1.2. These approaches are reviewed in Section 3.5.
- 5. Model-based safety assessment.** In contrast to approaches pertaining to traceability of safety requirements (see Section 3.4), approaches in this category do not focus on establishing dependencies, but visualizing them. Since the goal of this dissertation is concerned with visualization of the dependencies between hazard-mitigating requirements and hazard analysis worksheets (see Section 1.4), this category harbors potentially applicable approaches. These approaches are reviewed in Section 3.6.

3.1.2. Evaluation Criteria for Related Approaches

To evaluate related work we define a set of evaluation criteria based on the challenges outlined in Section 1.2. The approaches are evaluated with regard to their contribution to solve a challenge, their ability to solve a challenge in part, or their unsuitability to solve a challenge. The following Table 3-1 summarizes the challenges from Section 1.2, lists the three contribution levels *solved completely* (●), *partially solved* (⊙), and *unsolved* (○), and summarizes the necessary criteria for a related approach to meet a certain level.

Table 3-1 Evaluative Criteria for the Related Approaches in the State of the Art.

Challenge	Solved Completely ●	Partially Solved ⊙	Unsolved ○
Challenge 1: Indication, how hazards were mitigated.	The approach entails the creation of explicit dependencies from each identified hazard to the specific set of hazard-mitigating requirements intended to mitigate it.	The approach entails the systematic consideration of hazards or hazard-mitigating requirements, but does not establish explicit dependencies between the two.	The approach is not concerned with hazards in the sense of Definition 1 nor with the dependencies between hazards and hazard-mitigating requirements.
Challenge 2: Indication, how trigger conditions influence a hazard’s occurrence.	The approach entails the creation of explicit dependencies from each hazards’ trigger conditions to hazard-mitigating requirements.	The approach entails the systematic consideration of trigger conditions, but does not establish explicit dependencies between hazards’ trigger conditions and hazard-mitigating requirements.	The approach is not concerned with the dependencies between hazards’ trigger conditions in the sense of Definition 2 and hazard-mitigating requirements.
Challenge 3: Differentiate hazard-mitigating requirements from remainder of the functional requirements specification.	The approach differentiates hazard-mitigating requirements in the sense of Definition 5 pertaining to one hazards from hazard-mitigating requirements pertaining to other hazards or from other functional requirements with no impact on safety.	The approach is concerned with, e.g., “safety requirements” but does not distinguish hazard-mitigating requirements in the sense of Definition 5 from other requirements with no impact on safety.	The approach is not concerned with hazard-mitigating requirements in the sense of Definition 5, nor with safety requirements, nor with their differentiation.

3.2. Safety Requirements Elicitation

Requirements elicitation approaches are considered relevant in the context of this dissertation (see Section 1.4) as long as they are informed by hazard analyses in order to elicit hazard-mitigating requirements with regard to the identified hazards. These approaches can be distinguished into goal- or scenario-based approaches and formal safety requirements elicitation approaches.

3.2.1. Goal- and Scenario-Based Elicitation of Hazard-Mitigating Requirements

Goal-based requirements elicitation approaches are typically concerned with deriving concrete, implementable functional requirements from higher-level abstract goal descriptions.

The KAOS Approach

The KAOS approach was initially proposed in [Darimont and van Lamsweerde 1996]. A detailed elaboration can be found in [van Lamsweerde 2009a]. In the KAOS approach, goals and dependencies between goals are graphically depicted. The central concern of the KAOS approach to refine each goal [Letier and van Lamsweerde 2002]. These refinements are much akin to functional requirements and follow formal patterns. In [van Lamsweerde and Letier 1998], it is suggested to apply the concept of goal refinements onto what the authors call “obstructions.” Obstructions can be considered anti-goals, i.e. goals which must be avoided. Among other things, obstructions can arise due to impaired safety concerns, i.e. hazards in the sense of Definition 1. Refinements of such “hazard obstructions” [van Lamsweerde and Letier 1998] are called “resolutions” and are similar to hazard-mitigating requirements according to Definition

5. Following the refinement graph from a hazard obstruction to its resolution in this sense gives an indication how a hazard was mitigated, hence solving Challenge 1, but the resolutions of the hazard obstructions in the remainder are treated the same way as refinements of any other goal. Hazard-mitigating requirements can hence be only distinguished from other requirements with no impact on safety if an appropriate trace concept is in place which allows to associate hazard-mitigating requirements with the hazard obstructions. In consequence, the KAOS approach partially addresses Challenge 3. Furthermore, KAOS supports the formal specification of the system's behavioral goals using temporal logic [Letier and van Lamsweerde 2004; van Lamsweerde 2009b], however, is unconcerned with the influence of the resolutions of hazard obstruction on the occurrence of a hazard obstruction. In KAOS, Challenge 2 is therefore only partially addressed.

The i* Approach and Derivatives

Another prominent approach is i* [Yu 1997]. The i* approach is a software engineering methodology originally intended to refine requirements during early stages of development into concrete architectural components called “agents.” In i*, dependencies between business goals and goal-fulfilling system components or external actors can be visualized in strategic dependency models and strategic rationale models and can then be formally analyzed and refined. The i* approach is extended by the Tropos approach (e.g. [Bresciani et al. 2001]) with the specific aim to associate business goals with risks and security impairments. Furthermore, the Tropos “goal-risk-framework” [Asnar and Giorgini 2006] is concerned with risks at the organizational level and is not concerned with risks to safety (i.e., hazards) while the Secure Tropos extension [Mouratidis et al. 2003] introduces concepts to model trust and resources for the purpose of modeling possible security vulnerabilities and their resolution. Albeit safety engineering is concerned with mitigating hazards to parties external to the system, while security engineering is concerned with protecting the system from external threats, the mitigation of hazards and security vulnerabilities often involves similar strategies (see [Raspotnig and Opdahl 2013] for a comprehensive overview on the application of security engineering techniques on system safety). Therefore, through formal refinement of security goals, Secure Tropos partially addresses Challenge 1 from Section 1.2. However, since Secure Tropos is not concerned with trigger conditions in the sense of Definition 2 nor with differentiating hazard-mitigating requirements in the sense of Definition 5 from functional requirements with no impact on safety, Challenges 2 and 3 remain unaddressed.

The Approach by Menon and Kelly

Very similar to goal-oriented approaches is the approach presented by Menon and Kelly [Menon and Kelly 2010]. The authors propose separating safety arguments into modules such that each module addresses logical, contractual, or organizational concerns with particular regard of the interplay between system modules and software modules. Doing so allows separating the problem space into orthogonal aspects, which aids in eliciting a set of top-level safety goals pertaining to each aspect. This allows differentiating hazard-mitigating requirements pertaining to specific hazards to some degree (Challenge 1), yet does not consider the impact of the safety goal refinements with regard to the hazards' trigger conditions (Challenge 2). Conceivably, the idea of separating the system into safety-related "modules" supports differentiating those modules with impact on safety from those without. Yet, this is not explicitly suggested in [Menon and Kelly 2010], hence only partially addressing Challenge 3.

The Approach by Kotonya and Sommerville

Separation of the problem space is also done in the viewpoint-oriented approach in [Kotonya and Sommerville 1997]. In this approach, the authors aim at identifying safety-relevant aspects within the system's operational environment in order to guide hazard and risk assessment and subsequent systematic elicitation of safety constraints. Safety constraints are derived from abstract safety concerns and subsequently subjected to formal verification (albeit the authors use the term "validation"). While the combined safety analyses/requirements engineering process model is well illustrated in [Kotonya and Sommerville 1997], safety validation in this approach is mainly concerned with formally checking the requirements specification against constraint violations (see Section 3.3.2). Conceivably, such constraint violations could be considered trigger conditions for the hazards, hence addressing Challenge 2. However, the approach is not concerned with hazards in the sense of Definition 1 nor with hazard-mitigating requirements in the sense of Definition 5. Therefore, Challenge 1 and 3 are not addressed.

The Approach by Allenby and Kelly

Other approaches aim at eliciting safety requirements from scenarios and/or use cases. For example, in [Allenby and Kelly 2001], the authors propose an [ARP4761 1996]-compliant model-based approach to identify possible deviations from intended system behavior by making use of user scenarios and use cases. Possible hazards arising from the deviations can be identified and possible mitigations can be suggested by developing safety requirements from these use

cases. This allows specifying hazard-mitigating requirements specifically for hazards and addresses Challenge 1. Conceivably, deviations can be caused by the hazards' trigger conditions. However, details about the exact nature of the origin of such deviations are not addressed in [Allenby and Kelly 2001]. The approach is therefore categorized as partly fulfilling Challenge 2. Moreover, the approach is unconcerned with differentiating resulting requirements from the remainder of functional requirements. Doing so requires extensive additions of the approach, e.g., through the use of a qualified trace concept (see, e.g., Section 3.4). The approach by itself hence does not address Challenge 3.

The Approach by Dittel and Aryus

In [Dittel and Aryus 2010], an approach for the construction of [ISO26262 2011]-compliant safety cases is presented. The approach takes vehicle functions and use cases (which can be regarded as a logical consolidation of related scenarios) and performs a series of analysis steps in order to elicit safety requirements and construct an ISO26262 safety concept. The approach aids in creating a V&V plan, which allows assessing whether the safety goal has been met. Since some safety goals can be seen as negation of hazards [Bishop et al. 2004], the approach hence partly solves Challenge 1. However, the approach does not take into account the hazard's trigger conditions in the sense of Definition 2 and makes no mention of the creation of specific hazard-mitigating requirements in the sense of Definition 5. The approach hence does not consider Challenges 2 nor 3.

Misuse Cases

Another approach intended to derive hazard-mitigating requirements are misuse cases [Sindre and Opdahl 2001]. Misuse cases allow specifying the system's behavior by means of a collection of user-perceivable interactions with the system, i.e. use cases. The idea of such an interaction is extended to interactions that are hostile, e.g., originating from a malicious user, or to interactions that allow using the system in unforeseen ways. Such hostile or unintended interactions are considered misuse cases, for which appropriate system responses and system constraints must be found to maintain security. In [Sindre 2007], misuse cases are applied in a case study to safety-critical systems with the aim to identify hazards and find hazard resolutions. Albeit the results of the case study show that in principle, misuse cases are applicable and have advantages for safety assessment, particularly for hazard identification and tracing of hazard-mitigating requirements, misuse cases should be seen as a supplemental technique. Challenges 1 and 3 are hence addressed in part because tailoring of misuse cases is necessary in order to

fully address these challenges. Since misuse cases are unconcerned with the conditions from the operational context that trigger hazards to occur during operation in the sense of Definition 2, Challenge 2 as outlined above is not addressed.

3.2.2. Formal Elicitation of Hazard-Mitigating Requirements

There is a plethora of approaches concerned with eliciting hazard-mitigating requirements through formal analyses. These approaches have in common that the term “requirements” in these approaches typically refers to safety-critical invariants or constraints and often consider information provided by safety analyses.

The Approach by Chen and Motet

An example of a formal elicitation approach is the approach by Chen and Motet [Chen and Motet 2009a; Chen and Motet 2009b; Chen and Motet 2009c]. In this approach, safety requirements are considered constraining control structures for the behavior specified by the functional requirements. For this purpose functional requirements are formalized using interface automata. Formal safety constraints are elicited using control automata for each hazardous control structure. While the formal dependencies between control structures and safety constraints allow for a differentiation of safety constraints from non-safety-critical constraints (Challenge 3), neither hazards in the sense of Definition 1 nor trigger conditions in the sense of Definition 2 are considered (Challenges 1 and 2).

The Approach by Zafar and Dromey

In [Zafar and Dromey 2005], a formal approach is suggested, which elicits behavior constraints by making use of behavior trees. Behavior trees allow analyzing the formal behavior specification with regard to conflicts or defects in behavioral constraints. Under the assumption that conflict or defect might result in a hazard during operation, the behavior constraints can be revised in order to address this hazard. The approach hence partially addresses Challenge 1. However, there is no specific consideration of the hazards trigger conditions in the sense of Definition 2 and elicited safety constraints are not differentiated from non-safety-relevant constraints (Challenge 3).

Systems-Theoretic Process Analysis

Leveson’s Systems-Theoretic Process Analysis (STPA) approach [Leveson 2004] is an integrated approach that considers causes for hazards and accidents based on the operational purpose and emergent properties of the system. The STPA approach provides process guidance for

the identification of hazards as well as for the functions mitigating the hazard. For each hazard controlling functions, invariants can be identified and documented, thereby addressing Challenge 1. Furthermore, STPA fosters adequacy validation of such mitigations by investigating how such a hazard can occur (i.e. what are the trigger conditions in the sense of Definition 2), and how a hazard controlling function will impact the respective hazard. STPA therefore also addresses Challenge 2. Yet, since the key focus of STPA are hazards and their mitigation, the approach makes no mention how to proceed with the identified hazard-mitigating requirements. Hence, whether or not hazard-mitigating requirements are explicitly differentiated from other functional requirements without impact on safety in the sense of Challenge 3 is not discussed. Challenge 3 therefore is not explicitly considered in the STPA approach.

The Approach by Hansen et al.

Hansen et al. propose a formal elicitation approach based on Fault Tree Analyses (FTA) in [Hansen et al. 1998]. Fault Tree Analysis attempts to find possible sources of failures and document them in a tree structure using conjunctions and disjunctions. The approach by Hansen et al. translates these fault trees into timing constraints, which must be honored by the subsequently implemented software components. Any potential timing deviation is considered hazardous and must be mitigated. In this sense, the approach considers special types of hazards, specifically those that occur due to violated real-time constraints. In consequence, the approach neither considers hazards in the sense of Definition 1 nor trigger conditions in the sense of Definition 2 and hence neither addresses Challenge 1 nor 2 in the sense outlined above. Since safety-critical timing constraints are derived for specific system components, these timing constraints are explicitly distinguishable from non-safety related timing constraints. Yet, these are not hazard-mitigating requirements in the sense of Definition 5. Therefore, Challenge 3 is addressed in part by this approach.

The Approach by Troubitsyna

An approach that combines FMEA with FTA in order to elicit hazard-mitigating requirements is presented in [Troubitsyna 2008]. In this approach, FTA is conducted to find critical software components and possible failures therein. Then, statecharts are used to facilitate control structures for the identified hazardous components. Afterwards, FMEA is applied in order to find appropriate tolerance requirements with regard to the formal system behavior documented in the statecharts. Albeit the FTA-guided identification and subsequent systematic derivation of

hazard-mitigating requirements allow for finding candidate mitigations for each hazard (Challenge 1), the approach does not consider the conditions triggering a hazard (Challenge 2) in the sense outlined above. Furthermore, the statecharts used in this approach depict control structures for all hazards and hence do not allow differentiating between mitigations nor between hazard-mitigating requirements and functional requirements (Challenge 3).

The Approach by Belli et al.

In [Belli et al. 2007], an event-based approach is suggested which combines qualitative risk analyses and FTA in order to mitigate hazards that emanate from the system. This is done by first modeling user inputs and system outputs and analyzing the event chain from input to output. Unsafe interactions can thereby be mitigated. Hence, this approach is concerned with finding candidate mitigations for unsafe interactions with the user that are built into the system rather than eliciting safety requirements with regard to system hazards. Yet, these candidate mitigations have the character of “counter-examples,” where some assumed to be safety-critical condition is achieved in the system and hence resolves some hazardous interaction. Hence, the approach addresses Challenge 3. However, the approach is unconcerned with hazards in the sense of Definition 1 nor with trigger conditions in the sense Definition 2, and therefore does not address Challenge 1 or 2.

3.3. Requirements Quality Assurance

Approaches pertaining to requirements quality are considered relevant to the fulfillment of the goal of this dissertation (see Section 1.4) as long as they make use of information provided by hazard analyses against which the adequacy of hazard-mitigating requirements is evaluated. There are two types of such approaches: Tailored approaches to improve generic requirements validation techniques, and formal requirements quality approaches.

3.3.1. Approaches to Support Requirements Validation

In a state of practice report from 1994 [Flynn and Warhurst 1994], Flynn and Warhurst have indicated that unstructured reviews are the predominant technique to conduct validation. The authors pointed out that effectiveness and review coverage is considerably impaired by the size of the specification as well as missing traces between requirements. Furthermore, lack of available reference information and inherent subjectivity is seen as detrimental to review objectivity by practitioners. Although more recent investigations into the state of practice are desirable, it

becomes obvious that these issues prevail to this day: For example, more recent empirical findings were reported by Shull et al. [Shull et al. 2002] and Boehm and Basili [Boehm and Basili 2001], who also indicate that peer reviews are about 60% effective in identifying defects in engineering artifacts. A number of improvements to reviews [Wieggers 2002] and review-like techniques such as Fagan inspections [Fagan 1976; Fagan 1986] have been proposed to alleviate these issues (see [Porter et al. 1995] for an in-depth comparison). These techniques are applicable to the challenges outlined in Section 1.2, however require safety-specific tailoring as outlined in the following.

Perspective-based Reading

Perspective-based reading is a technique to improve upon requirements reviews and inspections [Shull et al. 2000]. The key component in perspective-based reading is to read the requirements specification from the *perspective* of different roles involved in the development process. For example, the hazard-mitigating requirements could be read from the perspective of the safety engineer, who has conducted the hazard analyses. Applying perspective-based reading has shown to improve effectiveness during reviews [Basili et al. 1996]. This techniques could therefore address Challenge 1 and Challenge 2, if the stakeholders reading the hazard-mitigating requirements were trained in to read the specification in the capacity of the safety engineer. However, stakeholders would still need to manually identify the hazard-mitigating requirements in the first place. Hence, perspective-based reading does not address Challenge 3.

Value-based Reading

Value-based reading is a technique intended to assist in the prioritization of artifacts [Li et al. 2011]. By applying value-based reading, the requirements in a specification can be assigned priorities to allow for more critical requirements to be validated first and hence increase validation effectiveness [Lee and Boehm 2005]. This is achieved by reading the requirements specification and assigning *values* through a conceived metric, for example in order to find the requirements with the highest impact on business risks, security, or potentially safety. Value-based reading is therefore suitable to identify hazard-mitigating requirements within the requirements specification, if an appropriate safety-centric metric was conceived. Yet, such an adaptation requires considerable extension of the value-based reading technique such that Challenge 3 can at best be evaluated as partly addressed. Moreover, value-based reading requires

conjunctive application with other techniques (e.g., the aforementioned perspective-based reading technique), in order to be applicable to Challenge 1 and Challenge 2, as by itself, value-based reading does not consider hazards nor their trigger conditions explicitly.

Checklist-based Reading

Checklist-based reading and derivative approaches such as defect-based reading are techniques aimed towards increasing the detection rate of defects in software artifacts [Porter et al. 1995]. The key component of such approaches is a guidance document (i.e., a checklist), which assists the stakeholder conducting the validation in finding common types of defects. Such guidance documents can be tailored to the operational purpose of the system or the specific defects. This could include defects like missing mitigations for specific hazards or inadequate consideration of the hazard’s trigger conditions. In consequence, checklist-based reading-like techniques are potentially applicable to address Challenge 1 and 2, but requires manual identification of hazard-mitigating requirements, much akin to perspective-based reading. Therefore, Challenge 1 and 2 are considered partly addressed. However, similar to perspective-based reading, stakeholders still need to manually identify the hazard-mitigating requirements in the first place such that checklist-based reading does not address Challenge 3.

SafeSpection

SafeSpection is a framework aimed at improving project-specific hazard identification in software-intensive systems [Denger et al. 2008]. The framework makes use of what is called “meta-questions,” which can be refined into “project-specific questions” in order to identify hazards and, conceivably, the hazard-mitigating requirements intended to mitigate the hazard. During the SafeSpection process, hazard analysis is guided by common guidewords. The impact of hazard mitigations on the trigger conditions can be assessed through what the SafeSpection framework calls “influence factors” [Denger 2009]. These can be tailored towards the system under development, its functions, and the specific hazards the system’s functions induce. In this sense, SafeSpection addresses Challenge 1 and Challenge 2. However, hazard-mitigating requirements must still be manually identified during the SafeSpection process, similarly to perspective-based and checklist-based reading techniques. Therefore, SafeSpection does not address Challenge 3.

3.3.2. Formal Quality Assurance Approaches

While the formal approaches discussed in Section 3.2.2 aim at analyzing system artifacts in order to elicit safety requirements, other formal approaches are concerned with identifying safety-critical defects within mitigations. Examples of these approaches are discussed in the following.

The Approach by Heitmeyer et al.

An example for approaches that are concerned with behavioral constraints is the approach by Heitmeyer et al. suggested in [Heitmeyer et al. 1998; Bharadwaj and Heitmeyer 1999]. The approach of Heitmeyer et al. interprets the requirements specification as a state machine and focuses on invariants, states, and transitions similar to the approach by Troubitsyna [Troubitsyna 2008]. The approach is capable of generating candidate solutions that may circumvent safety defects, but the focus of the approach is verification rather than elicitation or validation. Candidate solutions to resolve specific safety-critical defects can be found using this approach (hence addressing Challenge 1), given the violated safety-relevant invariants (hence addressing Challenge 2). However, the approach only generates candidate solutions that entail the entire system behavior. Hence, albeit the approach generates hazard-mitigating requirements, no distinction is made to remaining functional requirements. Therefore the approach by Heitmeyer et al. only partially addresses Challenge 3.

The Approach by Snelting et al.

The approach by Snelting et al. [Snelting et al. 2006] is an example of approaches concerned with data safety. Specifically, the approach is concerned with safety-critical invariants called “path conditions” between input and output variables of the system. The principle idea is to assert path conditions which must hold for any execution of the software and to use constraint solvers to check if the path conditions can be satisfied in all instances. If circumstances are found in which a path condition is violated, it is asserted that this constitutes a potentially unsafe execution. In this sense, the approach is concerned with the conditions, under which the system exerts hazardous behavior, hence addressing Challenge 2. However, the approach is neither concerned with dependencies to requirements mitigating these hazards (Challenge 1) nor with differentiating such requirements from the remainder of the specification (Challenge 3).

The Approach by Cheung and Kramer

In [Cheung and Kramer 1999], an approach is presented that allows reducing the state space of a behavior specification such that the reduced behavior model still reflects the same behavior as the initial model. To this extent, behavior which is not externally observable but may be safety-relevant is analyzed in the approach (e.g., internal control structures). The aim is to identify states and transitions that do not affect externally observable behavior and can be removed without impairing system safety. By making use of this approach while refining the behavior specification and/or the system design, error correction efforts can be reduced. The approach is hence an example of a class of approaches which assert that a system is safe, if there are no violated behavioral constraints. These behavioral constraints can be viewed as trigger conditions for hazards (Challenge 2). Yet, like the approach in [Snelting et al. 2006], the approach by Cheung and Kramer is not concerned with identifying specific mitigations for hazards in the sense of Definition 1 (Challenge 1). Similarly, the approach does not assist in differentiating hazard-mitigating requirements in the sense of Definition 5 from other functional requirements without impact on safety (Challenge 3).

The Approach by Saeed et al.

In [Saeed et al. 1993], an approach is suggested which assesses the robustness of the requirements specification against specific risks. The requirements specification is generated by conducting a series of qualitative analysis pertaining to different viewpoints of system safety with varying degrees of abstraction. In this sense, the approach is similar to the approach in [Kotonya and Sommerville 1997], however with emphasis on verifying suitability of the requirements specification against potential accidents, hazards, and risks. Subsequent quantitative analyses allow generating fault trees that can be used to find behavioral violations in the specification with regard to the identified hazards. In this sense, the approach addresses Challenge 1 insofar that it allows checking if a mitigation for a hazard is in place. However, neither the hazard's trigger conditions are considered in the sense of Challenge 2 nor are the hazard-mitigating requirements differentiated from the remainder of the specification (Challenge 3).

3.4. Traceability of Safety Requirements

Requirements traceability describes the ability of stakeholders to trace requirements from their origin to their implementation. Two types of requirements traceability can be distinguished [Gotel and Finkelstein 1994]: *pre-traceability* describes the ability to trace requirements to their origin, i.e. to the stakeholder, who elicited the requirement or to the analysis result that made

the requirement necessary. *Post-traceability* describes the ability to trace requirements to their realization in design documents or in the implementation. In the context of this work, pre-traceability of requirements is of particular relevance, as the hazard from the hazard analysis worksheet in the sense of Definition 5 becomes the origin of the hazard-mitigating requirements intended to mitigate the hazard. In the literature, a host of approaches, techniques, and tools have been suggested to establish and manage requirements traceability (see, e.g., [Bashir and Qadir 2006; Rochimah et al. 2007; Torkar et al. 2012] for overviews). In the following, we will review types of approaches that are particularly aimed at pre-traceability of safety requirements or pre-traceability of quality requirements³ that are applicable to safety requirements.

The Trace Queries Centric Approach by Cleland-Huang et al.

In [Cleland-Huang et al. 2012], an approach is proposed that allows tracing requirements for safety-critical systems by means of queries modeled in between artifacts. The aim of the approach is to support stakeholders in showing the mitigation of hazards for the purpose of certification. The approach consists of four steps. First, fault trees are built in order to determine “preliminary hazards” of the system. Each of these preliminary hazards is explored by a subtree to identify the specific failures which cause the hazard. Afterwards, cut sets are defined to identify the failures, whose simultaneous occurrence will result in the occurrence of the root fault of the system. This yields an understanding of the failure causes of the system and allows specifying safety-related requirements. These safety-related requirements can be related to the root fault in the fault tree to be mitigated. To do so, a graphical language is used to visualize the traces between hazards and their mitigations. Using this visualization, the safety-related requirements must be subjected to validation. The approach is based on a conceptual information model that establishes dependencies between, among other things, the preliminary hazards and the artifact containing the hazards (in this case, the fault tree), formal state-based models, software requirements, and system requirements. System requirements in this approach refer to hazard-mitigating requirements in the sense of Definition 5, which addresses Challenge 1. The state-based models allow capturing the formal system behavior, i.e. system states and transitions. They also capture “certain assumptions about the environment” ([Cleland-Huang et al. 2012], p. 182). Albeit the approach is concerned with “failures” which result in hazardous behaviors, the operational conditions that cause these failures in the sense of Definition 2 are

³ We use the term “quality requirement” to refer to the notion of “non-functional requirements” used by some authors to underline the fact that some quality properties, e.g., safety, are impacted by functional requirements, as outlined in [Glinz 2007].

not in the focus of the approach. Moreover, it must be noted that albeit the approach establishes and visualizes dependencies between hazards and their mitigations, the established dependencies trace individual hazard-mitigating requirements to hazards. This partly addresses Challenge 3. However, in order to fully address Challenge 3, the approach requires extension such that hazard-mitigating requirements pertaining to different hazards can be differentiated from one another.

The Softgoal-Centric Traceability Approach by Cleland-Huang et al.

In another approach [Cleland-Huang et al. 2005], Cleland-Huang and colleagues propose the use of goal analysis to trace the satisfaction of quality requirements. The approach consists of four steps. First, a goal model is created, which documents the qualities to be considered as softgoals and dependencies between softgoals. In a second step, dependencies between softgoals are analyzed using a probabilistic algorithm in order to delineate the impacts between goals and a functional model of the system. Furthermore, in this step, human users assess the dependency links manually prunes incorrectly delineated links. Afterwards, in a goal analysis step, human users can modify the functional elements to which the softgoals are related in order to optimize their satisfaction. This may include the addition, removal, or substitution of functional elements and is similar to the conception of hazard-mitigating requirements in order to fulfill some top-level safety goal (cf. Section 1.2). After the modification is done, goal satisfaction is recursively re-evaluated. In the last step, stakeholders make decisions on whether to accept or reject the modifications. Albeit the approach is intended for “non-functional” properties, specifically security, performance, and usability, the approach has been successfully applied to a case example in order to trace other qualities, including safety. This means that in principle, the approach can be applied to generating dependencies between hazards and hazard-mitigating requirements in the sense of Challenge 1. Since the approach entails the analysis of specific changes conducted by human users, the approach can conceivably also differentiate between hazard-mitigating requirements and the remainder of the functional requirements specification in the sense of Challenge 3. However, the approach is unconcerned with the specific trigger conditions leading to a hazard and hence does not Challenge 2.

The Event-Based Traceability Framework by Cleland-Huang et al.

Another contribution by Cleland-Huang and colleagues (see [Cleland-Huang et al. 2002a; Cleland-Huang et al. 2002b; Cleland-Huang et al. 2003]) makes use of the publisher-subscriber

model to inform *related* artifacts in a requirements specification about changes in *updated* artifacts. The contribution presents a framework, which consists of three parts [Cleland-Huang et al. 2002a]: a requirements manager maintains a repository of all requirements artifacts that can be updated or related to other requirements artifacts. The second component is an event server, which manages update events of requirements artifacts. Update events entail “change actions” [Cleland-Huang et al. 2002b], such as adding, removing, or replacing artifacts within the requirements manager. The third component is the subscriber manager, which listens to update events and notifies all related requirements artifacts about the change. The framework assumes that traceability information between requirements artifacts are already in place. The framework has been applied to model-based requirements [Cleland-Huang et al. 2003], but conceivably, the implementation of the publisher-subscriber model proposed in the framework can be applied to other artifacts as well, such as hazards or trigger conditions. In the sense of Challenge 1 and 2, hazards and/or trigger conditions would be “notified” (in the sense outlined in [Cleland-Huang et al. 2002a]) about requirements that have been changed in order to mitigate the hazard. However, the nature of the framework does not allow distinguishing between hazard-mitigating requirements and the remainder of the requirements artifacts in the sense of Challenge 3.

The Safety Artifact Traceability Model by Katta and Stålhane

In [Katta and Stålhane 2011], the authors propose a conceptual traceability model between the system development process and the safety assessment process. The conceptual model is closely related to the interaction between the two development processes first introduced in [Pumfrey 1999] and relates artifacts in both processes to one another. Specifically, the traceability model relates conceptual system functions in the system conceptualization phase to specific hazards identified during early stage hazard analyses. Mitigations are traced to hazards in the hazard analysis stage as well. Functional safety requirements in the conceptual traceability model take the role of hazard-mitigating requirements in the sense of Definition 5 and constitute a specialization of a mitigation. Functional safety requirements also trace to system safety requirements in the “requirements analysis” phase of system development in the conceptual model. These system safety requirements are special types of functional requirements of the system in the traceability model. In summary, the traceability model presented in [Katta and Stålhane 2011] takes a process-centric approach. It allows tracing hazards to their mitigation in the sense of Challenge 1 and differentiates those requirements that are part of a mitigation from those that are not in the sense of Challenge 3. However, the traceability must still be manually

established. Furthermore, albeit the traceability model considers the “effects” that lead to a hazard like the trigger conditions in the sense of Definition 2, these effects are not traced to any type of requirement.

The Approach by Hill and Tilley

The approach proposed by Hill and Tilley in [Hill and Tilley 2010] is aimed at establishing traceability between artifacts of legacy systems for the purpose of recertification. The approach is based on a risk taxonomy and database, which contains information about risks for the system to be certified. In this sense, the term “risk” means operational conditions which threaten the operational mission of the system. The purpose of the risk taxonomy and database is to allow product managers to identify risks and conceive strategies to avoid them. These strategies are documented as safety requirements and inform a risk evaluation process. However, the approach makes no mention of distinguishing safety requirements of the system to be certified from other requirements without impact on safety. In this sense, the approach is concerned with managing the development process and does not establish dependencies between the hazards in the sense of Definition 1 nor to its trigger conditions. The approach therefore does not address Challenge 1 nor 2, but in part addresses Challenge 3.

The Approach by Sánchez et al.

An approach to trace safety requirements in model-based development proposed in [Sánchez et al. 2011]. The approach is specific to the development of robotic applications. In such applications, hazards do not occur due to trigger conditions in the system’s operational context in the sense of Definition 2, but due to specific “tasks” a robots undertakes. The approach features a traceability metamodel, which describes traces between generic elements. These elements can either be elements from a safety analysis metamodel or a safety-based requirements metamodel. The safety analysis metamodel relates hazards to specific tasks as well as to “hazard solutions.” Hazard solutions in this sense can be understood as hazard-mitigating requirements in the sense of Definition 5. These hazard solutions are elicited in a systematic process: First a “problem analysis step” identifies hazards, tasks, and associated risks. Afterwards, in a manual step, recommendations regarding the mitigation of the hazards are conceived and documented as safety requirements for the system. These safety requirements become part of the system requirements specification together with other system requirements. In subsequent development steps, the implementation of the safety requirements as well as other requirements can be traced to specific functions in the code of the robots. In summary, albeit the approach related mitigations to

hazard and therefore addresses Challenge 1, the approach does not trigger conditions and therefore does not address Challenge 2. Furthermore, the approach makes no mention whether or not the safety requirements can be explicitly differentiated from the remainder of the system requirements specification.

The Approach by Peraldi-Frati and Albinet

An approach specific to the tracing of the satisfaction of requirements in the automotive domain is presented in [Peraldi-Frati and Albinet 2010]. The approach can be divided into three distinct steps: First, hazardous “effects” and their “causes” are identified using FMEA as well as FTA. The results of these analyses are stored in what the authors call a “system safety document,” which they propose to be traced to the requirements specification. In the requirements specification, these effects and causes are refined into safety requirements, which are intended to mitigate the effects. This mitigation relationship is described by means of traceability types proposed in the approach: “decompose,” “derive,” or “copy.” Afterwards, the satisfaction of each safety requirement in subsequent design and code artifacts is documented by means of “satisfy” relationships and test cases can be traced to the satisfaction of the safety requirements by means of “verify” relationships. It is to note that the approach is intended to the tracing of the satisfaction of hazard-mitigating requirements in late stages of development. The notion of a “failure mode” and a “failure effect” in this sense pertains to specific implementation aspects (cf. [Ericson 2005], p. 235ff) as opposed to hazards occurring during early stages of development in the sense of Definition 1 (cf. [Ericson 2005], p. 271ff). Albeit the approach could conceivably be applied to Challenge 1 and 2, it only partly addresses these challenges. Furthermore, the approach makes no mention of differentiating hazard-mitigating requirements in the sense of Definition 5 from the remainder of the requirements specification and hence does not address Challenge 3.

The Requirements Traceability Reference Model

In [Ramesh and Jarke 2001], a generic reference model for requirements traceability is presented. The reference model consists of four submodels, i.e. the requirements managements submodel, the rationale submodel, the design allocation submodel, and the compliance verification submodel. In addition, the reference model describes a more abstract “low-end” traceability model. The low-end model allows tracing requirements to their implementation into components. The management submodel takes a more abstract view on the development process and allows tracing requirements to strategic dependencies as well as system objectives. The

design allocation submodel takes a more detailed view on the allocation of requirements to implementation into components. The difference to the “low-end” model is merely in the level of detail concerning to the contained concepts and, together with the other four submodels, is geared towards complex “high-end” traceability tasks. The other two submodels, i.e. the rationale submodel and the compliance verification submodel, harbor potential application to the challenges from Section 1.2. The aim of the rationale submodel is to provide traceability between implementation decisions and the requirements documenting the decision. This idea could potentially be extended to incorporate “mitigation decisions” such that the rationale submodel documents the mitigations for each hazard and its trigger conditions from a hazard analysis worksheet. Similarly, the aim of the compliance verification submodel is, among other things, to document specific changes to requirements due to verification procedures such as tests, reviews, or inspections. This submodel could be extended to show which specific sets of hazard-mitigating requirements have been elicited in order to mitigate a hazard from the rationale submodel.

3.5. Safety Argumentation

The process of arguing that a system is sufficiently safe is one of the core concerns of safety assessment, as outlined in Section 1.1. Safety argumentation is considered relevant to the fulfillment of the goal of this dissertation (see Section 1.4) insofar that the process of providing evidence for claims regarding the system’s safety (e.g., such as the claim that hazard-mitigating requirements are adequate to mitigate a hazard) may address the challenges outlined in Section 1.2. Safety argumentation is aimed at providing a justifiable and objective argument that all identified hazards have been mitigated and are hence sufficiently unlikely to cause harm. This is done by means of an artifact type called safety cases.

Safety Cases

A safety case is an argument structure according to [Toulmin 1958] that consists of a high-level *claim* concerning the safety of a system, a set of quantitative or qualitative *evidence* satisfying the claim, and an *inference* structure, showing how the evidence satisfies the claim (see [Fenelon and McDermid 1993; Kelly and Weaver 2004]). In the safety engineering literature, other terms such as “assurance case” (e.g. [Bishop et al. 2004]) or “assurance argument” (e.g., [Kelly 2007]) have been used to describe the same idea. Safety cases can be visualized in a tree-like structure using Goal Structuring Notation (GSN, see [Kelly and Weaver 2004]). In GSN safety

cases, the top-level claim is called a safety goal in the sense of Definition 3. Safety goal fulfillment is achieved given specific contextual information. These contextual information could be the hazard's trigger conditions in the sense of Definition 2, but can also be organizational aspects such as laws, standards, or other constraints under which a safety goal must be fulfilled. Hazard-mitigating requirements can serve as evidence for the fulfillment of a safety goal and the inference structure of a safety case can be considered the considered mitigation according to Definition 7. If a safety goal is put in place for each identified hazard, the artifact structure in a GSN safety case can provide explicit dependencies between a hazard and the hazard-mitigating requirements it fulfills (Challenge 1). Moreover, considering the impact of trigger conditions on the fulfillment of a safety goal is also supported by visualizing the dependencies (Challenge 2). However, safety cases merely contain information *that* hazard-mitigating requirements are in place, not *whether* these hazard-mitigating requirements are adequate and do not distinguish hazard-mitigating requirements from other requirements for the purpose of validating the adequacy. In fact, accepting the mere presence of hazard-mitigating requirements as sufficient evidence for the mitigation of a hazard might be seen as argumentatively weak [Hawkins et al. 2011].

Confidence Cases

To alleviate the issue of weak argumentation, it has recently been suggested to augment safety cases with confidence cases (see [Sun 2012]). A confidence case is an “argument that justifies the sufficiency of confidence” ([Hawkins et al. 2011], p. 1) in a safety case. Therefore, in contrast to safety cases, confidence cases are not meant to argue *that* a safety goal is fulfilled, but *why* the safety goal is fulfilled adequately. Confidence cases can be visualized using the same GSN modeling elements as safety cases. In confidence cases, the top-level goal asserts desired safety argument for which confidence is to be demonstrated. The inference associates evidence with the asserted confidence goal, like in safety cases. Evidence in confidence cases take the form of activities carried out in order to show sufficiency of the evidence in the related safety case. For example, such an activity could include the systematic identification and subsequent validation of hazard-mitigating requirements to ascertain their adequacy. The safety argument of a system therefore consists of a safety case containing evidence which demonstrate what mitigations are in place for a hazard and a corresponding confidence case containing evidence which demonstrate why the mitigations are adequate (cf. [Hawkins et al. 2011]). With regard to the challenges from Section 1.2, this means that confidence cases by themselves neither address Challenge 1 nor Challenge 2, but may be used to demonstrate which hazard-mitigating

requirements have been identified and subjected to validation (Challenge 3). Yet, stakeholders need to identify and differentiate the hazard-mitigating requirements from other functional with no impact on safety manually. Moreover, it must be noted that both safety cases and confidence cases are a means to visualize dependencies between safety-relevant information pertaining to a safety argument. Establishing these dependencies, however, is done using alternative means.

3.6. Model-Based Safety Assessment

The term “model-based safety assessment” refers to techniques which are either intended to “facilitate better traceability between safety assessment and design models” ([Lisagor et al. 2010], p. 169) or to create system models based on safety assessment. In [Lisagor et al. 2011], a detailed review of several types of model-based safety approaches is given. According to the authors, these types can be classified using two orthogonal categories, as shown in Table 3-2.

Table 3-2 Classification Scheme of Model-Based Safety Approaches according to [Lisagor et al. 2011].

		Represented Aspects in the Models		
		Only Data, Energy, or Matter Flows	Data, Energy, or Matter with Failure Mode Flows	Only Failure Mode Flows
Model Purpose	Dedicated Safety Model	Failure Effects Modeling	Hybrid Type between Failure Effects and Logic Modeling	Failure Logic Modeling
	Partial Utilization of System Model (e.g., requirements, architecture)	Hybrid Approaches, e.g. Extended System Modeling	Hybrid Approach, e.g. Formal Methods, Error Modeling Annex of AADL	Hybrid Approaches, e.g., Integration of Safety Analysis Results into modeling frameworks
	Automated Construction or Utilization System Model	Failure Injection / Extended System Modeling	-	Hybrid Approaches, e.g., Software Deviation Analysis

In Table 3-2, rows depict the way a model is used, while columns depict the information depicted in the models. Cells represent types of approaches identified in [Lisagor et al. 2011]. For the purpose of this work, approaches pertaining to model-based safety assessment are considered relevant to the fulfillment of the goal of this dissertation (see Section 1.4) as long as they make use of information provided by hazard analyses against which the adequacy of hazard-mitigating requirements is evaluated and depict them together with functional requirements of the system in a unified fashion. This means that approaches that only document data, energy or matter flows (left content column in Table 3-2) as well as approaches that only model failure mode flows (right content column in Table 3-2) are beyond the scope of relevant approaches. Moreover, approaches that make use of a dedicated safety model (top content row in Table 3-2) are excluded also. In the following, we will therefore discuss the relevant type of approach identified in the middle content cell of Table 3-2 (highlighted in light grey). For details on the other types of approaches excluded from this review, please refer to [Lisagor et al. 2011].

As outlined above, approaches pertaining to model-based safety assessment are concerned with integrating safety analysis results into artifacts of the remainder of the system development process. There is a plethora of approaches that belong to this category. In addition to the categorization proposed in [Lisagor et al. 2011], these approaches can further be divided into overlapping subcategories:

- Approaches concerned with formal modeling (e.g., [US NASA 2006; Correa et al. 2010]);
- Approaches concerned with graphical representation of safety analyses (e.g., [Cancila et al. 2009; Beckers et al. 2013; Panesar-Walawege et al. 2013]);
- Approaches concerned with architecture modeling (e.g., [Sandberg et al. 2010; Correa et al. 2010; Kaiser et al. 2010]).
- Approaches concerned with safety standard compliance (e.g., [Kaiser et al. 2010; Sandberg et al. 2010; Panesar-Walawege et al. 2013; Beckers et al. 2013]).

In the following, examples of these approaches are summarized.

The Approach by Joshi and Heimdahl

In [Joshi and Heimdahl 2005a; Joshi and Heimdahl 2005b; US NASA 2006], a model-based safety analysis approach is proposed, which is specifically aimed at improving knowledge transfer between system developers and safety engineers. The approach is based on a modified development process model [Joshi and Heimdahl 2005a], which adds system level hazard analysis steps and fault tree analysis steps after an initial set of system requirements are conceived (see also [Pumfrey 1999] as well as Section 1.1). Based on the safety analysis steps in early stages of development, the derived safety requirements are formalized. The output of this step is a formal system model, which incorporates safety aspects as well as system development artifacts, which become input for further development activities. The central idea of this approach is that safety engineers and system developers work on a common, formal system model [Joshi and Heimdahl 2005b]. The developer using the approach is free to choose some formalization technique to formalize the system model [US NASA 2006]. For example, the authors propose temporal or higher-order predicate logic languages or modeling languages such as UML, SIMULINK, or SCADE (see [Joshi and Heimdahl 2005a; US NASA 2006] for details). Once formalization is complete, the system model can be subjected to further safety engineering techniques, such as formal proofs, simulation, verification, and model-checking. The approach makes no mention of special consideration of hazards, trigger conditions, or hazard-mitigating requirements and is more akin to a formal verification method (see also Section 3.3.2), seam-

lessly integrated into the development process, than an approach focused on graphical representation of dependencies as outlined in Section 1.4. In this sense, hazards as well as their trigger conditions become “axioms,” which must be disproven by showing that “safety properties” hold for the entire system model [US NASA 2006]. The approach is hence not concerned with hazard-mitigating requirements in the sense of Definition 5 (Challenge 3), the impact of hazard-mitigating requirements on trigger conditions (Challenge 2), nor with the dependencies of hazard-mitigating requirements to hazards (Challenge 1).

The SOPHIA Modeling Language

In [Cancila et al. 2009], the SOPHIA Modeling Language is presented. SOPHIA is a SysML-compatible language intended to improve the traceability between safety engineering artifacts and the artifacts produced during other development activities. SOPHIA is a meta-language, which proposes fundamental modeling concepts to document system accidents, i.e. operational scenarios that result in harm, as well as the relative risk and probability accidents have. For this purpose, three packages are defined: the “Accidents” package, which defines occurrence frequencies and relative risk of hazards, accidents, and accident consequences, the “Mitigations” package, which defines the measures taken to mitigate an accident, and a “FaultContainment-Region” package, which documents error propagations. Albeit there is no concrete visual syntax proposed in [Cancila et al. 2009], the authors suggest three integration strategies with SysML. On the one hand, the fundamental SOPHIA concepts can be implemented as a SysML extension using a UML profile in the sense of [Fuentes-Fernandés and Vallecillo-Moreno 2004; Selic 2007]. On the other hand, SOPHIA can be implemented as a stand-alone domain-specific language in the sense of [Kelly and Tolvanen 2008]. The third strategy treats SOPHIA fundamentals concepts as an orthogonal meta-model to SysML or other languages. SOPHIA does not prescribe nor restrict the specific system develop artifacts that can be traced to accidents. In this sense, SOPHIA is concerned with documenting dependencies between accidents and development artifacts at large. Albeit hazards play a role in SOPHIA, the language is more concerned with focusing on accident mitigation. The language hence partly addresses Challenge 1. Yet, SOPHIA is neither concerned with hazards’ trigger conditions, nor with the specific distinction between hazard-mitigating requirements and other development artifacts and therefore does not address Challenge 2 nor Challenge 3.

The Approach by Beckers et al.

In [Beckers et al. 2013], a model-based hazard analysis and risk assessment approach is presented. Like the approach by Kaiser et al., the approach by Beckers et al. is [ISO26262 2011]-compatible. Yet, in contrast to [Kaiser et al. 2010], the approach in [Beckers et al. 2013] is not concerned with creating a safety concept, i.e. a document summarizing how hazards have been mitigated by the system design, but with documenting the relative risk in terms of the Automotive Safety Integrity Level. The approach entails a seven-step process: After item definitions, i.e. a summary of the functionality of a system under development along with its interaction with the operational context, is defined, “fault-type guidewords” are instantiated. The purpose of this step is to assist the developer later on in identifying hazards. Afterwards, a situation classification step is executed. This step entails the identification of scenarios, in which the hazards identified in the previous step occur. This includes not only the operational conditions, but also their relative occurrence probability. Given the guidewords and classified situations, hazard analyses are conducted and documented in a tabular form, much like the hazard analyses worksheets proposed in [Ericson 2005] (p. 276). Once a list of hazards have been identified, the hazards are classified according to severity, exposure, and controllability as proposed in [ISO26262 2011] and safety goals are defined. In the last step, an independent review as required by [ISO26262 2011] is conducted. These reviews are informed by “validation conditions” proposed in the approach, which are system-agnostic well-formedness rules that must hold for the hazard and risk assessment results. By making use of a metamodel for the hazard analysis and risk assessment results, the approach allows importing the results into a design model of the system. Doing so enables tracing of hazards as well as their trigger conditions to the specific functional safety requirements that mitigate the hazard. The approach hence completely addresses Challenge 1 and partially addresses Challenge 2. However, the approach is focused on the hazard and risk assessment process up to the point of defining safety goals and hence does not concern the distinction of hazard-mitigating requirements from other functional requirements with no impact on safety in the sense of Challenge 3.

Error Modeling in EAST-ADL2 or AADL

In several publications, approaches have been proposed to extend architecture description languages by a means to model error occurrence and failure propagation. For example, in [Chen et al. 2008; Biehl et al. 2010; Sandberg et al. 2010], EAST-ADL2 architectural building blocks of the system [Chen et al. 2008] are integrated with GSN safety cases according to [Kelly and Weaver 2004] (see Section 3.5) on a process level. Safety architecture components are then

subsumed in a feature model of the system [Sandberg et al. 2010], which represents the mitigation for errors identified during preliminary hazard analyses [Biehl et al. 2010]. Based on a common domain model, the EAST-ADL2 architecture artifacts of the system can be traced to the hazards as well as the safety requirements that have been defined in order to fulfill the safety goal in the safety case. The term “safety requirements” in the approach refers to architectural constraints, similar to “technical safety requirements” in [ISO26262 2011]. This mainly comprises constraints to avoid particular errors and failures during operation. Hence, the approach is not concerned with hazards in the sense of Definition 1 (see also Assumption 2 in Section 1.3), but with errors that occur during operation due to specific properties in the design or implementation. In consequence, the approach is also not concerned with hazard-mitigating requirements in the sense of Definition 5 (Challenge 3). Consideration of trigger conditions in the sense of Definition 2 is not discussed in the approach as well (Challenge 2).

Similarly, in [Correa et al. 2010], AADL has been extended with the ability to model errors as well as their propagation, much like in [Biehl et al. 2010], but for the purpose of simulation rather than formal verification. The approach consists of six step process. At first, the requirements of the system under development are elicited and the system’s functions are modeled in order to simulate their interactions at runtime. Afterwards, the aspects of the operational context in which the system will operate once it is deployed is formalized and used to configure the simulation. Then, in an iterative process, the system’s functionality is refined into executable components, which are described in terms of their runtime behavior (e.g., threading, thread modes, etc.), and enhanced with execution time properties (e.g., worst case execution times, etc.). Finally, based on this information, simulation is conducted, in which the execution behavior of the system model is verified against safety constraints specified in temporal logic. The error modeling approach for AADL proposed in [Correa et al. 2010] is therefore reminiscent of a real-time verification approach (see also Section 3.3.2). Real-time constraints are considered to be safety requirements and hazards are violations of such constraints. Hazards in the sense of Definition 1 (Challenge 1), trigger conditions in the sense of Definition 2 (Challenge 2) or hazard-mitigating requirements in the sense of Definition 5 (Challenge 3) are not considered.

The Approach by Kaiser et al.

In [Kaiser et al. 2010], another approach concerned with the integration of safety analysis results and system architecture descriptions is presented. In contrast to [Sandberg et al. 2010; Correa et al. 2010], however, the approach by Kaiser et al. is not limited to error modeling and

does not prescribe a specific architecture language to be used. Instead, the architecture is described using generic modeling languages, such as UML or SysML. Once the architecture is documented, FMEA are conducted in order to identify system functions, malfunctions, and their effects. The FMEA results are documented in a failure net, which is a hierarchical structure similar to the architecture description and allows tracing of failures through the architecture. In a subsequent step, safety goals are defined in order to avoid the failure and the system architecture is refined by means of “safety measure.” Safety measures are mitigations in the sense of Definition 6, which comprise specific requirements to mitigate the failures identified in the failure net. Lastly, based on the architecture, the identified failures, conceived safety goals, and the documented safety measures, these information are comprised into a safety concept in the style of [ISO26262 2011]. An ISO26262 safety concept is an artifact, which documents the dependencies between these items and allows certification authorities to trace possible failures from origin to their mitigation. Albeit the approach in [Kaiser et al. 2010] is concerned with “failures” as opposed to hazards in the sense of Definition 1, the approach established dependencies between failures and their mitigation. This partly addresses Challenge 1. Moreover, due to the nature of FMEA, it is possible to trace the trigger conditions of a failure (i.e. “failure mode,” see [Ericson 2005], p. 235), thereby partly addressing Challenge 2. Safety requirements that are part of the safety concept, but become a constituent of the requirements specification that are not explicitly documented as such. The approach hence partly addresses Challenge 3.

The Approach by Panesar-Walawege et al.

In [Panesar-Walawege et al. 2013], a model-based approach to prove compliance with safety standards such as [ARP4761 1996; ISO26262 2011] is presented. The central aspect of the approach is the creation of a conceptual model of the standard that compliance shall be established with. This allows establishing dependencies between the system models and the modeling elements documented therein and the concepts of the standard in question in order to allow manufacturers and suppliers to demonstrate compliance with the standard. The proposed approach makes use of UML profiles to establish these dependencies between concrete models. To do so, the approach defines a process consisting of four steps, which closely resembles the lightweight extension approach proposed by Lagarde et al. (see also [Lagarde et al. 2007; Lagarde et al. 2008]. The first step in this process is concerned with creating a conceptual model of the standard. This is akin to steps 1 and 2 in [Lagarde et al. 2007; Lagarde et al. 2008]. Once a sufficiently stable conceptual model of the standard exists, a UML profile of the standard is created. This allows integration of the conceptual model of the standard with the system models

developed for the system under development and is similar to step 3 in [Lagarde et al. 2007; Lagarde et al. 2008]. Based on the UML profile of the standard and the system models, an elaboration step is carried out, in which the compliance of the standard is ascertained. This step is similar to the domain analysis and profile optimization steps in [Lagarde et al. 2007; Lagarde et al. 2008], as the approach in [Panesar-Walawege et al. 2013] considers the system the domain to be certified. In this sense, this step entails the creation of dependencies between the conceptual model of the standard and the domain model of the system to be certified along with invariants that must hold between the models. The result of this step is a mapping between the system and the standard, which is instantiated for a specific certification task in the final step of the approach. In this sense, the approach is capable of documenting and visualizing dependencies between hazards and hazard-mitigating requirements. However, this is only the case if the conceptual model of the safety standard as well as the domain model of the system under development allow for such modeling elements. Challenge 1 is therefore only partly addressed. Similarly, the approach allows considering trigger conditions in the sense of Definition 2 only if the safety standard as well as the domain model allow for such modeling concepts. Challenge 2 is therefore partly addressed as well. The approach treats the information specified in the system models as abstract system requirements, which must be concretized for a certification task. The approach is not concerned with hazard-mitigating requirements in the sense of Definition 5 and hence does not address Challenge 3.

3.7. Evaluation Summary of the State of the Art

In Section 1.2, the following three challenges regarding the validation of adequacy of hazard-mitigating requirements were identified:

- Challenge 1: Stakeholders have no indication how hazards have been mitigated.
- Challenge 2: Stakeholders have no indication how trigger conditions are influenced by the hazard-mitigating requirements.
- Challenge 3: Stakeholders have no indication which hazard a specific hazard-mitigating requirement is related to.

In Section 3.1.2, these challenges were concretized into evaluation criteria for the state of the art. Table 3-3 summarizes the findings of the analysis of the state of the art against the evaluation criteria using the symbols regarding fulfillment of challenges from Table 3-1.

Table 3-3 Evaluation Summary of Related Works.

Approach Type	Approach	Challenge 1	Challenge 2	Challenge 3
		Indication, how hazards were mitigated	Indication, how trigger conditions influence a hazard's occurrence.	Differentiate HMR from remainder of the FRs.
Goal- and Scenario Based Elicitation of Hazard-Mitigating Requirements	KAOS	●	⊙	⊙
	i* and derivatives	⊙	○	○
	Menon and Kelly	⊙	○	⊙
	Kotonya and Sommerville	○	●	○
	Allenby and Kelly	●	⊙	○
	Dittel and Aryus	⊙	○	○
	Misuse Cases	⊙	○	⊙
Formal Elicitation of Hazard-Mitigating Requirements	Chen and Motet	○	○	●
	Zafar and Dromey	⊙	○	○
	STPA	●	●	○
	Hansen et al.	○	○	⊙
	Troubitsyna	●	○	○
	Belli et al.	○	○	●
Approaches to Support Requirements Validation	Perspective-based Reading	⊙	⊙	○
	Value-based Reading	○	○	⊙
	Checklist-based Reading	⊙	⊙	○
	SafeSpection	●	●	○
Formal Quality Assurance	Heitmeyer et al.	●	●	⊙
	Snelting et al.	○	●	○
	Cheung and Kramer	○	●	○
	Saeed et al.	●	○	○
Approaches Pertaining to Traceability of Safety Requirements	Trace Query Approach	●	○	⊙
	Softgoal Approach	⊙	○	⊙
	Event-Based Approach	⊙	⊙	○
	Safety Artifact Traceability Model	⊙	○	⊙
	Hill and Tilley	○	○	⊙
	Sánchez et al.	●	○	○
	Peraldi-Frati and Albinet	⊙	⊙	○
	Requirements Traceability Reference Model	⊙	⊙	⊙
Approaches Pertaining to Safety Argumentation	Safety Cases	●	●	○
	Confidence Cases	○	○	⊙
Approaches Pertaining to Model-Based Safety Assessment	Joshi and Heimdahl	○	○	○
	SOPHIA	⊙	○	○
	Beckers et al.	●	●	○
	Error Modeling in EAST-ADL2 or AADL	○	○	○
	Kaiser et al.	⊙	⊙	⊙
	Panesar-Walawege et al.	⊙	⊙	○

In Table 3-3, “●” denotes that a challenge is completely solved. The symbol “⊙” denotes that a challenge is partially solved, and “○” indicates that a challenge is not addressed. As can be seen, there is no single approach, which completely addresses all three challenges. Albeit some

approaches can in principle be used to support stakeholders with certain aspect during the validation of the adequacy of hazard-mitigating requirements, the analysis of the state of the art has confirmed the *gap in research* with regard to the goal outlined in Section 1.4.

A warranted question becomes whether or not tailoring or combining some of the approaches outlined above may be applicable. Such an undertaking is burdened by the specific purpose and underlying assumptions of the approaches. For example, formal quality assurance approaches (see Section 3.2.2) as well as formal model-based safety assessment and error modeling approaches (see Section 3.6) make the implicit assumption that correctness and fault tolerance alone is sufficient to establish safety. Yet, as has been argued in depth by many authors (e.g., [Heimdahl 2007; Leveson 2011a; Hatcliff et al. 2014]), a formally correct system (i.e., where the development artifacts are contain no formal error), may still be hazardous during operation. Similarly, elicitation approaches (see Section 3.2) promise to generate “constructively correct” requirements and typically allow the stepwise refinement (and tracing of dependencies) from the origin to the implementation of a requirement (i.e. “post-traceability” in the sense of [Gotel and Finkelstein 1994]). While these are valuable and useful contributions for the purpose of this work, they do not relieve the developer from validating whether or not the hazard-mitigating requirements are adequately mitigate a hazard. Conducting validation in this sense is mandated by safety standards (see, e.g., [ARP4761 1996; ISO26262 2011]).

A promising avenue to tailor and combine existing approaches is to combine Leveson’s STPA approach [Leveson 2004] with a model-based assessment approach, which makes heavy use of conceptual models (e.g., [Beckers et al. 2013; Panesar-Walawege et al. 2013]). Such a combined approach would yield dependencies on a meta-model level between hazards, trigger conditions, and the hazards’ mitigations and would allow for traceability in the sense of, e.g., [Sánchez et al. 2011; Cleland-Huang et al. 2012] such that hazard-mitigating requirements can be identified (see Section 3.4). Through extension of modeling languages these dependencies can be visualized to support validation. The diagrams instantiating these conceptual models can be subjected to validation using, e.g., perspective-based, checklist-based, or value-based reading or safety-specific validation processes like SafeSpection (see Section 3.3.1), to ascertain the adequacy of the hazard-mitigating requirements. Our approach can be seen as an extension of these concepts. However, rather than combining and extending the approaches outlined above, a novel, ontology-centric approach was developed. This was done because of the following reasons:

- **Generic Applicability of the Approach.** Leveson’s STPA approach [Leveson 2004] focuses on a systems-theoretic hazard causation model, but requires a tailored and safety

assessment-centric development process. It replaces traditional development processes rather than integrating the safety assessment process with the development process. In order to maintain a high degree of adaptability and generic applicability, the approach developed in this dissertation makes use of the systems-theoretic understanding of hazards and their causes (see [Leveson 2011a]), yet focuses on ontological integration of the work products to integrate work products of requirements engineering and hazard analyses, as outlined in Section 1.2.

- **Tailoring of Ontologies.** The approaches proposed by [Beckers et al. 2013] and [Panesar-Walawege et al. 2013] make use of ontologies to establish dependencies between hazard analysis results and safety requirements. In [Beckers et al. 2013], the purpose is to support risk assessment, while in [Panesar-Walawege et al. 2013], the purpose is to provide proof of adherence to safety standards. These ontologies must be adapted to express the dependencies between safety requirements and hazard analysis worksheets using the systems-theoretic understanding of the causation of hazards (cf. Section 2.1). Therefore, the approach developed in this dissertation builds on the idea of using ontologies to document dependencies between safety-related requirements and hazards.
- **Visual Notation for Validation.** The traceability-centric approaches in [Sánchez et al. 2011; Cleland-Huang et al. 2012] as well as the model-based safety assessment approaches in [Beckers et al. 2013; Panesar-Walawege et al. 2013] establish traces between hazards and requirements, yet, the trace relationships produced by the respective approaches are not visualized graphically. To visualize the trace relationship, a graphical notation must be developed to represent the trace relationships and ontological concepts.

Part II:
Hazard Relation Diagrams

4. Overview

In Section 1.5, we presented an overview over the solution approach. The principle idea of the solution approach is to define *Hazard Relation Diagrams* in order to visualize the dependency between the specific *hazard-mitigating requirements* (in the sense of Definition 5) that make up the *conceptual mitigation* (in the sense of Definition 7) of the one *hazard* (in the sense of Definition 1). Hazard Relation Diagrams furthermore visualize the dependency between the hazard-mitigating requirements and the contextual information about the hazard, i.e. the *trigger conditions* (in the sense of Definition 2) and the *safety goal* (in the sense of Definition 3) that were identified for that hazard during hazard analysis. These dependencies are described in the ontology for Hazard Relation Diagrams. An overview over the dependencies within Hazard Relation Diagrams is given in Section 4.1. Section 4.2 briefly overviews the creation approach before Section 4.3 overviews tool support for Hazard Relation Diagrams.

4.1. Modeling Concepts of Hazard Relation Diagrams

Figure 4-1 exemplarily illustrates the instantiations of the dependencies in Hazard Relation Diagrams. The hazard-inducing requirements *HIR 1* causes all three hazards from the hazards analysis worksheet (dashed light grey double-headed arrows). Dashed boxes represent partial mitigations according to Definition 6. The hazard-mitigating requirements within the partial mitigation are depicted as a box with an inverted corner labeled “HMR.” Dependencies between partial mitigations and mitigated hazards are depicted as solid black double-headed arrows.

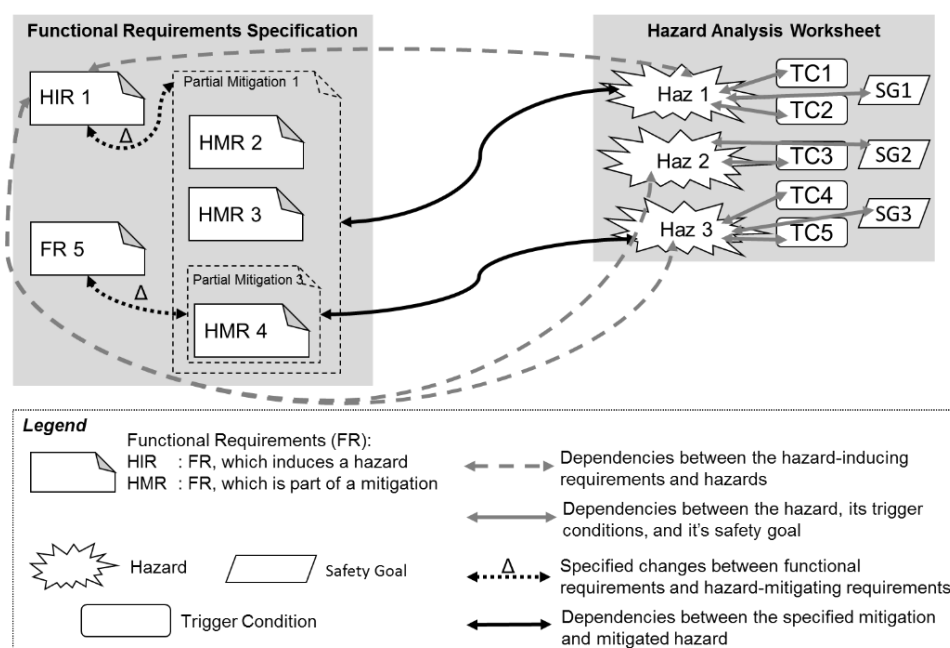


Figure 4-1 Example for the Definition of Ontological Relationships between Artifact Types.

As can be seen, the partial mitigation comprises the hazard-mitigating requirements in the sense of Definition 5 and Definition 6. These hazard-mitigating requirements are defined as specific changes to the functional requirements. To this end, the approach makes use of corresponding template-based format to specify mitigations called a “*mitigation template*.” For each hazard in the hazard analysis worksheets, a mitigation template is specified, detailing the hazard-mitigating requirements needed to mitigate the respective hazard. Doing so will make explicit how the hazards have been mitigated, i.e. which hazard-mitigating requirements have been defined to mitigate which hazards and allows instantiating the dependencies for the purpose of visualization. In Assumption 4 in Section 1.3, we have assumed that functional requirements are documented using UML activity diagrams. This allows defining Hazard Relation Diagrams as extensions of UML activity diagrams as well, using the UML profile mechanism defined in MOF [OMG 2015a]. Chapter 5 discusses the ontological foundations, visual notation, and well-formedness rules of the modeling concepts of Hazard Relation Diagrams.

4.2. Creation Approach for Hazard Relation Diagrams

As discussed in Section 4.1, Hazard Relation Diagrams are based on an ontology, which extends UML activity diagrams by concepts from the hazards analysis worksheets. This means that Hazard Relation Diagrams must be created by unifying modeling elements from UML activity diagrams and modeling elements that represent hazard analysis worksheet concepts. In this sense, Hazard Relation Diagrams require the integration of two meta-models, i.e. UML activity diagrams defined in MOF and the ontology underlying hazard analysis worksheets. OMG QVT Operational Mappings (QVTo, see [OMG 2016]) have been selected as the approach to create Hazard Relation Diagrams, as QVTo transformation scripts allow a stepwise transformation from hazard-inducing requirements to hazard-mitigating requirements and furthermore addition of extraneous modeling elements to represent information from the hazard analysis worksheets. Chapter 5.5 discusses the approach to create Hazard Relation Diagrams. In Section 7.3, the tool prototype implementing this approach is discussed.

4.3. Tool Support for Hazard Relation Diagrams

The approach taken in this dissertation is focused on extending UML activity diagrams by the ability to visualize the dependencies between hazard-mitigating requirements and hazard analysis results. The approach reuses existing UML activity diagram modeling elements to document functional requirements as well as their semantics. Therefore, the lightweight extension approach by Lagarde et al. is applicable. In [Tenbergen et al. 2013], the approach by Lagarde

et al. has been adapted in order to support the definition of method-specific profiles, which is used to meet the specific goal of this dissertation (see Section 1.4). Section 5.1 explains the ontological foundations pertaining the definition of Hazard Relation Diagrams and thereby describe how Hazard Relation Diagrams extend UML activity diagrams. In Section 7.2, the UML profile is discussed, which implements the extensions from Section 5.1.

5. Modeling Concepts of Hazard Relation Diagrams

We introduce the underlying modeling concepts Hazard Relation Diagrams outlined in Section 4.1. The principles underlying Hazard Relation Diagrams have first been described in [Tenbergen et al. 2015] and extensions have been proposed in [Tenbergen et al. 2017]. In Section 5.1, the ontological foundations is discussed. Section 5.2 explains their integration into the existing UML activity diagram ontology. Section 5.3 presents the visual notation of the modeling concepts for functional requirements as well as for the extensions specific to Hazard Relation Diagrams. In Section 5.4, we illustrate how Hazard Relation Diagrams can be used to address the non-trivial multiplicities between hazards and hazard-mitigating requirements outlined in Section 1.2. Finally, in Section 5.5, we introduce well-formedness rules of Hazard Relation Diagrams that are based on the ontological foundations from Section 5.1.

5.1. Ontological Foundations

By defining conceptual mitigations for each hazard, which subsume partial mitigations, the chosen approach supports the different types of multiplicity between hazards and hazard-mitigating requirements discussed in Section 1.1. For this reason, Hazard Relation Diagrams contain exactly one hazard and one conceptual mitigation. This way, each Hazard Relation Diagram will contain all hazard-mitigating requirements that are part of the conceptual mitigation for the same hazard. During validation, each Hazard Relation Diagram is reviewed individually, thereby allowing for each conceptual mitigation to be validated.

In order to visualize these dependencies, Hazard Relation Diagrams comprise the following modeling concepts:

- **Hazard.** The modeling concept *hazard* represents a hazard (in the sense of Definition 1) identified during hazard analysis (and hence represents a trace thereto). During validation, the adequacy of hazard-mitigating requirements must be judged with regard to this hazard, for example, to assess whether or not its harmful effects have been reduced or controlled (i.e. made less likely to occur or possible harm has lessened, see [Leveson 1995]).
- **Trigger condition.** The modeling concept *trigger condition* represents the trigger conditions of the hazard (in the sense of Definition 2) identified during hazard analysis. Incorporating the trigger conditions into the Hazard Relation Diagrams allows stakeholders to assess whether or not the conceptual mitigation was successful in preventing the hazard from occurring in the first place (see [Leveson 2011a]). All trigger conditions for the specific hazard featured in the Hazard Relation Diagram must be depicted.

- **Safety goal.** The modeling concept *safety goal* represents the goal in the sense of Definition 3 that has conceived as part of the hazard analysis in response to the hazard. This enables stakeholders to assess whether the hazard-mitigating requirements adequately refine the safety goal and hence serve as a conceptual mitigation for the hazard.
- **Hazard-mitigating requirements.** The *hazard-mitigating requirements* (see Definition 5) in Hazard Relation Diagrams represent the requirements whose adequacy are to be validated. For this, it is necessary that between conducting hazard analyses and conducting validation, an engineering process took place, during which hazard-mitigating requirements were elicited and documented (see Section 1.3).
- **Partial mitigation.** In model-based specifications, diagrams typically contain hazard-mitigating requirements that pertain to multiple hazards and may also contain model elements, which are unspecific for any conceptual mitigation (e.g. other requirements that are not safety-relevant, annotations, or the like). It is therefore necessary to differentiate the hazard-mitigating requirements that pertain to some particular hazard from those that pertain to any other hazards or other model elements. The modeling concept *partial mitigation* in a Hazard Relation Diagram subsumes all hazard-mitigating requirements belonging to the specific hazard depicted in the Hazard Relation Diagram. It is to note, that as outlined in Section 1.1, there might be overlap in hazard-mitigating requirements, i.e. a hazard-mitigating requirement may aid in mitigating two or more different hazards. In this case, this specific hazard-mitigating requirement is subsumed by two or more partial mitigations, but in different Hazard Relation Diagrams (see Section 5.4 for more details).
- **Hazard Relation.** The *Hazard Relation* is the central modeling concept, which associates exactly one *hazard* to its *trigger conditions*, *safety goal*, and the *hazard-mitigating requirements* to be validated, subsumed by the *partial mitigation*. A Hazard Relation can hence be thought of as an n-ary association between these modeling concepts. Each Hazard Relation Diagram contains exactly one Hazard Relation.

5.2. Integration with UML Activity Diagrams

In [Tenbergen et al. 2015], we have introduced Hazard Relation Diagrams as an extension of UML/SysML activity diagrams. These were extended in [Tenbergen et al. 2017] and are shown in the ontology in Figure 5-1.

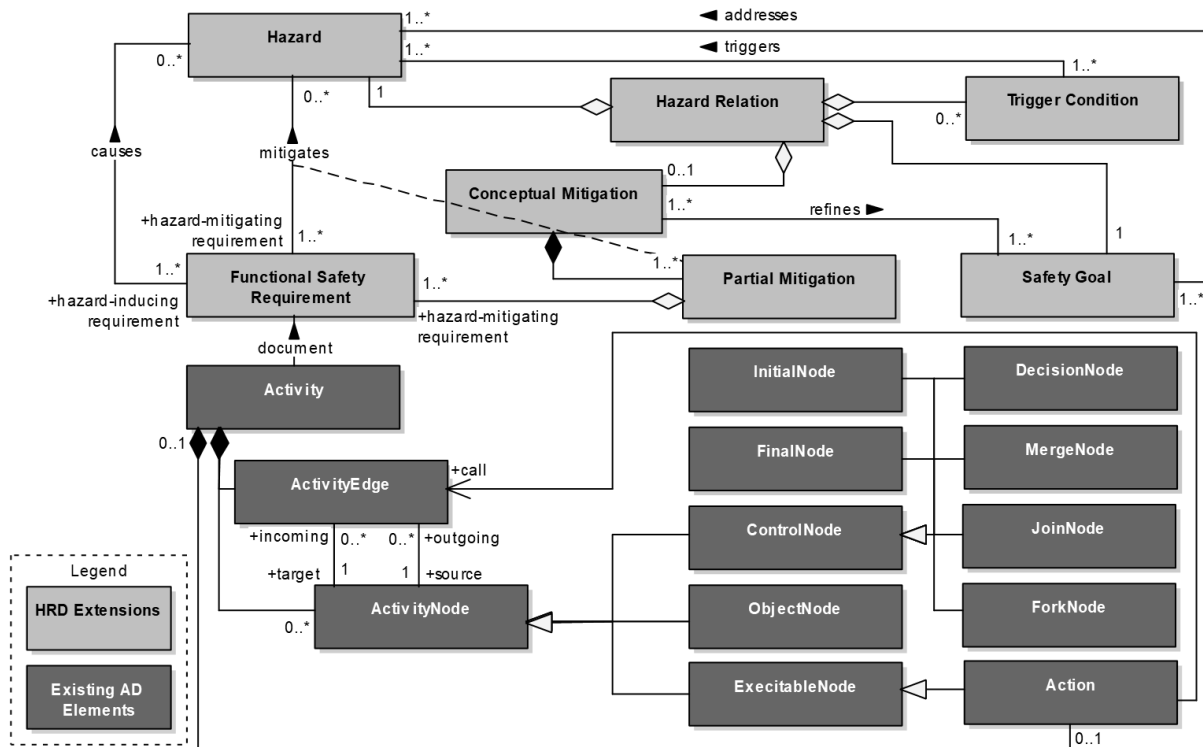


Figure 5-1 Ontological Foundation of Hazard Relation Diagrams.

In Figure 5-1, the modeling concepts which constitute the extensions specific to Hazard Relation Diagrams and are depicted in light grey boxes. Dark grey boxes depict the elements of the UML meta-model for conventional activity diagram from [Störle 2004] upon which Hazard Relation Diagrams build. It is to note that in accordance with Definition 4 and Definition 5, the ontological element *Functional Safety Requirement* subsumes both hazard-inducing and hazard-mitigating requirements, since depending on the development situation, a functional safety requirement can have either role. In either case, UML activities and activity diagram control structures (i.e. fork, join, decision, and merge nodes) are used to document functional safety requirements. As can be seen, the core of a Hazard Relation Diagram is the *Hazard Relation* which associates one *hazard* to its set of *trigger conditions*, *safety goals*, and at least one *partial mitigation*. A partial mitigation (see Definition 6) subsumes a set of hazard-mitigating requirements (see Definition 5). It is to note that functional requirements for a system are in practice not specified in a single but in multiple activity diagrams. In consequence, partial mitigations could comprise hazard-mitigating requirements scattered across multiple diagrams, as we will illustrate in Section 5.4. In the following Section 5.3 shows the visual notation for this ontology.

5.3. Visual Notation for the Modeling Concepts

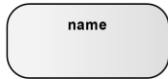

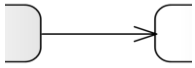
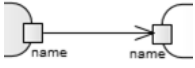


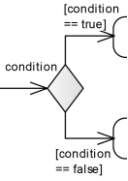
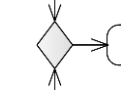
A visual language for Hazard Relation Diagrams was developed in accordance with Moody’s “Physics of Notations” principles outlined in [Moody 2009]. This visual language consists of

two components: a collection of diagrammatic elements to depict functional requirements (Section 5.3.1) and a collection of diagrammatic elements to depict the modeling concepts defined for Hazard Relation Diagrams (Section 5.3.2). In Section 5.3.3, an example of the usage of the visual notation is presented.

5.3.1. Visual Notation for Functional Requirements

As outlined in Section 5.2, Hazard Relation Diagrams are an extension of UML activity diagrams. Hence, in order to depict functional requirements, UML activity diagram notation is used. Table 5-1 shows the notational elements defined in UML activity diagrams used to depict functional requirements in Hazard Relation Diagrams.

Table 5-1 UML Activity Diagram Elements Used to Depict Functional Requirements in Hazard Relation Diagrams.

Modeling Concept	Diagrammatic Element	Description of Visual Notation	Example Visual Notation
Functional Requirement	Action / Activity	Rounded edge rectangle bearing the name of the action or activity.	
	ObjectNode / Input Pin / Output Pin	Square placed on the border of an action or activity bearing the name of the object being transmitted.	
	Control Flow	Directed arrow between two activities. Corners or bends are permissible.	
	Data Flow	Directed arrow between two object nodes.	
	Fork Node	Horizontal or vertical bold black bar with one incoming control or data flow and at least two outgoing control or data flows.	
	Join Node	Horizontal or vertical bold black bar with at least two incoming control or data flows and one outgoing control or data flow.	
	Decision Node	Diamond shape “standing” on one corner, bearing the decision condition as the name, and with one incoming control or data flow and at least two outgoing control or data flows, respectively. Guards in square brackets represent disjoint decision alternatives.	
	Merge Node	Diamond shape “standing” on one corner with two incoming control or data flows and one outgoing control or data flow, respectively.	








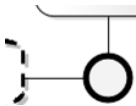
As outlined in Section 2.2, functional requirements comprise both *hazard-mitigating requirements* and *hazard-inducing requirements*. In Hazard Relation Diagrams, this comprises UML actions, fork-, join-, decision-, and merge nodes, as well as control flows and data flows (see

Moody’s principles of Cognitive Fit and Cognitive Integration in [Moody 2009]). Safety assessment requires a precisely defined scope of the system under development in order to be able to ascertain effect of the system onto its operational context under consideration of the inputs the system receives from humans or other systems [Ericson 2005]. Therefore, the *system boundary* is documented in Hazard Relation Diagrams using UML activities, which contain the aforementioned modeling elements to document the system’s functional requirements (see Moody’s Principle of Complexity Management in [Moody 2009]). System inputs and system outputs are depicted using input pins and output pins, respectively. ObjectNodes are used to depict input parameters and output parameters for actions. Data flows are directed activity edges between input pins, output pins, or ObjectNodes. Control flows are seen as “primitive” data flows, which merely call the next action, and are documented as edges with no ObjectNodes.

5.3.2. Visual Notation for Hazard Relation Diagram Extensions

As discussed in Section 5.2, Hazard Relation Diagrams extend UML activity diagrams with a minimal set of additional modeling concepts (see Moody’s Principle of Graphic Economy in [Moody 2009]). These modeling concepts are shown in Table 5-2.

Table 5-2 Modeling Concepts of Hazard Relation Diagrams and their Visual Notation.

Modeling Concept	Diagrammatic Element	Description of Visual Notation	Example Visual Notation
Hazard	Hazard	Rectangle featuring a flash symbol in the middle and bearing the stereotype <<Hazard>> as well as the name of the hazard.	
Trigger Condition	Atomic Trigger Condition	Rounded edge rectangle with a dashed border bearing the name of the condition.	
	Trigger Condition Conjunction	Circle featuring two ampersand characters.	
	Trigger Condition Disjunction	Circle featuring two vertical lines (i.e. “pipe” operators)	
Safety Goal	Safety Goal	Rectangle bearing the stereotype <<Safety Goal>> as well as the name of the safety goal.	
Partial Mitigation	Mitigation Partition	Transparent rounded edge rectangle bearing the word “Mitigation.”	
Hazard Relation	Hazard Relation	Empty circle with a bold border.	
Hazard Association	Hazard Association	Line connecting a Hazard Relation with either a Hazard, with a Safety Goal, a Trigger Condition tree, and at least one Mitigation Partition. Corners and bends are permissible.	

As can be seen in Table 5-2, *Hazards* are depicted using UML class-style boxes with a lightning bolt in the middle to symbolize the potential for harm (see Moody’s principles of Semiotic Clarity and Semantic Transparency in [Moody 2009]). In the literature, a number of visual notations for the modeling concept “goal” have been proposed, e.g. a parallelogram in KAOS [Darimont and van Lamsweerde 1996], a rounded-edge rectangle or variant thereof in i* [Yu 1997], a cloud-symbol in Tropos [Giorgini et al. 2005], or a rectangle in GSN [Kelly and Weaver 2004]. Since Hazard Relation Diagrams extend UML’s meta-object facility and because Hazard Relation Diagrams can conceivably be used in combination with any of these goal modeling languages, no specific shape for *safety goals* is mandated. Instead, safety goals are modeled using a UML class stereotyped “<<Safety Goal>>,” thereby allowing future extensions to derive dedicated symbols in combination with other model-based development approaches (see also the principles of Cognitive Integration and Dual Coding in [Moody 2009]).

The hazard’s *trigger condition* is documented using rounded-edge rectangles, similar to UML states. In order to emphasize that these states are not system states, but conditions in the assumed operational context of the system, the trigger conditions are depicted using a dashed border. Furthermore, since trigger conditions in FHA may also be complex Boolean expressions rather than individual states (see Table 2-1), condition and disjunctions can be used in combination with trigger conditions to graphically represent Boolean expressions as a binary tree. Conjunctions and disjunctions between atomic trigger conditions are represented using node-elements bearing ampersand and “pipe” symbols reminiscent of modern programming languages (e.g. Java).

In Hazard Relation Diagrams, the partial mitigation outlined in Section 5.1 (see also Definition 6) are depicted using bold, dashed, rounded-edge *mitigation partitions* which encapsulate the *hazard-mitigating requirements*. This follows Moody’s Principle of Perceptual Discriminability (see [Moody 2009]). It is to note, that there may be multiple mitigation partitions (and, consequently, also multiple system boundaries) within one Hazard Relation Diagram, as is discussed in Section 5.4 in more detail.

A *Hazard Relation* can hence be thought of as an n-ary association between the hazard, the safety goal, the binary tree of trigger conditions, and the mitigation partitions. In UML class diagrams, n-ary associations are represented as diamond shapes, which are visually identical to UML activity diagram decision and merge nodes. Since decision and merge nodes can occur (as part of the UML activity diagram modeling elements) in Hazard Relation Diagrams, this may lead to confusion (see Moody’s principles of Visual Expressiveness and Semiotic Clarity in [Moody 2009]). Hence, Hazard Relations are depicted using a circle with a bold border and

associate the hazard to its safety goal, its trigger conditions and to the mitigation partition by means of individual *hazard associations*. Hazard associations are depicted using simple, undirected UML class diagram-type associations. Hazard associations bear no role and multiplicity information, as the roles are determined by the respective modeling elements (e.g. a hazard can only have the role of a hazard) and multiplicity is implicitly always “1”.

5.3.3. Example of the Visual Notation

Figure 5-2 shows an example of all notational elements (with the exception of fork nodes and merge nodes) used in a Hazard Relation Diagram.

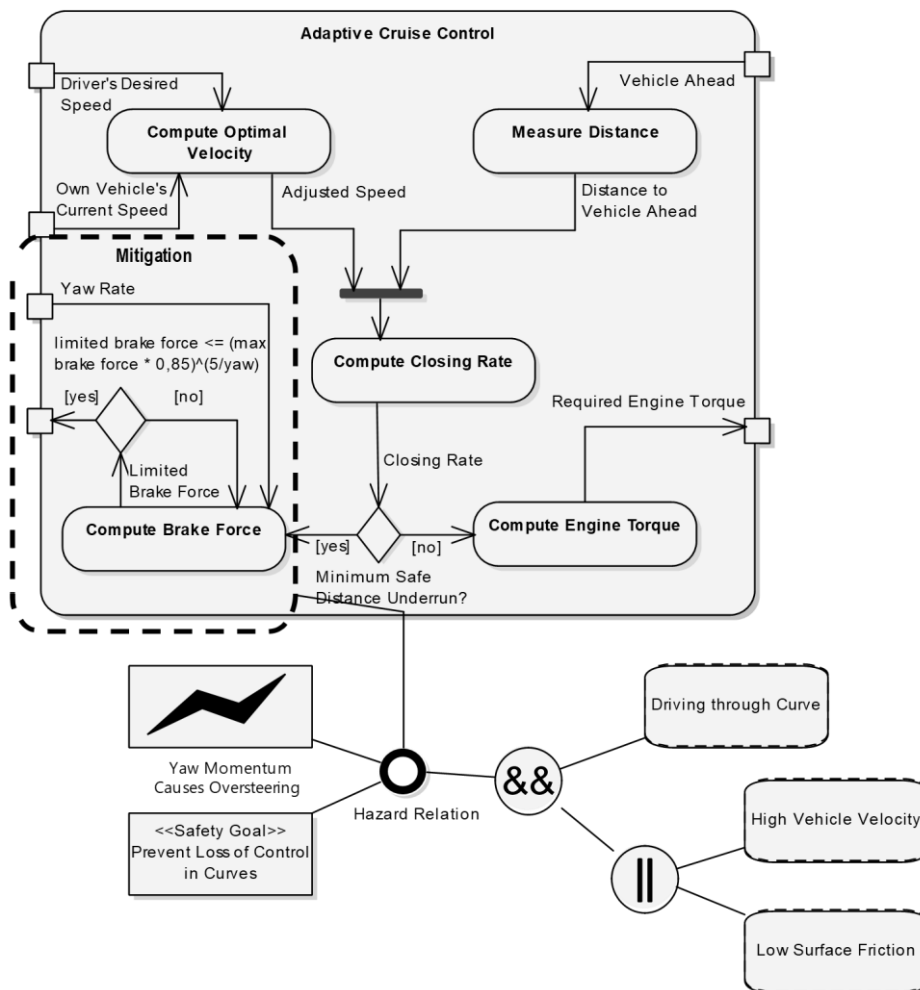


Figure 5-2 Example of a Hazard Relation Diagram featuring Hazard H1 from Table 2-1.

The Hazard Relation Diagram in this example features the ACC example from Section 2.1 and the hazard-mitigating requirements suggested in Section 2.1. As explained above, the hazard-inducing requirements of the adaptive cruise control system from Figure 2-1 have been modeled using the notational elements of conventional UML activity diagrams. In Section 2.1, the functional requirement “Compute Brake Force” has been determined to induce the hazard “Yaw

Momentum Causes Oversteering” (H1 in Table 2-1). This *hazard* is depicted using the lightning symbol from Table 5-1 and the associated *safety goal* “Prevent Loss of Control in Curves” (SG1 in Table 2-1) is depicted a UML class stereotyped <<Safety Goal>>. The tree trigger conditions from Table 2-1 have been documented as a binary tree using a Trigger Condition Conjunction as well as a Trigger Condition Disjunction. In Section 2.1, it was suggested to limit the brake force depending on the rate of yaw in order to mitigate hazard H1. This conceptual mitigation has been refined into one *partial mitigation* containing several *hazard-mitigating requirements* surrounded by the bold, dashed, rounded-edge *mitigation partition*. In Figure 5-2, the mitigation partition hence comprises the added input pin, which queries the car’s yaw rate sensor, and the added decision node, which checks if the brake force was appropriately limited.

5.4. Relationship Types between Hazards and Conceptual Mitigations

In Section 1.2, we have discussed the typical relationships between hazard-mitigating requirements and hazards, as outlined in [US DOD 2010] (see p. 101). These relationships are more complicated in Hazard Relation Diagrams. In the following subsections, these cases are summarized and applied on Hazard Relation Diagrams.

5.4.1. Case 1: One Hazard and One Conceptual Mitigation Documented in One Partial Mitigation

In this case, hazard-mitigating requirements which implement the conceptual mitigation were added locally within that activity diagram using a single mitigation partition. The example for the Hazard Relation Diagram from Figure 5-2 shows an example of this case. This case corresponds to the “One-to-One” multiplicity discussed in Section 1.2. However, as outlined in Section 1.1, the multiplicity between hazards and the conceptual mitigation is rarely 1:1, because diagrams depicting functional requirements are not “cut” according to potential hazards, but according to the level of detail needed for the specific development situation (see, e.g., [Conway 1967; Finkelstein et al. 1992]).

5.4.2. Case 2: One Hazard and One Conceptual Mitigation Documented in Several Partial Mitigations, Within the Same Activity Diagram

In these cases, several hazard-mitigating requirements that pertain to the same conceptual mitigation for exactly one hazard were added to different locations of one activity diagram. Albeit in principle, Hazard Relation Diagrams contain only one conceptual mitigation specific to one hazard (see Section 5.5), a conceptual mitigation in the sense of Definition 7 comprises several

hazard-mitigating requirements. This case hence corresponds to the “One-to-Many” multiplicity in Section 1.2. Other than the counterpoint that all these hazard-mitigating requirements specific to some hazard must necessarily be part of the same conceptual mitigation, there is no ontological constraint (see Figure 5-1, Well-Formedness Rule 12, and Well-Formedness Rule 15) that there is any other relationship between these hazard-mitigating requirements (e.g. that they must be documented in the same diagram, see Well-Formedness Rule 13). To allow for such non-trivial relationships between hazard-mitigating requirements, their conceptual mitigations, and hazards, the visual notation (see Section 5.3) of Hazard Relation Diagrams differentiates between the term “conceptual mitigation” according to Definition 7 and the term “partial mitigation” according to Definition 6. Moreover, in Section 5.3, the notational element “*mitigation partition*” was introduced to visually represent a partial mitigation. This allows associating multiple mitigation partitions (which comprise hazard-mitigating requirements pertaining to the same conceptual mitigation, see Well-Formedness Rule 15) to the Hazard Relation of a Hazard Relation Diagram. An example of such a case is shown in Section 5.4.8.

5.4.3. Case 3: One Hazard and One Conceptual Mitigation Documented in Several Partial Mitigations, Within the Several Activity Diagrams

In these cases, several hazard-mitigating requirements that pertain to the same conceptual mitigation for exactly one hazard were added to different locations of several activity diagrams. This case is an extension of Case 2 and hence also corresponds to the “One-to-Many” multiplicity in Section 1.2. The difference to Case 2 is that in this particular case, one or more mitigation partitions surround hazard-mitigating requirements, which have been added to different activity diagrams. This case also comprises combinations of Case 2 and Case 3. For example, in some Hazard Relation Diagram, three mitigation partitions exist. Two mitigation partitions have been added to one activity diagram, whilst the third mitigation partition has been added to a second activity diagram. The example in Section 5.4.8 shows exactly this combination case to illustrate Case 2 and Case 3 together.

5.4.4. Case 4: Several Hazards and One Conceptual Mitigation

In this case, several hazards exist that are addressed by the same conceptual mitigation. Since the purpose of Hazard Relation Diagrams is to visualize the dependencies of one conceptual mitigation and the hazard-mitigating requirements subsumed therein with respect to a specific hazard, the conceptual mitigation must be validated with respect to each hazard it is meant to

mitigate (see Well-Formedness Rule 14). Hence, for Case 4, the ability of the conceptual mitigation to mitigate a hazard must be validated in individual Hazard Relation Diagrams. In consequence, for every combination of conceptual mitigation and hazard, one Hazard Relation Diagram must be created and subjected to validation. This case hence corresponds to the “Many-to-One” multiplicity in Section 1.2.

5.4.5. Case 5: One or More Hazards and Several Conceptual Mitigations

In this case, several hazards exist that are addressed by several, alternative conceptual mitigations (possibly with varying degrees of adequacy). This case corresponds to the “Many-to-Many” multiplicity in Section 1.2. During development, combinations of Case 4 and 5 may occur frequently: For example, three hazards could be addressed by four conceptual mitigations, where one conceptual mitigation addresses two of the identified hazards and the other three conceptual mitigations address the last hazard. Like in Case 4, each combination of hazard and conceptual mitigation must be depicted in a dedicated Hazard Relation Diagram and subsequently subjected to validation. An example for this case is outlined in Section 2.1, where two conceptual mitigations were suggested for the same hazard. Hence, like illustrated in Section 2.1, it must be noted that Cases 5 is limited to conceptual mitigation alternatives, i.e. conceptual mitigations that each address the same hazard entirely, regardless of the respective other conceptual mitigations for other hazards. The purpose of validation in this case would be to assess not only adequacy of the conceptual mitigations, but also to assess *optimality* of conceptual mitigations, for instance to find a conceptual mitigation that is (1) adequate, (2) takes the least time to implement, or (3) is the most cost effective to implement. These are beyond the scope of this research.

5.4.6. Combination Cases

As outlined in Sections 5.4.3 and 5.4.5, combinations of the above cases are possible and likely quite common during development. For example, the case could exist, where multiple mutually dependent hazards, which require one or more mutually dependent conceptual mitigations exist. This might occur when one hazard “causes” another hazard. Albeit this might happen in practice, typically, Fault Tree Analysis is used to identify such interactions (i.e. to identify “minimal cut-sets” of hazards and their causes, see [Ericson 2005]). This is beyond the scope of this article. In principle, this can be addressed by integrating the top-most hazard from the

fault tree into the Hazard Relation Diagram, in satisfaction of Well-Formedness Rule 1. Mutually dependent conceptual mitigations can then be represented as several partial mitigations, similarly to Case 2.

5.4.7. Other Cases

In any other case, i.e. in cases which do not conform to Case 1 through Case 5, no syntactically valid Hazard Relation Diagram can be created, as in these cases, at least one of the well-formedness rules from Section 5.5 is violated. This might be the case, for example, if a Hazard Relation Diagram consists of several activity diagrams, is meant to contain a single conceptual mitigation for some hazard, but the Hazard Relation Diagram does not contain a mitigation partition. Such a case would be in violation of Well-Formedness Rule 13. For another example, if a Hazard Relation Diagram contains several mitigation partitions within the same activity diagram, but no hazard, this would violate Well-Formedness Rule 1.

5.4.8. Summary and Example of Non-Trivial Relationships in Hazard Relation Diagrams

Table 5-3 summarized the cases outlined in the previous subsections. An example for Case 2 and 3 is given in Figure 5-3. In this example, the conceptual mitigation aims at detecting the actual yaw rate by querying individual wheels and delimiting the brake force accordingly (i.e. Mitigation 2 suggested in Section 2.1). As can be seen in Figure 5-3, the conceptual mitigation consists of hazard-mitigating requirements, which were depicted using three mitigation partitions, where one belongs to the ACC and two belong to the electronic stability program (ESP). The hazard-mitigating requirements for the ACC comprise functionality to query the current yaw rate as well as the current yaw momentum and comprise functionality to compute the optimal brake force for each individual wheel (see Section 2.1). In addition, hazard-mitigating requirements have been included in the functional requirements of the ESP (Figure 5-3 shows an excerpt of a simplified ESP based on [Reif 2010]). Specifically, a function to compute the current yaw momentum has been introduced, which not only submits the current yaw momentum to the function *Compute Yaw Momentum Set Point*, but also submits the yaw momentum to the ACC. Furthermore, the ESP accepts input from the ACC regarding the necessary brake force for each wheel and the ESP double-checks the brake force distribution for each wheel before initiating the deceleration through the wheels' brake actuators.

Table 5-3 Types of Relationships between Hazards and Conceptual Mitigations.

Case	# Hazards	# Conceptual Mitigations	# Mitigation Partitions	# Activity Diagrams	Description	Impact on HRDs
1	1	1	1	1	Exactly one hazard is addressed by exactly one conceptual mitigation and comprised in exactly one mitigation partition. The conceptual mitigation comprises hazard-mitigating requirements which were added to exactly one activity diagram.	Default case shown in Figure 5-2.
2	1	1	2..*	1	Exactly one hazard is addressed by exactly one conceptual mitigation, but the conceptual mitigation comprises hazard-mitigating requirements which are scattered across geometrically distant areas within the same activity diagram.	Multiple partial mitigations are defined for the conceptual mitigation. For each partial mitigation, a mitigation partition is included in the Hazard Relation Diagram.
3	1	1	2..*	2..*	Exactly one hazard is addressed by exactly one conceptual mitigation, but the conceptual mitigation comprises hazard-mitigating requirements which are scattered across multiple activity diagrams.	Extension of Case 2: Multiple partial mitigations are defined for the conceptual mitigation. At least one partial mitigation is defined for each activity diagram. The partial mitigations contain hazard-mitigating requirements pertaining to the same conceptual mitigation. For each partial mitigation, the activity diagram and the corresponding mitigation partition is included in the Hazard Relation Diagram.
4	2..*	1	1..*	1..*	More than one hazard is addressed by the same conceptual mitigation, regardless whether the hazard-mitigating requirements are scattered across one or more activity diagrams.	The adequacy of the candidate conceptual mitigation must be validated with regard to each hazard, hence requiring one Hazard Relation Diagram for each hazard.
5	1	2..*	1..*	1..*	Multiple conceptual mitigations exist for the same hazard. Conceptual mitigations are independent from one another, i.e. comprise alternative hazard-mitigating requirements, which might be scattered across different or the same activity diagram.	Reverse case of Case 4: The adequacy of each candidate conceptual mitigation must be reviewed with regard to the same hazard. Just like in Case 4, this requires one Hazard Relation Diagram for each conceptual mitigation.

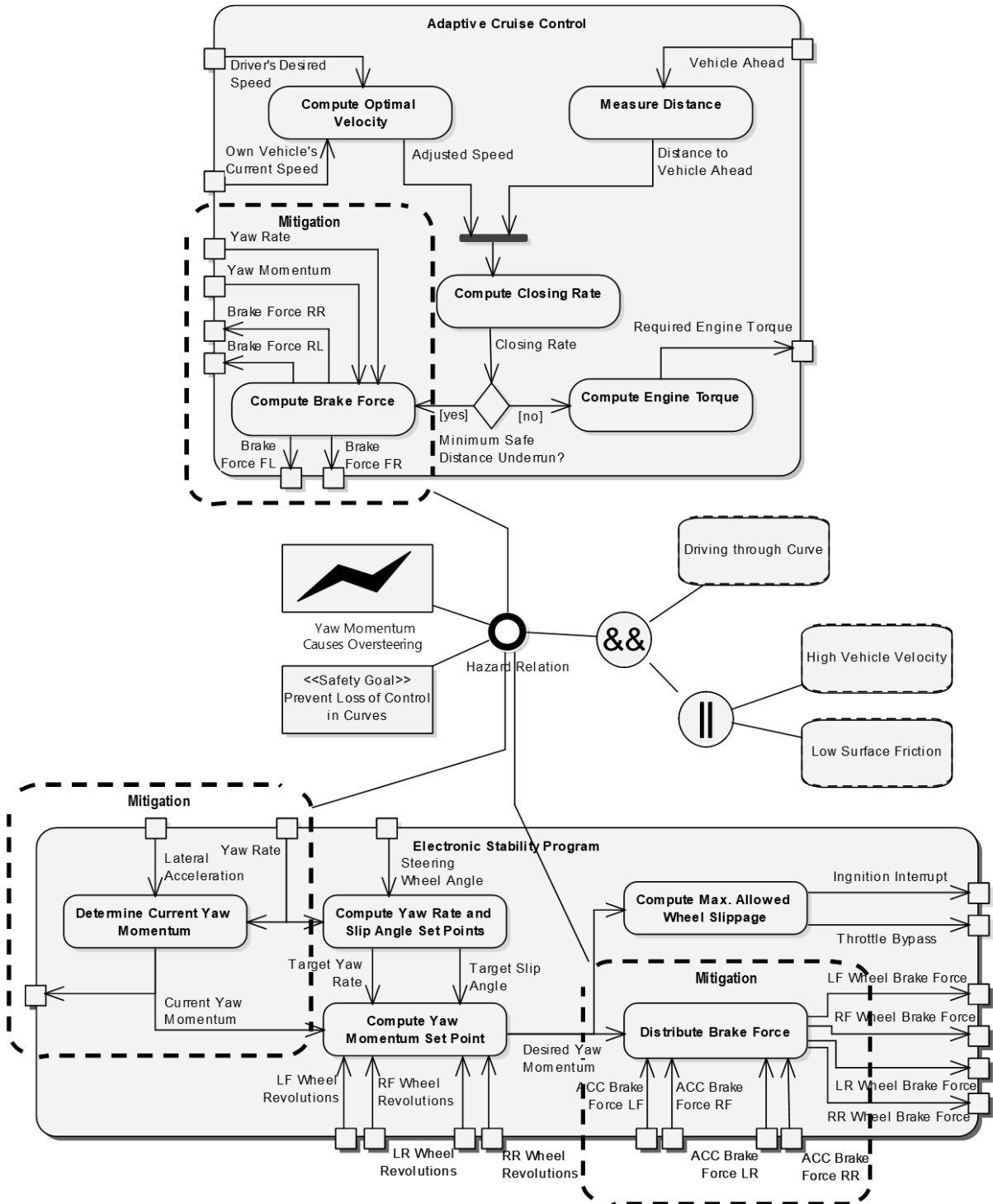


Figure 5-3 Example of a Hazard Relation Diagram featuring Multiple Mitigation Partitions.

5.5. Well-Formedness Rules

A number of well-formedness rules have been defined in order to check if Hazard Relation Diagrams are syntactically correct. The strength of Hazard Relation Diagrams lies in their ability to visualize the dependency between hazard-mitigating requirements with regard to their ability to mitigate one particular hazard. We define the following well-formedness rule:

Well-Formedness Rule 1:

A Hazard Relation Diagram contains exactly one hazard documented in hazard analysis worksheets.

However, since during certification, the safety of a system is argued by means of the adequate fulfillment of a safety goal (see Definition 3), which was put in place in response to a hazard. Therefore, that safety goal must be contained in a Hazard Relation Diagram:

Well-Formedness Rule 2:

A Hazard Relation Diagram contains exactly one safety goal documented in hazard analysis worksheets.

At least one hazard-mitigating requirement must be defined to fulfill the safety goal into a conceptual mitigation (see Section 2.1). However, missing hazard-mitigating requirements are likely to cause the fulfillment to be inadequate and hence the conceptual mitigation to be inadequate, which must be ascertained during validation. The adequacy of hazard-mitigating requirements depends on the adequate fulfillment of the safety goal that corresponds to the hazard, since it can only be judged whether or not the conceptual mitigation adequately fulfills the safety goal, if the safety goal and the hazard match:

Well-Formedness Rule 3:

The safety goal in a Hazard Relation Diagram is specific to the conceptual mitigation of the hazard depicted in the Hazard Relation Diagram.

In addition, whether or not a hazard still occurs during operation depends on the circumstance of in the operational context (see Definition 2). Therefore, validation must consider the hazard's specific trigger conditions:

Well-Formedness Rule 4:

A Hazard Relation Diagram contains the trigger condition identified during hazard analyses that are document the Boolean conditions from the operational context in which the hazard occurs.

In practice, however, trigger conditions are rarely Boolean atomic states, but a list of operational conditions (see [Ericson 2005], p. 276), which may or may not occur together in order to induce

a hazard. A trigger condition can hence be viewed as a tree of atomic states consisting of conjunctions and disjunctions in root nodes and atomic states in leaf nodes:

Well-Formedness Rule 5:

The trigger conditions are represented in a Hazard Relation Diagram in a tree structure.

In principle, n-ary tree structures may be permissible. However, for the purpose of creating Hazard Relation Diagrams (see Section 5.5), we restrict the tree structure to that of a binary tree, where the root of the binary trigger condition tree is either a Trigger Condition Conjunction or a Trigger Condition Disjunction (see Table 5-1). Since it is an inherent property of binary trees that there are at most two leaves, we define the following well-formedness rule:

Well-Formedness Rule 6:

The nodes of the binary trigger condition tree must be either a Trigger Condition Conjunction or a Trigger Condition Disjunction and there must be at most two leafs.

Since a trigger condition can in turn be caused by some other trigger condition in a cascaded fashion [Leveson 1995], a binary trigger condition tree might contain further subtrees:

Well-Formedness Rule 7:

The leafs of the binary trigger condition tree must be atomic states from the operational context of the system under development identified during hazard analyses or the root for a subtree.

During validation, it is assessed whether the hazard-mitigating requirements, which are special types of functional requirements (see Definition 5). In Section 5.1, it was outlined that hazard-mitigating requirements in Hazard Relation Diagrams shall be documented using the same notational elements as conventional UML activity diagrams. We therefore define the following well-formedness rule:

Well-Formedness Rule 8:

The hazard-mitigating requirements depicted in a Hazard Relation Diagram are documented using the notational elements of UML activity diagrams.

The hazard-mitigating requirements in a Hazard Relation Diagram must be syntactically correct, as syntactic correctness can be seen as a prerequisite for semantic validity (see Montague’s View of the role of syntax as described in [Heim and Kratzer 1998]). We hence define the following well-formedness rules.

Well-Formedness Rule 9:

There are no unconnected (“dangling”) activity edges in a Hazard Relation Diagram.

Well-Formedness Rule 10:

There are no unconnected (“orphaned”) activities, control nodes, or pins in a Hazard Relation Diagram.

Well-Formedness Rule 11:

There are no cliques of circularly connected, but otherwise unconnected activities and control nodes in a Hazard Relation Diagram.

In order to be able to assess whether the hazard is adequately mitigated, the hazard-mitigating requirements for this particular hazard must be contained in the Hazard Relation Diagram. According to Definition 6, these hazard-mitigating requirements must hence pertain to the same conceptual mitigation. Since the conceptual mitigation represents merely one possible way to render the hazard sufficiently improbable to occur during operation, there may be multiple candidate conceptual mitigations which address the same hazard (see Section 5.4). Furthermore, as indicated above, the conceptual mitigation may be scattered across the underlying activity diagram (a more detailed discussion of the relationships between hazard-mitigating requirements and hazards can be found in Section 5.4). In Section 5.3, we have introduced the modeling element of mitigation partitions for this purpose. Therefore, the following well-formedness rules follow:

Well-Formedness Rule 12:

A Hazard Relation Diagrams contains exactly one conceptual mitigation documented in at least one mitigation partition.

Well-Formedness Rule 13:

A Hazard Relation Diagrams contains all mitigation partitions pertaining to the same conceptual mitigation.

Well-Formedness Rule 14:

The conceptual mitigation depicted in a Hazard Relation Diagram is specific to the hazard depicted in the Hazard Relation Diagram.

Well-Formedness Rule 15:

The mitigation partitions depicted in a Hazard Relation Diagram subsume all hazard-mitigating requirements specific to the hazard depicted in the Hazard Relation Diagram.

The most central concept in Hazard Relation Diagrams is the Hazard Relation. It serves as a graphical representation of the dependencies between the hazard (see Well-Formedness Rule 1), safety goal (see Well-Formedness Rule 3), the trigger conditions (see Well-Formedness Rule 4), and the hazard-mitigating requirements (see Well-Formedness Rule 15). In consequence, there can only be one Hazard Relation in a Hazard Relation Diagram that connects hazard, safety goal, trigger conditions, and mitigation partitions, as otherwise the dependencies would be ambiguous:

Well-Formedness Rule 16:

A Hazard Relation Diagram contains exactly one Hazard Relation.

Well-Formedness Rule 17:

The Hazard Relation contained in a Hazard Relation Diagram is associated with the hazard, the safety goal, the top-most element of the binary trigger condition tree, and all mitigation partitions depicted in the Hazard Relation Diagram.

6. Creation Approach for Hazard Relation Diagrams

In this section, we outline the creation of Hazard Relation Diagrams. Section 6.1 gives an overview over the approach. Section 6.2 discusses the canonical artifact type formalizations that the approach is based on. Section 6.3 discusses the creation of Hazard Relation Diagrams based on the artifact type formalizations.

6.1. Overview

The approach to create Hazard Relation Diagrams makes use of the artifacts and their ontological dependencies shown in Figure 4-1. In order to create Hazard Relation Diagrams, it is therefore necessary to first canonically formalize the artifacts as well as their dependencies before a two-step process can be applied to create Hazard Relation Diagrams.

6.1.1. Artifacts

The following artifacts must hence be formalized:

- 1. Functional Requirements.** Functional requirements are UML activity diagram modeling elements, which are used to depict hazard-mitigating, hazard-inducing requirements, as well as other functional requirements with no impact on safety (but which are necessary to describe the system's functionality). This artifact type is formalized in Section 6.2.1.
- 2. Hazard Analysis Results.** Hazard Analysis Results comprise the identified hazards, the hazards' respective trigger conditions, and conceived safety goal. This artifact type is formalized in Section 6.2.2.
- 3. Partial Mitigations.** In order to formalize the dependency types depicted in Figure 4-1, this dissertation proposes a mitigation template, which allows stakeholders to document the dependencies between hazard-mitigating requirements, hazards, trigger conditions, and safety goals. The purpose of the mitigation template is therefore not only define the dependencies for a Hazard Relation Diagram, but also the hazard-mitigating requirements depicted therein. Hence, a mitigation template is defined for each partial mitigation to be contained in the Hazard Relation Diagram. This artifact type is formalized in Section 6.2.3.
- 4. Hazard Relation Diagrams.** This is the artifact type to be created. It is formalized in Section 6.2.4.

6.1.2. Creation Steps

Mitigation templates therefore facilitate the creation process of Hazard Relation Diagrams. They contain the transformation steps necessary to obtain hazard-mitigating requirements. Moreover, they contain the dependencies to the hazard, trigger conditions, and the safety goal. Creation of Hazard Relation Diagrams can hence be enacted using two steps:

1. **Create Hazard-Mitigating Requirements.** Functional requirements are changed by enacting the transformation steps to add, remove, or substitute model elements from an UML activity diagram. This is done for each mitigation template that pertains to the same conceptual mitigation (see Case 1, 2, and 3 in Section 5.4). The result of this step is a set of UML activity diagrams, which have been changed to contain hazard-mitigating requirements. This step is described in Section 6.3.1.
2. **Create Hazard Relation Diagrams.** The UML activity diagrams resulting from Step 1 are merged into one Hazard Relation Diagram and Hazard Analysis Results are appended. This is done based on the dependencies specified in the corresponding mitigation templates. This step is described in Section 6.3.2

Figure 6-1 gives an overview over the creation process.

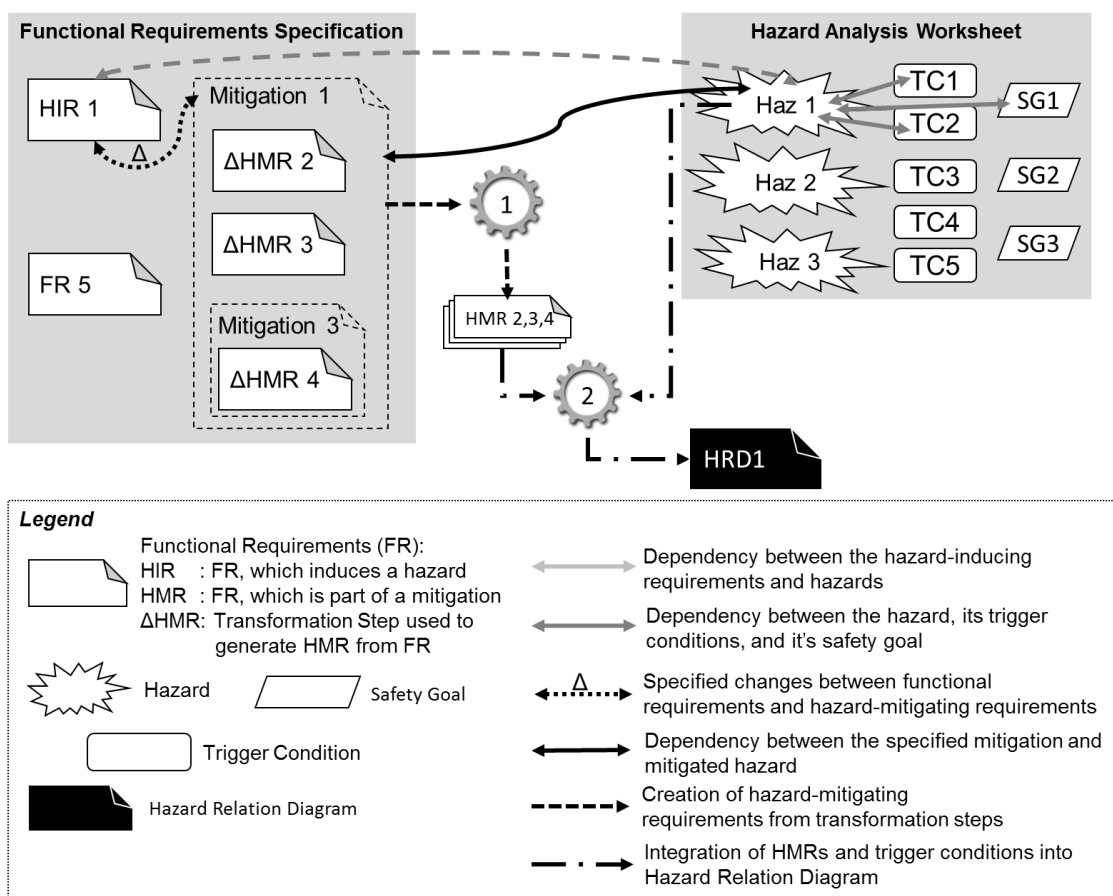


Figure 6-1 Exemplary Overview over the Creation of Hazard Relation Diagrams.

In Figure 6-1, one mitigation template, i.e. *Mitigation 1*, specifies the conceptual mitigation for hazard *Haz 1*, consisting of the transformation steps $\Delta HMR 2$, $\Delta HMR 3$, and $\Delta HMR 4$. In Step 1, hazard-mitigating requirements *HMR 2*, *HMR 3*, and *HMR 5* are created (cogwheel labeled “1” in Figure 6-1). In Step 2, following the dependency to the hazard analysis worksheet from *Mitigation 1*, hazard *Haz 1*, its trigger conditions *TC1* and *TC2*, as well as its safety goal *SG1* are appended (cogwheel labeled “2” in Figure 6-1). The Hazard Relation Diagram therefore contains the hazard-mitigating requirements specified in *Mitigation 1* as well as the hazard *Mitigation 1* is meant to mitigate, along with that hazard’s trigger conditions.

6.1.3. Well-Formedness Rule Satisfaction

Using this process, the well-formedness rules from Section 5.5 can be satisfied. Their satisfaction is summarized in Section 6.4.

6.2. Canonical Artifact Type Formalization

There is a number of artifacts which are either necessary inputs to create Hazard Relation Diagrams or are intermediate results of the approach. In the following, we provide formalizations of these artifacts’ canonical representations in order to present an approach to create Hazard Relation Diagrams in Section 6.3.

6.2.1. Functional Requirements

In the following, we simplify the UML activity diagrams proposed in [OMG 2015b] to incorporate only those concepts needed to express system functionality that is input for hazard analysis worksheets (i.e. activities and actions, see [Ericson 2005]). Since these are equivalent with regard to their consideration in hazard analyses, we use the term “activity” to describe opaque actions, activities, events, etc. In addition, it is to note that in contrast to semantic formalization of UML activity diagrams, e.g., as proposed in [Eshuis and Wieringa 2001], the following formalism represents a purely canonical form of simplified UML activity diagram representations.

Formalization

For this work, every activity diagram $actD$ can be described as a tuple

$$actD = (ad, A^{ad}, E^{ad}, P^{ad}, C^{ad}) \quad (6.1)$$

where ad is a unique, user-defined name of the activity diagram, $A^{ad} = \{a_1^{ad}, a_2^{ad}, \dots, a_n^{ad}\}$ is a set of activities in that activity diagram, $E^{ad} = \{e_1^{ad}, e_2^{ad}, \dots, e_n^{ad}\}$ is a set of activity edges,

$P^{ad} = \{p_1^{ad}, p_2^{ad}, \dots, p_n^{ad}\}$ is a set of input/output ports, and $C^{ad} = F^{ad} \cup J^{ad} \cup D^{ad} \cup X^{ad}$ is a set of control nodes, which consists of a set of fork nodes $F^{ad} = \{f_1^{ad}, f_2^{ad}, \dots, f_n^{ad}\}$, a set of join nodes $J^{ad} = \{j_1^{ad}, j_2^{ad}, \dots, j_n^{ad}\}$, a set of decision nodes $D^{ad} = \{d_1^{ad}, d_2^{ad}, \dots, d_n^{ad}\}$, and a set of merge nodes $X^{ad} = \{x_1^{ad}, x_2^{ad}, \dots, x_n^{ad}\}$. An activity edge can be described as a tuple

$$e = (src^e, m^e, tar^e) | src^e, tar^e \in (A^{ad} \cup P^{ad} \cup C^{ad}) \quad (6.2)$$

where the source src^e and the target tar^e is an activity, a port, or a control node, respectively, with a message m^e being transmitted over the activity edge. An activity edge is called a control flow if $m^e = \varepsilon$ and a data flow if $m^e \neq \varepsilon$. A port p is called an input port if it is the source of a data flow e , i.e.

$$\begin{aligned} P_{in}^{ad} &\subseteq P^{ad} | \forall p \in P^{ad}: \exists e = (src^e, m, tar^e) \in E^{ad} \\ &\wedge p = src^e \wedge m \neq \varepsilon \end{aligned} \quad (6.3)$$

A port p is called an output port if it is the target of a data flow e , i.e.

$$\begin{aligned} P_{out}^{ad} &\subseteq P^{ad} | \forall p \in P^{ad}: \exists e = (src^e, m, tar^e) \in E^{ad} \\ &\wedge p = tar^e \wedge m \neq \varepsilon \end{aligned} \quad (6.4)$$

To further simplify the following formalizations, it is assumed that Activities are specified without ActivityParameterNodes and that function parametrization is achieved through the message on an activity edge.

A fork node $f \in F^{ad}$ is a control node that has one incoming activity edge and at least two outgoing activity edges which carry the same message, i.e.

$$\begin{aligned} \forall f \in F^{ad}: \exists e^{in} = (src^{in}, m^{in}, tar^{in}) \in E^{ad} \wedge tar^{in} = f \\ \wedge \exists E' = \{e_1, e_2, \dots, e_n\} \subseteq E^{ad} \wedge n \geq 2 \\ \wedge \forall e = (src^{out}, m^{out}, tar^{out}) \in E': src^{out} = f \wedge m^{in} = m^{out} \end{aligned} \quad (6.5)$$

A join node is a control node with multiple incoming edges and one outgoing edge:

$$\begin{aligned} \forall j \in J^{ad}: \exists E' = \{e_1, e_2, \dots, e_n\} \subseteq E^{ad} \wedge n \geq 2 \\ \wedge \forall e = (src^{in}, m^{in}, tar^{in}) \in E' \wedge tar^{in} = j \\ \wedge \exists e^{out} = (src^{out}, m^{out}, tar^{out}) \in E^{ad} \wedge src^{out} = j \end{aligned} \quad (6.6)$$

A decision node $d \in D^{ad}$ is a control node that has one incoming activity edge and at least two outgoing activity edges carrying the same message, and where each outgoing edge has a different guard:

$$\begin{aligned}
 \forall d \in D^{ad}: \exists e^{in} = (src^{in}, m^{in}, g^{in}, tar^{in}) \in E^{ad} \\
 \wedge tar^{in} = d \wedge \exists E' = \{e_1, e_2, \dots, e_n\} \subseteq E^{ad} \wedge n \geq 2 \\
 \wedge \forall e = (src^{out}, m^{out}, g^{out}, tar^{out}) \in E' \wedge src^{out} = d \\
 \wedge g_{n-1}^{out} \neq g_n^{out} \wedge m^{in} = m_n^{out}
 \end{aligned} \tag{6.7}$$

In (6.2), we defined activity edges as a 3-tuple without a guard, as the guard plays no role for the remainder of the formalization. However, in order to differentiate decision nodes from fork nodes, it is necessary to specify activity edges as a 4-tuple, as shown in (6.7).

A merge node is a control node merging multiple incoming edges after a decision node into one outgoing activity edge, i.e.

$$\begin{aligned}
 \forall x \in X^{ad}: \exists E' = \{e_1, e_2, \dots, e_n\} \subseteq E^{ad} \wedge n \geq 2 \\
 \wedge \forall e = (src^{in}, m^{in}, tar^{in}) \in E' \wedge tar^{in} = x \\
 \wedge \exists e^{out} = (src^{out}, m^{out}, tar^{out}) \in E^{ad} \wedge src^{out} = x
 \end{aligned} \tag{6.8}$$

Example

The activity diagram in Figure 2-1 in Section 2.1 is repeated in the following Figure 6-2 shows the excerpt of the functional requirements of the Adaptive Cruise Control.

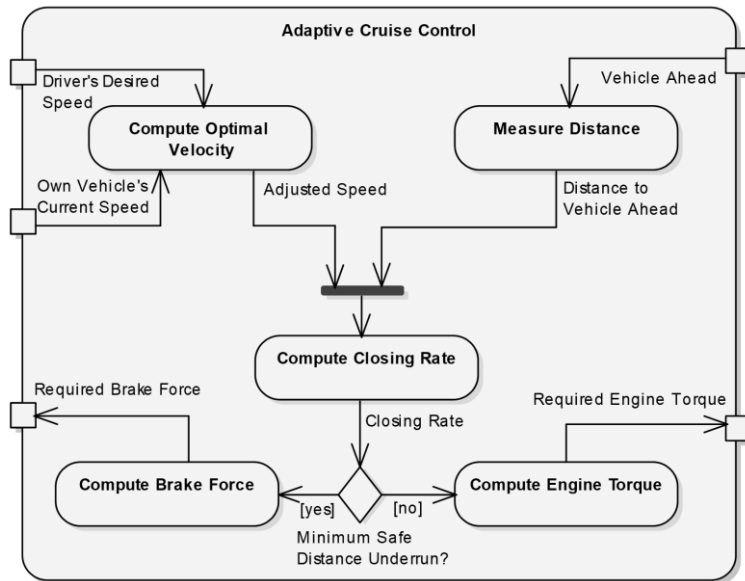


Figure 6-2 Excerpt of the Functional Requirements of the Example ACC from Figure 2-1.

The activity diagram in Figure 6-2 can be written as:

$$actD^{Adaptive\ Cruise\ Control} = \left(\begin{array}{l} Adaptive\ Cruise\ Control, \\ A^{Adaptive\ Cruise\ Control}, \\ E^{Adaptive\ Cruise\ Control}, \\ P^{Adaptive\ Cruise\ Control}, \\ C^{Adaptive\ Cruise\ Control} \end{array} \right) \quad (6.9)$$

In order to increase readability of the example equations hereinafter, we introduce an abbreviated notation of the equations. In the following, the superscript indicating the name of an entity shall be left out if the references name is the name of the entity defined in the tuple. Therefore, (6.9) can be written in the following abbreviated form:

$$actD = (Adaptive\ Cruise\ Control, A, E, P, C) \quad (6.10)$$

The name of this activity diagram can be any unique ID or human-readable designation, e.g.:

$$ad = Adaptive\ Cruise\ Control \quad (6.11)$$

As can be seen, the activity diagram contains five activities, three input ports, two output ports, and two control nodes, one decision node and one fork node. In order to create Hazard Relation Diagrams, it is necessary to provide a unique name to control nodes, although UML does not prescribe control nodes to have an identity. In the following, we use anonymous unique names for ports, fork nodes, join nodes, and merge nodes as well as the human-readable decision.

$$A^{Adaptive\ Cruise\ Control} = \left\{ \begin{array}{l} Compute\ Optimal\ Velocity, \\ Measure\ Distance, \\ Compute\ Closing\ Rate, \\ Compute\ Brake\ Force, \\ Compute\ Engine\ Torwque \end{array} \right\} \quad (6.12)$$

$$P^{Adaptive\ Cruise\ Control} = \{i_1, i_2, i_3, o_1, o_2\} \quad (6.13)$$

$$D^{Adaptive\ Cruise\ Control} = \{Minimum\ Safe\ Distance\ Underrun?\} \quad (6.14)$$

$$F^{Adaptive\ Cruise\ Control} = \{f_1\} \quad (6.15)$$

$$\begin{aligned} C^{Adaptive\ Cruise\ Control} \\ = F^{Adaptive\ Cruise\ Control} \cup D^{Adaptive\ Cruise\ Control} \end{aligned} \quad (6.16)$$

Activity edges describe data flows and control flows between activity, ports, and control nodes. The activity diagram in Figure 2-1 contains the following eleven activity edges:

$$E^{Adaptive\ Cruise\ Control} = \quad (6.17)$$

$$\left(\begin{array}{l} (i_1, Driver's\ Desired\ Speed,)', (i_2, Own\ Vehicle's\ Current\ Speed,)', \\ (Compute\ Optimal\ Velocity,)', (Compute\ Optimal\ Velocity,)', \\ (Compute\ Optimal\ Velocity,)', (i_3, Vehicle\ Ahead, Measure\ Distance,)', \\ Adjusted\ Speed, f_1 \\ (Measure\ Distance,)', (f_1, Adjusted\ Speed + \\ (Distance\ to\ Vehicle\ Ahead,)', \\ (Distance\ to\ Vehicle\ Ahead,)', \\ Compute\ Closing\ Rate \\ (Compute\ Closing\ Rate, Closing\ Rate,)', (Minimum\ Safe\ Distance\ Underrun?,)', \\ (Minimum\ Safe\ Distance\ Underrun?,)', (Closing\ Rate, Compute\ Engine\ Torque)', \\ (Compute\ Engine\ Torque,)', (Minimum\ Safe\ Distance\ Underrun?,)', \\ (Required\ Engine\ Torque, o_1)', (Closing\ Rate, Compute\ Brake\ Force)', \\ (Compute\ Brake\ Force,)', \\ (Required\ Brake\ Force, o_2) \end{array} \right)$$

6.2.2. Hazard Analysis Results

The term “hazard analysis result” has thus far been used as an umbrella term which describes the output of some type of hazard analysis, documented in a hazard analysis worksheet (see Assumption 6 in Section 1.3). A “hazard analysis result” yields potential hazards which may be induced by hazard-inducing requirements, the trigger conditions for each hazard, and a possible safety goal which has been defined in response to the hazard. Hazard Relation Diagrams extend the functional requirements of the system (see Section 6.2.1) by the information elicited during hazard analysis.

Formalization

A hazard analysis result can hence be described as a function on some activity diagram which yields a hazard list, i.e.

$$fha(actD) = H \quad (6.18)$$

with $actD$ being the activity diagram according to (6.1) and H being the resulting hazard list consisting of a number of hazards, $H = \{h_1, h_2, \dots, h_n\}$. A hazard can be described as a tuple:

$$h = (a, tc, sg) \quad (6.19)$$

with a unique hazard name h , which has an associated tuple consisting of an activity $a \in A^{actD}$ that gives rise to the hazard, a trigger condition tc , and the safety goal sg that has been conceived in response to the hazard.

The trigger conditions pertaining to a hazard are described as a binary tree, in which, like in the example in [Lehman et al. 2016], nodes AND- or OR-nodes associate exactly two binary trigger condition subtrees:

$$tc = (tc_{left}, r, tc_{right}): r \in \{c, conj, disj\} \quad (6.20)$$

$$\forall (r = c) \rightarrow tc_{left}, tc_{right} = \emptyset$$

In (6.20), tc_{left}, tc_{right} represent the left and right binary trigger condition subtree, respectively, which are associated through the root r . The root may be a conjunction (i.e. AND-node) $conj$, indicating that both the left and right subtree must evaluate to true for the entire tree tc to evaluate to true. Alternatively, the root may be a disjunction (i.e. OR-node) $disj$, in which either or both subtrees must evaluate to true for the entire tree tc to evaluate to true. If the root is neither a conjunction nor a disjunction, the root may be is a human-definable condition in the operational context of the system c , in which case it is implied that there are not further subtrees (i.e. $tc_{left}, tc_{right} = \emptyset$).

Example

In Table 6-1, an excerpt of the Functional Hazard Analysis from Table 2-1 is shown, featuring only hazard H1 and both of its associated safety goals and its trigger conditions.

Table 6-1 Excerpt of the Functional Hazard Analysis from Table 2-1 featuring Hazard H1.

System Hazard-Inducing Requirement	Adaptive Cruise Control Hazard		Functional Hazard Analysis			
	ID	Description	Effect	Trigger Condition	ID	Description
Compute Brake Force	H1	Yaw Momentum Causes Oversteering	Loss of Control, caus- ing Crash	Driving through Curve and (High Vehicle Veloc- ity or Low Road Surface Friction)	SG1	Prevent Loss of Control in Curves
					SG3	Adjust Brake Force In-de- pendently for All Four Wheels

In this example, n refers to the description of the hazard in Table 6-1, i.e. *Yaw Momentum Causes Oversteering*. Since there are two safety goals associated with hazard H1, sg could refer the description of either, i.e. *Prevent Loss of Control in Curves* (SG1 in Table 6-1) or *Adjust Brake Force Independently for All Four Wheels* (SG2 in Table 6-1).

The trigger condition from Table 6-1 is shown in Figure 6-3.

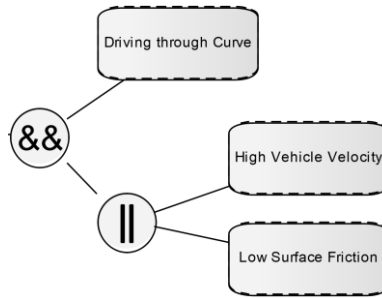


Figure 6-3 Trigger Condition of Hazard H1 from Table 6-1 as a Binary Tree.

In this example, the conditions *Driving through Curve*, *High Vehicle Velocity*, as well as *Low Road Surface Friction* have been defined. There is one disjunction associating *High Vehicle Velocity* with *Low Road Surface Friction*, thereby forming a subtree. This subtree is conjoined with the condition *Driving through Curve*. This binary trigger condition tree can be formalized as follows:

$$tc^{h_1} = \left(\begin{array}{l} conj, Driving\ through\ Curve, \\ (High\ Vehicle\ Velocity, disj), \\ (Low\ Road\ Surface\ Friction) \end{array} \right) \quad (6.21)$$

The remainder of the hazard analysis result table in Table 2-1 in Section 2.1 for the activity *Compute Optimal Velocity* from Figure 2-1, can hence be written as follows:

$$fha(Adaptive\ Cruise\ Control) = \{h_1, h_2, h_3\} \quad (6.22)$$

$h_1^{Yaw\ Momentum\ Causes\ Oversteer}$

$$= \left(\begin{array}{l} Compute\ Brake\ Force, \\ (Driving\ through\ Curve, conj), \\ (High\ Vehicle\ Velocity, disj), \\ (Low\ Road\ Surface\ Friction) \end{array} \right), \quad (6.23)$$

Prevent Loss of Control in Curve

$h_2^{Understeering\ due\ to\ Skidding}$

$$= \left(\begin{array}{l} Compute\ Brake\ Force, \\ (Driving\ through\ Curve, disj), \\ (Low\ Road\ Surface\ Friction) \end{array} \right), \quad (6.24)$$

Use Anti – Lock Brakes to Prevent Wheel Skid

$h_3^{Brake\ Force\ too\ low\ due\ to\ Skidding}$

$$= \left(\begin{array}{l} Compute\ Brake\ Force, Low\ Surface\ Friction, \\ Adjust\ Brake\ Force\ Independently\ for\ all\ Four\ Wheels \end{array} \right) \quad (6.25)$$

6.2.3. Partial Mitigations

Partial mitigations in the sense of Definition 6 subsume the hazard-mitigating requirements that are meant to mitigate a hazard. We have there introduced the modeling concept “mitigation partition” in Section 5.3 as the notational element to highlight hazard-mitigating requirements within a Hazard Relation Diagram (see Table 5-1 in Section 5.3). In Section 6.1, we introduced the concept “mitigation template” to document the ontological dependencies between hazards, trigger conditions, safety goals, and hazard-mitigating requirements from Section 5.1. Mitigation templates are used to document transformation steps to be enacted on activity diagrams containing hazard-inducing requirements in order to create activity diagrams containing hazard-mitigating requirements with are enriched with hazard analysis results in order to create Hazard Relation Diagrams (see Section 6.1). A mitigation template hence represents a mitigation partition in the created Hazard Relation Diagram. Table 6-2 shows a mitigation template that serves this purpose.

Table 6-2 Mitigation Template to Specify Changes to Activity Diagrams Containing Functional Requirements in order to Create Hazard-Mitigating Requirements.

Mitigation Name	Conceptual Mitigation name									
Reference	Source Activity Diagram name									
Hazard	Hazard name									
Insert Activity	ID	Activity Name								
Insert Activity Edge	ID	Source	Message			Guard	Target			
Insert Pin	ID	Pin Name								
Insert Control Node	ID	Node Type			Node Name					
Remove Activity	ID	Activity Name								
Remove Activity Edge	ID	Source	Message			Guard	Target			
Remove Pin	ID	Pin Name								
Remove Control Node	ID	Node Name								
Substitute Activity	ID	Old Activity Name					New Activity Name			
Substitute Activity Edge	ID	Old Source	Old Message	Old Guard	Old Target	New Source	New Message	New Guard	New Target	
Substitute Pin	ID	Old Pin Name					New Pin Name			
Substitute Control Node	ID	Old Node Name					New Node Type	New Node Name		

In Section 5.4, we outlined that in practice, it might be the case that a hazard is mitigated by introducing hazard-mitigating requirements to different locations within the same activity diagram, across multiple activity diagrams, or both. This means that multiple mitigation templates

may be needed in order to visualize the hazard’s conceptual mitigation in the created Hazard Relation Diagram, i.e. one mitigation template for each mitigation partition.

Formalization

It follows, that the modeling concept “conceptual mitigation” from Section 5.1 is a collective set of partial mitigations, each of which represents a set of removed, inserted, or substituted activity diagram modeling elements:

$$CM = \{pm_1, pm_2, \dots, pm_n\} \quad (6.26)$$

Each partial mitigation pm is a tuple consisting of a human-readable name mn identifying the common conceptual mitigation in the sense of Definition 7. This means that all partial mitigations with the same name make up the conceptual mitigation. In addition, the partial mitigation pm consists of a source activity diagram $actD$ according to (6.1) upon which the partial mitigation shall be committed, and a hazard h according to (6.19) that shall be mitigated. A partial mitigation furthermore consists of a set of elements removed, inserted, or substituted, respectively. For some partial mitigation pm , it follows:

$$pm = (mn, actD, h, R, I, S) \quad (6.27)$$

Any type of element el of the referenced activity diagram can be inserted, removed, or substituted (i.e. $el \in A^{actD} \cup E^{actD} \cup P^{actD} \cup C^{actD}$). To simplify the following formalizations, substitutions are expressed as a remove operation followed by an insertion operation. Hence, elements to be removed, inserted, or substituted can be written as

$$R = \{el_1, el_2, \dots, el_n\} \quad (6.28)$$

$$I = \{el_1, el_2, \dots, el_n\} \quad (6.29)$$

$$S = \{(el_1^{old}, el_1^{new}), (el_2^{old}, el_2^{new}), \dots, (el_n^{old}, el_n^{new})\} \quad (6.30)$$

It must also be noted, that insertion, removal, and substitution operations may result in syntactically invalid activity diagrams. For example, if an activity is removed, any activity edge that had the removed activity as a source or target will be “dangling” in the sense of Well-formedness Rule 9. For this reason, we have defined several well-formedness rules in Section 5.5 pertaining to the syntactic correctness of the activity diagrams underlying Hazard Relation Diagrams. However, these well-formedness rules only pertain to syntactic correctness as it is rele-

vant for the creation of Hazard Relation Diagram and do not cover any case of syntactic incorrectness of activity diagrams. It therefore remains the responsibility of the developer to ensure that that hazard-mitigating requirements are specified in a syntactically correct manner.

In case there are two or more partial mitigation with the same name mn , in which the same hazard h and the same activity diagram $actD$ are referenced, but where the elements to be inserted, removed, or substituted differ, the following holds:

$$\begin{aligned} \forall(pm_i, pm_j) | pm_i &= (mn_i, h_i, actD_i, R_i, I_i, S_i) \wedge \\ pm_j &= (mn_j, h_j, actD_j, R_j, I_j, S_j) \wedge mn_i = mn_j \wedge \\ h_i &= h_j \wedge actD_i = actD_j \wedge (R_i \cup I_i \cup S_i) \cap (R_j \cup I_j \cup S_j) = \emptyset \end{aligned} \quad (6.31)$$

In this case, all partial mitigations belong to the same conceptual mitigation and must be enacted on the activity diagram $actD$. This corresponds to Case 2 from Table 5-3. If there are two or more partial mitigations with the same name mn , in which different hazards are referenced, regardless of the activity diagrams referenced therein, this represents an example of Case 4 from Table 5-3. This means the same conceptual mitigation is a candidate to mitigate multiple, independent hazards. In this case, the changes from all partial mitigations belong to the same conceptual mitigation, but must be enacted on each respective activity diagram for the conceptual mitigation to be completely enacted. This corresponds to Case 3 from Table 5-3.

If there are two or more partial mitigations with the same name mn , in which different hazards are referenced, regardless of the activity diagrams referenced therein, the following must hold:

$$\begin{aligned} \forall(pm_i, pm_j) | pm_i &= (mn_i, h_i, actD_i, R_i, I_i, S_i) \wedge \\ pm_j &= (mn_j, h_j, actD_j, R_j, I_j, S_j) \wedge mn_i = mn_j \wedge h_i \neq h_j \end{aligned} \quad (6.32)$$

This case corresponds to Case 4 from Table 5-3 and is indicative of the same conceptual mitigation being a candidate to resolve multiple, independent hazards.

In case two or more partial mitigations have different names, they pertain to two or more conceptual mitigations (see above). If in all of these partial mitigation, the same hazard h is referenced, regardless of the activity diagrams referenced therein, the partial mitigations represent an example of Case 5 from Table 5-3, where alternative conceptual mitigations to resolve the same hazard, for which the following holds (see also Well-Formedness Rule 12):

$$\begin{aligned} \forall(pm_i, pm_j) | pm_i = (mn_i, h_i, actD_i, R_i, I_i, S_i) \wedge \\ pm_j = (mn_j, h_j, actD_j, R_j, I_j, S_j) \wedge mn_i \neq mn_j \wedge h_i = h_j \end{aligned} \quad (6.33)$$

Partial mitigation can be documented using mitigation templates. Mitigation templates document the specific changes that are enacted on the source activity diagram containing the hazard-inducing requirements in order to create a new activity diagram containing the hazard-mitigating requirements.

Example

The example in Table 6-3 shows the partial mitigation for the hazard-mitigating requirements in Figure 5-2 for Hazard H1 from Table 2-1.

Table 6-3 Example of a Specified Mitigation Template to Mitigate Hazard H1 from Table 2-1 according to Mitigation 1 from Section 2.1.

Mitigation	Mitigation for Hazard H1				
Reference	ACC Functional Requirements				
Corresponding Hazard	Yaw Momentum Causes Oversteering				
Insert Activity Edge	ID	Source	Message	Guard	Target
	4	i4	Yaw Rate	true	Compute Brake Force
	5	Compute Brake Force	Limited Brake Force	true	$\max brake\ force * 0.85^{\frac{1}{yaw}}$
	6	$\max brake\ force * 0.85^{\frac{5}{yaw}}$	Limited Brake Force	yes	o2
Insert Pin	ID	Pin Name			
	3	i4			
Insert Control Node	ID	Node Type	Node name		
	2	Decision	$\max brake\ force * 0.85^{\frac{5}{yaw}}$		
Remove Activity Edge	ID	Source	Message	Guard	Target
	1	Compute Brake Force	Required Brake Force	true	o2

As can be seen, the partial mitigation in in Table 6-3 bears the human-readable name “Mitigation for Hazard H1.” Furthermore, the partial mitigation references the activity diagram name given in (6.11) and the hazard from (6.23). Therefore:

$$mn = \text{Mitigation for Hazard H1} \quad (6.34)$$

$$actD = \text{Adaptive Cruise Control} \quad (6.35)$$

$$h = \text{Yaw Momentum Causes Oversteering} \quad (6.36)$$

No element is substituted in Table 6-3. One activity edge is removed, namely the data flow from the activity “Compute Brake Force” to the output pin o_2 :

$$e_1^{remove} = \left(\begin{array}{c} \text{Compute Brake Force,} \\ \text{Required Brake Force, true, } o_2 \end{array} \right) \quad (6.37)$$

In consequence, the set of activity edges to be removed is as follows:

$$R = \{e_1^{remove}\} \quad (6.38)$$

In addition, several elements are inserted. Specifically, one additional input pin called i_4 is added along with an activity edge that transmits the yaw rate to the activity “Compute Brake Force.” Furthermore, a decision node is inserted along with three data flows as follows:

$$e_1^{insert} = i_4 \quad (6.39)$$

$$e_2^{insert} = (i_4, \text{Yaw Rate}, \text{Compute Brake Force}) \quad (6.40)$$

$$e_3^{insert} = \left(\max \text{ brake force} * 0.85^{\frac{5}{yaw}} \right) \quad (6.41)$$

$$e_4^{insert} = \left(\begin{array}{c} \text{Compute Brake Force, Limited Brake Force,} \\ \max \text{ brake force} * 0.85^{\frac{5}{yaw}} \end{array} \right) \quad (6.42)$$

$$e_5^{insert} = \left(\begin{array}{c} \max \text{ brake force} * 0.85^{\frac{1}{yaw}}, \text{ Limited Brake Force,} \\ \text{no, Compute Brake Force} \end{array} \right) \quad (6.43)$$

$$e_6^{insert} = \left(\begin{array}{c} \max \text{ brake force} * 0.85^{\frac{5}{yaw}}, \\ \text{Limited Brake Force, yes, } o_2 \end{array} \right) \quad (6.44)$$

$$I = \{e_1^{insert}, e_2^{insert}, e_3^{insert}, e_4^{insert}, e_5^{insert}, e_6^{insert}\} \quad (6.45)$$

The partial mitigation pm_i in Table 6-3 can hence be written as

$$pm_1 = \left(\begin{array}{c} \text{Mitigation for Hazard H1,} \\ \text{Adaptive Cruise Control,} \\ h_1, R, I, S \end{array} \right) \quad (6.46)$$

with $S^{\text{Mitigation for Hazard H1}} = \emptyset$ because Table 6-3 does not specify any elements to be substituted. Since there is only one mitigation partition in Figure 5-2, the partial mitigation from Table 6-3 is the only in order to mitigate hazard H1. Therefore, the conceptual mitigation for this hazard is the set containing $pm_1^{\text{Mitigation for Hazard H1}}$.

$$CM^{\text{Mitigation for Hazard } H1} = \{pm_1\} \quad (6.47)$$

Each partial mitigation belonging to the same conceptual mitigation must be contained in a Hazard Relation Diagram in the form of one mitigation partition (see Section 5.3). It is to note however, that mitigation partitions do not depict the human-readable name, as this is merely used to group those partial mitigations that belong to the same conceptual mitigation.

6.2.4. Hazard Relation Diagram

The artifact type “Hazard Relation Diagram” is the output artifact produced by the creation steps outlined in Section 6.1. On the one hand, they contain UML activity diagrams containing hazard-mitigating requirements, hazard-inducing requirements, and other functional requirements without impact on safety. On the other hand, they contain hazard-analysis results, i.e. hazards, trigger conditions, and safety goals. Moreover, Hazard Relation Diagrams visualize the dependencies between these modeling concepts.

Formalization

A Hazard Relation Diagram $hazRD$ is described as the following tuple:

$$hazRD = (hrd, AD, h, tc^h, sg^h, CM, hr, HA) \quad (6.48)$$

where hrd is a unique designation of the Hazard Relation Diagram. In a Hazard Relation Diagram, a set of activity diagrams $AD^{hrd} = \{ad_1^{hrd}, ad_2^{hrd}, \dots, ad_n^{hrd}\}$ according to (6.1) is associated with exactly one hazard h^{hrd} from (6.19), its trigger condition tc^h in its Boolean representation according to (6.20), and the safety goal sg^h . Figure 6-4 shows the Hazard Relation Diagram from Figure 5-2 overlaid with the formal elements from (6.48).

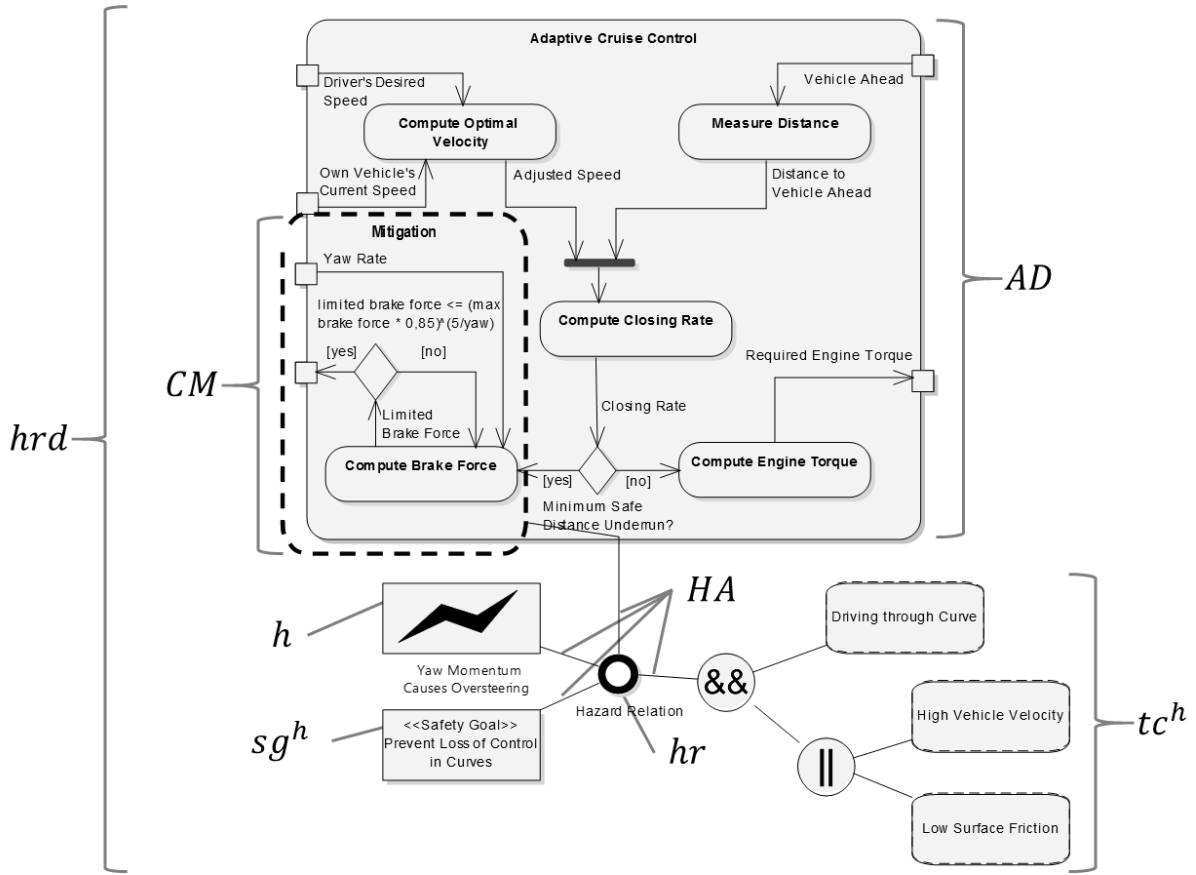


Figure 6-4 Hazard Relation Diagram from Figure 5-2 overlaid with the formal elements from (6.48).

Since the well-formedness rules from Section 5.5 prescribe that the trigger condition and the safety goal pertain to the hazard, we define⁴ the following relationship:

$$tc^h \in_t h \quad (6.49)$$

$$sg^h \in_t h \quad (6.50)$$

In addition to trigger conditions, safety goals, and the hazard they pertain to, Hazard Relation Diagrams comprise a conceptual mitigation CM^{hrd} according to (6.47) which in turn consists of a set of partial mitigations according to (6.27). For each partial mitigation $pm \in CM^{hrd}$, the activity diagram referenced in the partial mitigation must correspond to one of the activity diagrams in AD^{hrd} :

$$\forall pm \in CM^{hrd} | \exists ad \in AD^{hrd} : ad \in_t pm \quad (6.51)$$

⁴ Here and in the following, we use the operator $x \in_t U$ to denote that some element x is element of some tuple U , regardless of its index within U . Since the tuples in this research are unambiguous due to their membership order and type-specificity, the following holds as a simplification of [Awodey 2011]:

$$\in_t := \exists x | U = (u_1, u_2, \dots, u_n) \wedge x = u_t, 0 < t < n$$

As outlined in Section 6.2.3, each partial mitigation is depicted in the Hazard Relation Diagram using a mitigation partition. However, in contrast to the partial mitigation, the mitigation partition in the Hazard Relation Diagram will not include items that have been removed from the source activity diagram containing the hazard-inducing requirements. This is because a mitigation partition signifying removed elements would be empty, which may introduce ambiguity as to what information is highlighted in the mitigation partition. Yet, validation will not take into account individual changes, but instead will consider whether the collective impact of all enacted changes on the requirements are adequate to mitigate a hazard (see Section 6.2.2).

Hazard Relation Diagram consists furthermore of a Hazard Relation hr^{hrd} and a set of hazard associations $HA = \{ha_1, ha_2, \dots, ha_n\}$. Hazard associations can be seen as tuples connecting the Hazard Relation with either the hazard, the trigger condition, the safety goal, or the mitigation partitions:

$$ha^{hrd} = (srt, end) | srt = hr^{hrd} \vee \left(\begin{array}{l} end = h^{hrd} \\ \wedge end = tc^h \wedge end = sg^h \wedge \\ end = pm^h: \forall pm \in CM^{hrd} \end{array} \right) \quad (6.52)$$

Example

In Figure 5-2, we have shown a Hazard Relation Diagram which contains a conceptual mitigation for Hazard H1 from Table 2-1. The hazard-mitigating requirements upon which this Hazard Relation Diagram is based are created by applying the changes from the partial mitigation in Table 6-3 on the activity diagram in Figure 5-2 (see Section 6.3.1). According to (6.48), the Hazard Relation Diagram in Figure 5-2 can hence be written as:

$$hazRD = \left(\begin{array}{l} HRD \text{ for Hazard H1 and Mitigation M1,} \\ AD, h, tc^h, sg^h, CM, hr, HA \end{array} \right) \quad (6.53)$$

As the name for this Hazard Relation Diagram, the following human-readable designation was assigned:

$$hrd = HRD \text{ for Hazard H1 and Mitigation M1} \quad (6.54)$$

The hazard depicted in Figure 5-2 is hazard H1 in Table 6-1, which has the safety goal “Prevent Loss of Control in Curve,” and the trigger conditions from (6.21). The hazard in this Hazard Relation Diagram is therefore the same as the one in as shown in (6.23) and the trigger conditions are the same as in (6.21),

$$h^{HRD \text{ for Hazard } H1 \text{ and Mitigation } M1} = h_1 \quad (6.55)$$

$$tc^h = tc^{h_1} \quad (6.56)$$

while the safety goal is:

$$sg^h = \textit{Prevent Loss of Control in Curve} \quad (6.57)$$

According to (6.47), the conceptual mitigation in the Hazard Relation Diagram from in Figure 5-2 is a set consisting of exactly one partial mitigation:

$$CM = \{pm^{Mitigation \text{ for Hazard } H1}\} \quad (6.58)$$

Therefore, the set of activity diagrams in Figure 5-2 is also a set, specifically the activity diagram containing hazard-inducing requirements (upon which the partial mitigation is applied), as shown in Figure 2-1:

$$AD = \{actD^{Adaptive \text{ Cruise Control}}\} \quad (6.59)$$

Since there is only one mitigation partition in the Hazard Relation Diagram in Figure 5-2, this means that there are exactly four hazard associations, which connect the Hazard Relation, hr , to the hazard, the safety goal, the trigger conditions, and the mitigation partition, respectively:

$$ha_1 = (hr, h) \quad (6.60)$$

$$ha_2 = (hr, sg^h) \quad (6.61)$$

$$ha_3 = (hr, tc^h) \quad (6.62)$$

$$ha_4 = (hr, pm^{Mitigation \text{ for Hazard } H1}) \quad (6.63)$$

$$HA = \{ha_1, ha_2, ha_3, ha_4\} \quad (6.64)$$

6.3. Artifact Creation

Creation of Hazard Relation Diagrams makes use of OMG's Query/View/Transformation Operational Mappings language (QVTo, see [OMG 2016]). QVTo is a transformation language, which allows enacting UML model transformations by either manipulating specific model elements (i.e. adding, removing, or substituting model elements), or by converting diagrams on an ontological level. In Section 6.1, we have outlined that creating Hazard Relation Diagrams is a

two-step process. Both steps involve the use of QVTo and is explained in the following subsections. Specifically, in Section 6.3.1, we outline how an intermediate activity diagram containing hazard-mitigating requirements can be created using the mitigation templates from Section 6.2.3. This corresponds to the cogwheel labeled “1” in Figure 6-1 (see Section 6.1). Subsequently in Section 6.3.2, we explain how to append the contextual hazard information from the hazard analyses results (see Section 6.2.2), thereby creating the Hazard Relation Diagram. This corresponds to the cogwheel labeled “2” in Figure 6-1 (see Section 6.1). Moreover, in the following subsections, we will demonstrate how the satisfaction of the formalized well-formedness rules for Hazard Relation Diagrams from Section 5.5 can be ensured.

6.3.1. Creating Hazard-Mitigating Requirements

As outlined in Section 5.1, hazard-mitigating requirements documented in activity diagrams are the basis of Hazard Relation Diagrams. Hazard-mitigating requirements can be understood as specific changes to the functional requirements.

Principle Mechanism to Create Hazard-Mitigating Requirements

In the example in Figure 6-5, the principle mechanism to create hazard-mitigating requirements from transformation steps specified by means of the mitigation template from Table 6-2 is shown.

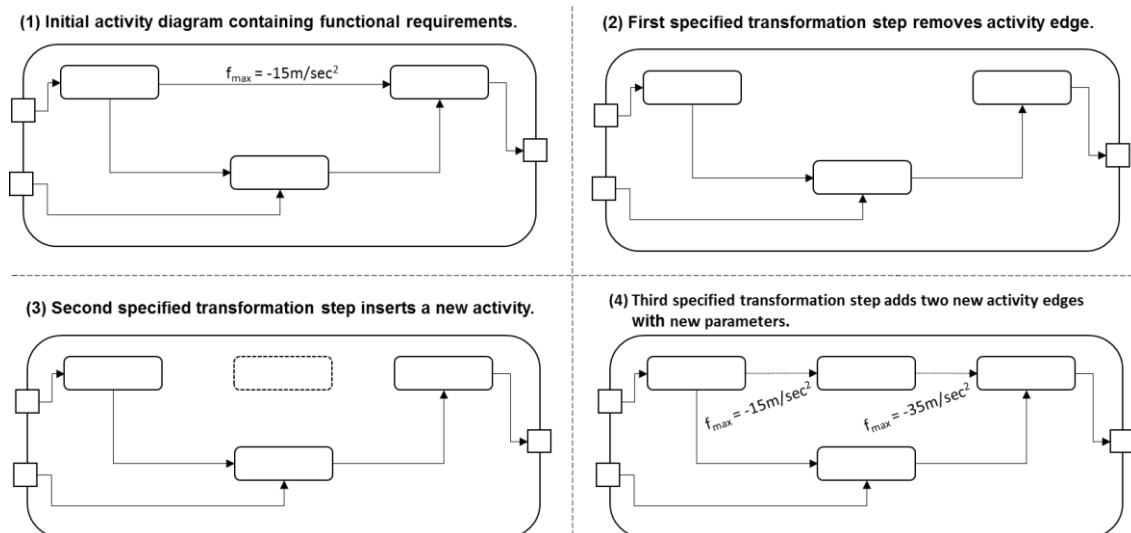


Figure 6-5 Simplified Example of Transformation Steps to Create Hazard-Mitigating Requirements.

In this example, an activity edge bearing a specific message (panel (1) in Figure 6-5) is replaced by a new activity and corresponding incoming and outgoing activity edges. As can be seen in panel (2), the stakeholder first specifies a transformation step to remove the old activity edge. Next, as can be seen in panel (3), the stakeholder specified an insertion operation to add a new

activity. Finally, in panel (4), the stakeholder specifies two new activity edges with new messages to add a “context” (in the sense of the call order indicated by incoming and outgoing activity edges according [OMG 2015b]) to the new activity. This example also illustrates the case, where a hazard-mitigating requirements corresponds to the change of a parameter for an activity. In accordance with [OMG 2015b], parameter nodes for UML activities and opaque actions are represented in a simplified manner using the message payload on activity edges. This means that in a case, where a hazard-mitigating requirement consists of a changed parameter for an activity (e.g., the substitution of a value, tolerance, worst case execution time, etc.) can be represented using mitigation templates by substituting activity edges with modified messages.

QVTo Script to Enact Transformation Steps

Documenting these changes is done using partial mitigation (see Section 6.2.3). These changes are automatically enacted on the activity diagram containing the functional requirements ad^{fr} in order to create an activity diagram containing the hazard-mitigating requirements ad^{hmr} :

$$(actD^{fr}, CM^h) \xrightarrow{q^{hmr}} actD^{hmr} \quad (6.65)$$

where $CM^h = \{pm_1, pm_2, \dots, pm_n\}$ is a set of all partial mitigations referencing hazard h . Enacting these specific changes is done using a QVTo script q^{hmr} which contains code operations op to systematically perform all insertions, removals, and substitutions of elements in all partial mitigations $pm_i^h \in CM^h$ that pertain to the same conceptual mitigation. It follows:

$$\begin{aligned} q^{hmr} &= (sig, op^{insert}, op^{remove}, op^{substitute}, chk) | \\ op^{insert} &= I^{cm_1} \cup I^{pm_2} \cup \dots \cup I^{pm_n} \\ \wedge op^{remove} &= R^{pm_1} \cup R^{pm_2} \cup \dots \cup R^{pm_n} \\ \wedge op^{substitute} &= S^{pm_1} \cup S^{pm_2} \cup \dots \cup S^{pm_n} \\ \wedge pm_i &\in CM^h \wedge 0 < i < |CM| \end{aligned} \quad (6.66)$$

where sig is QVTo script signature, i.e. the head of every QVTo script which specifies input and output models and selects their meta-models (see [OMG 2016] for more details on QVTo code) and chk is code needed to ensure that the resulting hazard-mitigating requirements are syntactically correct according to Well-Formedness Rule 9 through Well-Formedness Rule 11.

Script Signature

Listing 1 below shows QVTo-style pseudo code which can be used for *sig*.

```

1  --- select meta-model types of input and output
2  modeltype AD uses 'http://www.eclipse.org/uml2/5.0.0/UML';
3  modeltype mitigation uses 'http://sse.uni-due.de/mitigationtemplate';
4  --- define transformation function signature
5  ---  $actD^{fr}$  is the activity diagram containing hazard-inducing requirements to be
6  --- changed to hazard-mitigating requirements  $actD^{hmr}$ .
7  ---  $CM^h$  is a set of partial mitigations
8  transformation generateHazardMititgatingReqs
9      (in  $actD^{fr}$ :AD, in  $CM^h$ :mitigation, out  $actD^{hmr}$ :AD);
10
11 main() {
12     --- select mitigated hazard by checking which hazard is referenced in first
13     --- mitigation template in the array
14     Let  $haz = h \in_i pm_i^h | pm_i^h \in CM^h$ 
15     --- check if all templates reference same hazard
16     foreach  $pm_i^h \in CM^h$  {
17         if  $h \in_i pm_i^h \neq haz$  {
18             throw error("Multiple Conceptual Mitigations detected. Aborting");
19             return  $ad^{hmr} = \emptyset$ ;
20         }
21     }

```

Listing 1 Pseudo-Code Signature *sig* of the QVTo Script q^{hmr} .

As can be seen, QVTo requires the meta-models of the input and output artifacts to be specified. Conventionally, the fictitious URI shown in line 2 of Listing 1 is used to point to the UML superstructure for use in QVTo scripts to be executed using the QVT Operational Mapping Plugin [Eclipse QVT v3.5.0] available for the Eclipse Modeling Tool project [Eclipse Modeling Tools]. The fictitious URI given in line 3 points to the URI of an Ecore model representing the meta-model for the partial mitigations $pm_i \in CM^h$ which must be part of the implementation. In line 9, the source activity diagram containing hazard-inducing requirements ad^{fr} and the set of mitigation templates CM^h are defined as inputs and the target activity diagram that will contain the hazard-mitigating requirements ad^{hmr} is referenced. Line 10 designates the beginning of the transformation script before the pseudo-code in lines 14 to 20 check if every mitigation template references the same hazard. If not, an error is thrown and transformation is aborted by returning $ad^{hmr} = \emptyset$. If the references hazards are all the same Well-Formedness Rule 13 is satisfied.

Inserting Diagram Elements

The next elements of the QVTo script q are the insertion, removal, and substitution operations op^{insert} , op^{remove} , and $op^{subsistute}$, respectively.

```

1  --- insert activity diagram elements from each mitigation template
2  foreach partial mitigation  $pm_i^h \in CM^h$  {
3    foreach operation  $op_i \in I \mid I \in pm_i^h$  {
4      foreach element to be inserted  $el \in op_i^h$  {
5        if  $el \in op_i^h$  is of type activity {
6          --- insert new activity by computing new set of activities
7           $A^{actD^{hmr}} = A^{actD^{fr}} \cup el$  }
8        if  $el \in op_i^h$  is of type pin {
9          --- insert new pin by computing new set of pins
10          $p^{actD^{hmr}} = p^{actD^{fra}} \cup el$  }
11        if  $el \in op_i^h$  is of type control node {
12          --- insert new control node by computing new set of control nodes
13           $C^{actD^{hmr}} = C^{actD^{fr}} \cup el$  }
14        if  $el \in op_i^h$  is of type activity edge {
15          ---insert new activity edge by computing new set of activity edges
16          --- make sure source and target of the activity edge exist
17           $E^{actD^{hmr}} = E^{actD^{fr}} \cup el$ 
18           $el = (src, m, tar): src, tar \in A^{actD^{hmr}} \cup p^{actD^{hmr}} \cup C^{actD^{hmr}}$  }
19        }
20      }
21    }

```

Listing 2 Pseudo-Code of insertion operation op^{insert} of the QVTo Script q^{hmr} .

Listing 2 shows the insertion of elements. In line 2, the insertion operations are conducted for every partial mitigation in CM^h . As can be seen, for every insertion operation (line 3), it is checked what type the element to be inserted is (lines 5, 8, 11, and 14, respectively) and the appropriate set of diagram elements is created by unifying the old set of activities (line 7), pins (line 10), control nodes (line 13), or activity edged (line 17). For activity edges, however, the activity edge to be inserted must have a source and a target that is already part of the hazard-mitigating requirements (see line 18), as otherwise there is a chance that “dangling” activity edges may be inserted into the diagram. Hence, before inserting activity edges, all other activity diagram elements are inserted, thereby partly satisfying Well-Formedness Rule 9.

Example

The result of the application of the partial mitigation from Table 6-3 by means of the pseudo-code in Listing 2 is shown in Figure 6-6.

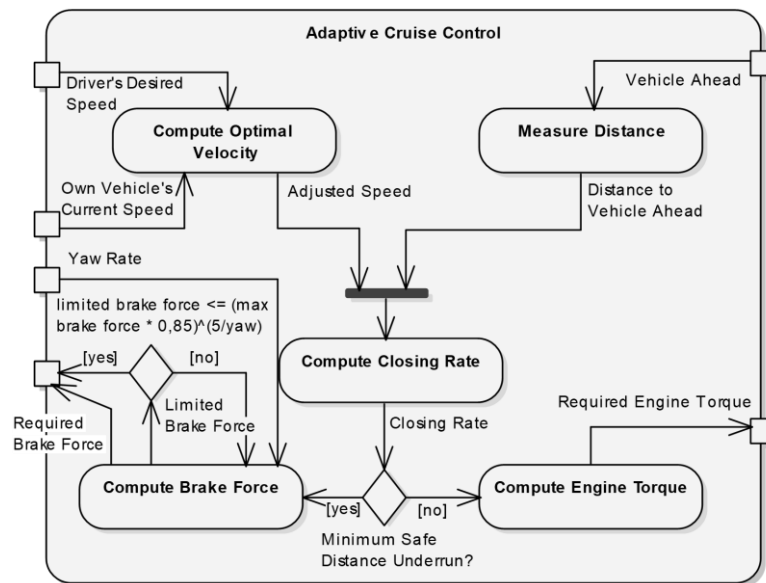


Figure 6-6 Result of Applying the Partial Mitigation from Table 6-3 by Means of Listing 2.

As can be seen, one anonymous input pin i_4 has been inserted, along with the decision node $\max \text{brake force} * 0.85^{5/yaw}$. Furthermore, a number of activity edges were inserted, connecting the newly inserted pin and control node to the activity “Compute Brake Force.” As can be seen, the activity edge “Required Brake Force” from “Compute Brake Force” to the output pin o_2 is still in place, as the remove operation specified in Table 6-3 hasn’t been enacted yet.

Removing Diagram Elements

Listing 3 shows the pseudo-code to remove elements. As can be seen, removing elements from the activity diagram is similar to their insertion: depending on what type the element to be removed is, a new set of such elements is computed that does not include the element to be removed. However, in contrast to insertion operations, when elements are removed, any activity edge that had the removed element as a source or a target must be removed as well in order to avoid dangling activity edges to remain in the hazard-mitigating requirements once removing operations have completed. Therefore, an additional removing operation is contained in Listing 3 in line 18 in partial satisfaction of Well-Formedness Rule 9.

```

1  --- remove activity diagram elements from each mitigation template
2  foreach partial mitigation  $pm_i^h \in CM^h$  {
3    foreach operation  $op_i \in_t R \mid R \in_t pm_i^h$  {
4      foreach element to be removed  $el \in_t op_i^h$  {
5        if  $el \in_t op_i^h$  is of type activity {
6          --- remove activity by computing new set of activities
7           $ActD^{hmr} = ActD^{dr} \setminus el$ 
8        if  $el \in_t op_i^h$  is of type pin {
9          --- remove pin by computing new set of pins
10          $pactD^{hmr} = pD^{dr} \setminus el$ 
11        if  $el \in_t op_i^h$  is of type control node {

```

```

12      --- remove control node by computing new set of control nodes
13       $C^{actD^{hmr}} = C^{actD^{fr}} \setminus \{el\}$ 
14      if  $el \in_t op_i^h$  is of type activity edge {
15      ---remove activity edge by computing new set of activity edges
16       $E^{actD^{hmr}} = E^{actD^{hir}} \setminus \{el\}$ 
17      --- remove activity edges connected to the removed element
18       $E^{actD^{hmr}} = E^{actD^{fr}} \setminus \{ex \mid \forall ex \in E^{ad^{hmr}} : src^{ex} \vee tar^{ex} = el\}$ 
19      }
20  }
21  }

```

Listing 3 Pseudo-Code of remove operation op^{remove} of the QVTo Script q^{hmr} .

Example

The result of applying the remove operations specified in Table 6-3 by means of the pseudo-code in Listing 3 is shown in Figure 6-7. The superfluous activity edge “Required Brake Force” from “Compute Brake Force” to the output pin o_2 has now been removed, leaving only those hazard-mitigating requirements in place that were newly inserted.

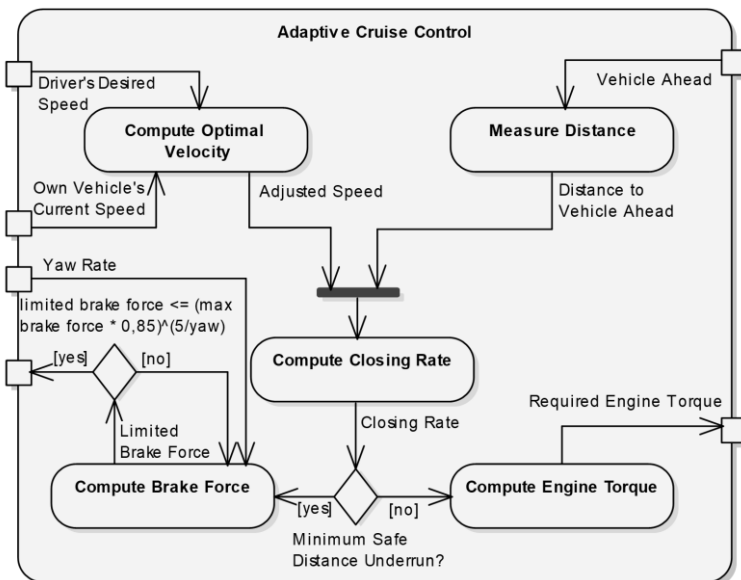


Figure 6-7 Result of Applying the Partial Mitigation from Table 6-3 by Means of Listing 2 and Listing 3.

Substituting Diagram Elements

In the following Listing 4, pseudo-code to substitute elements is outlined. For simplicity, it is assumed Listing 4 that only elements of the same type can be substituted with one another (see lines 6, 12, 18, and 24) albeit in principle non-homogenous substitutions may be permitted. As can be seen, substitution is basically an insertion operation followed by a removing operation as outlined in Listing 2 and Listing 3, respectively. However, in contrast to the pseudo-code in Listing 2 and Listing 3, some additional effort must be spent in order to identify whether the substituted element was a source of an activity edge (see line 33) or a target of an activity edge (see line 45). All activity edges for which this is the case must be replaced by new ones

which connect to the substituting element in as source or target, respectively. Again, doing so results in partial fulfillment of Well-Formedness Rule 9.

```

1  --- substitute activity diagram elements from each mitigation template
2  foreach partial mitigation  $pm_i^h \in CM^h$  {
3      --- for each two elements to be substituted..
4      foreach operation  $op_i \in_t R \mid R \in_t pm_i^h$  {
5          foreach element to be removed  $el^{old}, el^{new} \in_t op_i^h$  {
6              if  $el^{old}, el^{new} \in_t op_i^h$  are of type activity {
7                  --- insert new activity by computing new set of activities
8                   $A' = A^{actD^{fr}} \cup el^{new}$ 
9                  --- remove old activity by computing new set of activities
10                  $A^{actD^{hmr}} = A' \setminus el^{old}$ 
11             }
12             if  $el^{old}, el^{new} \in_t op_i^h$  are of type pin {
13                 --- insert new pin by computing new set of pins
14                  $P' = P^{actD^{fr,a}} \cup el^{new}$ 
15                 --- remove pin by computing new set of pins
16                  $P^{actD^{hmr}} = P' \setminus el^{old}$ 
17             }
18             if  $el^{old}, el^{new} \in_t op_i^h$  are of type control node {
19                 --- insert new control node by computing new set of control nodes
20                  $C' = C^{actD^{fr}} \cup el^{new}$ 
21                 --- remove control node by computing new set of control nodes
22                  $C^{ad^{hmr}} = C' \setminus el^{old}$ 
23             }
24             if  $el^{old}, el^{new} \in_t op_i^h$  are of type activity edge {
25                 ---insert new activity edge by computing new set of activity edges
26                 --- make sure source and target of the activity edge exist
27                  $E' = E^{ad^{fr}} \cup el^{new}$ 
28                  $el^{new} = (src, m, tar) : src, tar \in A^{actD^{hmr}} \cup P^{actD^{hmr}} \cup C^{actD^{hmr}}$ 
29                 ---remove activity edge by computing new set of activity edges
30                  $E^{ad^{hmr}} = E' \setminus el^{old}$ 
31             }
32             --- find all activity edges that have  $el^{old}$  as source
33              $E'^{actD^{fr}} = E^{actD^{hmr}} \setminus e \mid \forall e = (src, m, tar) \in E^{actD^{fr}} : src \in_t el^{old} \vee tar \in_t el^{old}$ 
34             --- for every element that has  $el^{old}$  as source
35             foreach  $(e^{old} = (src, m, tar) \in_t E'^{actD^{fr}} \mid src = el^{old})$  {
36                 --- make a new activity edge to connect to the new element
37                 --- with same message, guard, and target
38                  $e^{new} = (src, m, tar) \mid m, tar \in_t e^{old} \wedge src = el^{new}$ 
39                 --- insert new activity edge into set of activity edges
40                  $E' = E^{ad^{hmr}} \cup e^{new}$ 
41                 ---remove activity edge by computing new set of activity edges
42                  $E^{actD^{hmr}} = E' \setminus e^{old}$ 
43             }
44             --- for every element that has  $el^{old}$  as target
45             foreach  $(e^{old} = (src, m, tar) \in_t E'^{ad} \mid tar = el^{old})$  {
46                 --- make a new activity edge to connect to the new element
47                 --- with same source and message
48                  $e^{new} = (src, m, tar) \mid src, m \in_t e^{old} \wedge tar = el^{new}$ 
49                 --- insert new activity edge into set of activity edges
50                  $E' = E^{ad^{hmr}} \cup e^{new}$ 
51                 ---remove activity edge by computing new set of activity edges
52                  $E^{ad^{hmr}} = E' \setminus e^{old}$ 
53                 ---remove activity edge by computing new set of activity edges
54                  $E^{ad^{hmr}} = E' \setminus e^{old}$ 

```

```

55     }
56     }
57 }

```

Listing 4 Pseudo-Code of substitution operation $op^{substitute}$ of the QVTo Script q^{hmr} .

Example

When applying the pseudo-code in Listing 4, all substitute operations specified in some partial mitigation is enacted in a similar fashion to the insert and remove operation from Listing 2 and Listing 3. However, since the partial mitigation from Table 6-3 does not contain any substituted elements, the result of applying that partial mitigation to the result from op^{remove} using the pseudo-code in $op^{substitute}$ will not change the hazard-mitigating requirements any further and yield the same diagram as shown in Figure 6-7.

Additional Well-Formedness Rule Checks

When Listing 1 through Listing 4 have been executed, the hazard-mitigating requirements will consist of activity diagram modeling elements. In complete satisfaction of Well-Formedness Rule 9 as well as in order to satisfy Well-Formedness Rule 10 and Well-Formedness Rule 11, checks must be performed, as the developer that specified the mitigation template may have made a mistake resulting in incorrect diagram syntax. In Listing 5, pseudo-code for such checks are shown. Albeit some effort was undertaken in Listing 1 through Listing 4 to avoid dangling activity edges, doing so only captures dangling activity edges, where the source or the target of an existing activity edge was manipulated. Since in principle, it could be possible for the developer to intentionally add (albeit most likely by accident) activity edges that are not connected to some other modeling element, this check must be repeated after all insertion, remove, or substitution operations are complete. In Listing 5, this is shown in lines 1 through 6, where it is checked if there is any activity edge that has no source or no target, thereby satisfying completely. The following section of pseudo-code from line 7 to 20 is checking for orphaned cliques of circularly connected, but otherwise unconnected activities or control nodes in satisfaction of Well-Formedness Rule 11. This is done by checking every possible subset of sequential activity edges (i.e. where the target of one activity edge is the source of the next activity edge, see line 10) can be traced to an input pin and an output pin. This is done by first assuming that the sequence of activity can neither be traced to either an input or an output and then checking for every activity edge, if its source is an input pin or the target is an output pin. If either an input pin or an output pin cannot be detected as a source or target of any activity edge in the clique, then the clique is considered orphaned and removed from the hazard-mitigating requirements. It is to note that removing the clique only extends to removing the associated activity edges.

Therefore, it has to be checked next, if removing these activity edges orphaned activities, pins, or control nodes. This is done Listing 5 from lines 21 to 63 in fulfillment of Well-Formedness Rule 10. First, it is assumed that all activities (lines 22 to 35), pins (lines 36 to 49), and control nodes (lines 49 to 63) are orphaned. Next, for each activity, pin, or control node, it is checked if there is at least one activity edge in the hazard-mitigating requirements that has the current activity, pin, or control node as either the source or the target. If one such activity edge was found, the current modeling element is not orphaned. Orphaned modeling elements are removed from the hazard-mitigating requirements.

```

1  --- check for dangling edges
2  --- find all activity edges where source or target is null
3  foreach  $e = (src, m, tar) | e \in E^{ad^{hmr}}$  {
4      if  $src = \varepsilon \vee tar = \varepsilon$  {
5           $E^{actD^{hmr}} = E^{actD^{hmr}} \setminus e$  }
6      }
7  --- check for orphaned cliques
8  --- select all possible cliques where at least one activity edge has a target
9  --- that is the same also the source of some other activity edge
10 foreach  $V \subseteq E^{actD^{hmr}} | V = \{e_1, e_2, \dots, e_n\} : e_i = (src^{e_i}, m^{e_i}, tar^{e_i}) \wedge tar^{e_i} = src^{e_{i+1}}$ 
11     --- assume  $V$  is an orphaned clique
12     boolean inputFound = false
13     boolean outputFound = false
14     foreach  $e_i \in V$  {
15         --- check if a path can be traced to at least one input pin
16         if  $src^{e_i} = p \in P^{ad}$  { inputFound = true }
17         if  $tar^{e_i} = p \in P^{ad}$  { outputFound = true }
18     }
19     if inputFound  $\vee$  outputFound = false {  $E^{ad^{hmr}} = E^{ad^{hmr}} \setminus V$  }
20 }
21 --- check for orphaned activities, pins, control nodes
22 --- find all activities that are not source or target of an activity edge
23 foreach  $a \in A^{actD^{hmr}}$  {
24     --- assume  $a$  is orphaned
25     boolean orphaned = true
26     foreach  $e = (src, m, tar) | e \in E^{actD^{hmr}}$  {
27         --- if  $src$  or  $tar$  point to  $a$ , then  $a$  isn't orphaned
28         if  $src = a \vee tar = a$  {
29             orphaned = false
30         }
31         --- if no activity edge with pointing to  $a$  was found, it will be removed
32     if orphaned = true {
33          $A^{actD^{hmr}} = A^{actD^{hmr}} \setminus a$ 
34     }
35 }
36 --- find all pins that are not source or target of an activity edge
37 foreach  $p \in P^{ad^{hmr}}$  {
38     --- assume  $p$  is orphaned
39     boolean orphaned = true
40     foreach  $e = (src, m, tar) | e \in E^{actD^{hmr}}$  {
41         --- if  $src$  or  $tar$  point to  $p$ , then  $p$  isn't orphaned
42         if  $src = p \vee tar = p$  {
43             orphaned = false
44         }

```

```

45     --- if no activity edge with pointing to p was found, it will be removed
46     if orphaned = true {
47          $p^{actD^{hmr}} = p^{actD^{hmr}} \setminus p$ 
48     }
49 }
50 --- find all control nodes that are not source or target of an activity edge
51 foreach  $c \in C^{actD^{hmr}}$  {
52     --- assume c is orphaned
53     boolean orphaned = true
54     foreach  $e = (src, m, tar) | e \in E^{actD^{hmr}}$  {
55         --- if src or tar point to c, then c isn't orphaned
56         if  $src = c \vee tar = c$  {
57             orphaned = false
58         }
59     --- if no activity edge with pointing to p was found, it will be removed
60     if orphaned = true {
61          $C^{actD^{hmr}} = C^{actD^{hmr}} \setminus c$ 
62     }
63 }
64 return  $actD^{hmr}$ 
65 }

```

Listing 5 Pseudo-Code Signature *chk* of the QVTo Script q^{hmr} .

Example

Listing 5 hence constitutes the last constituent *chk* of the QVTo script q^{hmr} , and would change the output of $op^{substitute}$ by removing orphaned individual modeling elements, cliques thereof, or dangling activity edges. In the case of the partial mitigation in Table 6-3, the pseudo-code in *chk* would not find any such modeling elements and therefore not change the hazard-mitigating requirements any further, thereby rendering the same output as $op^{substitute}$, i.e. again the same diagram as shown in Figure 6-7.

Result of QVTo Script Application

In practice, all listings can be written into one code file (as signified by the return statement and closing curly brace in Listing 5, lines 64 and 65). Executing all listings will hence result in applying the changes specified in the partial mitigation to the hazard-inducing requirements formalized in Section 6.2.1, yielding hazard-mitigating requirements from Figure 6-7, which can be formally written as outlined in the following.

Since q^{hmr} does not override the name, the name can be freely chosen, e.g.:

$$ad = ACC \text{ Hazard } H1 \text{ Mitigation} \quad (6.67)$$

As can be seen from Figure 6-7, the activity diagram contains five activities and three control nodes, two of which are decision nodes and one is a fork node after Table 6-3. In order to create Hazard Relation Diagrams in the next step (see Section 6.3.2), it is necessary to provide a unique

name to control nodes, although UML does not prescribe control nodes to have an identities. In the following, anonymous unique names will refer to the fork, join, and merge nodes as well as the human-readable name will refer to the decision node added by the partial mitigation.

$$A^{ACC \text{ Hazard } H1 \text{ Mitigation}} = \left\{ \begin{array}{l} \text{Compute Optimal Velocity,} \\ \text{Measure Distance,} \\ \text{Compute Closing Rate,} \\ \text{Compute Brake Force,} \\ \text{Compute Engine Torwque} \end{array} \right\} \quad (6.68)$$

$$P^{ACC \text{ Hazard } H1 \text{ Mitigation}} = \{i_1, i_2, i_3, i_4, o_1, o_2\} \quad (6.69)$$

$$D^{ACC \text{ Hazard } H1 \text{ Mitigation}} = \left\{ \begin{array}{l} \text{Minimum Safe Distance Underrun?}, \\ \text{limited brake force} \leq \left(\text{max brake force} * 0.85^{\frac{5}{yaw}} \right) \end{array} \right\} \quad (6.70)$$

$$F^{ACC \text{ Hazard } H1 \text{ Mitigation}} = \{f_1\} \quad (6.71)$$

$$C^{ACC \text{ Hazard } H1 \text{ Mitigation}} = F^{ACC \text{ Hazard } H1 \text{ Mitigation}} \cup D^{ACC \text{ Hazard } H1 \text{ Mitigation}} \quad (6.72)$$

Furthermore, application of partial mitigation in Table 6-3, the following fourteen activity edges are part of the set of activity edges in the hazard-mitigating requirements:

$$\mathbb{E}^{ACC \text{ Hazard } H1 \text{ Mitigation}} = \quad (6.73)$$

$$\left(\begin{array}{l} (i_1, \text{Driver's Desired Speed}), (i_2, \text{Own Vehicle's Current Speed}), \\ \text{Compute Optimal Velocity}, \text{Compute Optimal Velocity} \\ \text{Compute Optimal Velocity}), (i_3, \text{Vehicle Ahead, Measure Distance}), \\ \text{Adjusted Speed, } f_1 \\ \text{Measure Distance,} \\ \text{Distance to Vehicle Ahead, } f_1, \left(\begin{array}{l} f_1, \text{Adjusted Speed} + \\ \text{Distance to Vehicle Ahead,} \\ \text{Compute Closing Rate} \end{array} \right), \\ \text{Compute Closing Rate, Closing Rate}, \left(\begin{array}{l} \text{Minimum Safe Distance Underrun?}, \\ \text{Minimum Safe Distance Underrun?}, \\ \text{Closing Rate, Compute Engine Torque} \end{array} \right), \\ \text{Compute Engine Torque,} \\ \text{Required Engine Torque, } o_1, \left(\begin{array}{l} \text{Minimum Safe Distance Underrun?}, \\ \text{Closing Rate, Compute Brake Force} \end{array} \right), \\ \left(\begin{array}{l} \text{Compute Brake Force,} \\ \text{Limited Brake Force,} \\ \text{limited brake force} \leq \\ \left(\text{max brake force} * 0.85^{\frac{5}{yaw}} \right) \end{array} \right), \left(\begin{array}{l} \text{limited brake force} \leq \\ \left(\text{max brake force} * 0.85^{\frac{5}{yaw}} \right), \\ \text{Limited Brake Force,} \\ \text{Compute Brake Force} \end{array} \right), \\ \left(\begin{array}{l} \text{limited brake force} \leq \\ \left(\text{max brake force} * 0.85^{\frac{5}{yaw}} \right), \\ \text{Limited Brake Force, yes, } o_2 \end{array} \right), (i_4, \text{Yaw Rate, Compute Brake Force}) \end{array} \right)$$

It is to note, that albeit the pseudo-code above does create activity diagrams containing hazard-mitigating requirements, dependence checks of operations have not been included as they are beyond the scope of this research. Yet, in some cases, such checks might be desirable, for example, to avoid that no element is removed twice, inserted and immediately removed, etc., or to verify other coherence issues between subsequent operations. In Chapter 8, we will critically review such issues.

6.3.2. Creating Hazard Relation Diagrams

In Section 6.3.1, we have presented an approach which allows for the creation of hazard-mitigating requirements from conceptual mitigations specified in mitigation templates. Since Hazard Relation Diagrams are extensions of activity diagrams containing hazard-mitigating requirements $actD^{hmr}$, a Hazard Relation Diagram $hazRD$ can be created by means of the QVT Operational Mappings language as well.

QVTo Script to Create Hazard Relation Diagrams

The QVTo Script to create Hazard Relation Diagrams can be described as:

$$\begin{aligned}
 \text{hazRD} &= (\text{hrd}, AD, h, tc^h, sg^h, CM, hr, HA) | \\
 h \in fha(\text{actD}^{\text{hir}}) \wedge (\text{actD}^{\text{hir}}, pm_i) &\xrightarrow{q^{\text{hmr}}} \text{actD}_i^{\text{hmr}} \\
 \wedge \forall pm_i \in CM \wedge \forall \text{actD}_i^{\text{hmr}} \in AD: & \\
 (\text{actD}_i^{\text{hmr}}, CM^h, fha(\text{actD}^{\text{fr}})) &\xrightarrow{q^{\text{hrd}}} \text{hazRD}
 \end{aligned} \tag{6.74}$$

where $CM^h = \{pm_1^h, pm_2^h, \dots, pm_n^h\}$ is the set of all partial mitigations referencing hazard h , which have been used to create the hazard-mitigating requirements ad^{hmr} out of the functional requirements ad^{fr} . Furthermore, $fha(ad^{\text{fr}})$ is a hazard analysis conducted on the functional requirements ad^{fr} yielding a list of Hazards that contains the hazard h referenced by the partial mitigations in CM^h . The purpose of the QVTo script q^{hmr} is to append contextual information about hazard h , thereby creating a Hazard Relation Diagram. Similarly to the QVTo script from Section 6.3.1, the QVTo script q^{hrd} can hence be defined as:

$$q^{\text{hrd}} = \left(\begin{array}{l} sig, \text{append}^{AD}, \text{append}^h, \text{append}^{sg}, \text{append}^{tc}, \\ \text{append}^{hr}, \text{append}^{CM}, \text{append}^{HA} \end{array} \right) \tag{6.75}$$

where sig is QVTo script signature head and append^x are operations to append the activity diagrams containing hazard-mitigating requirements, $x = AD$, the hazard, $x = h$, the safety goal $x = sg$, the trigger conditions tc specific to h , $x = tc$, a set of mitigation partitions for the partial mitigations, $x = CM$, the Hazard Relation, $x = hr$, and hazard associations, $x = HA$.

Script Signature

Listing 6 shows pseudo-code for the QVTo script signature head sig . As can be seen, the script takes three input parameters: an activity diagram containing hazard-mitigating requirements, a set of specified mitigation templates specific for one hazard, and the results of a hazard analysis, which contain contextual information about the mitigated hazard (among others). The output of the script is a Hazard Relation Diagram. In lines 22 to 27, the same check for hazard specificity is conducted like in Listing 1, thereby fulfilling Well-Formedness Rule 14. After this check is complete, lines 28 to 39 check whether the hazard part of the hazard analysis results $fha(\text{actD}^{\text{fr}})$. If the hazard is not part of $fha(\text{actD}^{\text{fr}})$, contextual information about the hazard cannot be added to the Hazard Relation Diagram and creation fails.

```

1  --- select meta-model types of input and output
2  modeltype AD uses 'http://www.eclipse.org/uml2/5.0.0/UML';
3  modeltype mitigation uses 'http://sse.uni-due.de/mitigationtemplate';
4  modeltype hazardanalysis uses 'http://sse.uni-due.de/FHA';
5  modeltype HRD uses 'http://sse.uni-due.de/HRD';
6  --- define transformation function signature
7  ---  $AD^{hmr}$  is the set of activity diagrams containing
8  --- hazard-mitigating requirements out of which a
9  --- Hazard Relation Diagram  $hazRD$  is to be created.
10 ---  $CM^h$  is a set of partial mitigations needed to append mitigation partitions
11 ---  $fha(actD^{fr})$  is the results of the hazard analyses that was conducted on the
12 --- on the hazard-inducing requirements  $actD^{fr}$  which yield contextual information
13 --- about the hazard  $h$  that was mitigated in  $CM^h$ 
14 transformation generateHazardRelationDiagram
15     (in  $actD^{hmr}$ :AD, in  $CM^h$ :template, in  $fha(actD^{fr})$ : hazardanalysis, out  $hazRD$ :HRD);
16
17 main() {
18     --- select mitigated hazard by checking which hazard is referenced in first
19     --- partial mitigation in the array
20     let  $haz = h \in_t pm_1^h | pm_1^h \in CM^h$ 
21     --- check if all templates reference same hazard
22     foreach  $pm_i^h \in CM^h$  {
23         if  $h \in_t pm_i^h \neq haz$  {
24             throw error("Multiple Conceptual Mitigations detected. Aborting");
25             return  $hazRD = \emptyset$ ;
26         }
27     }
28     --- check if the mitigated hazard is part of the hazard analysis results
29     --- assume the mitigated hazard is not part of the hazard analysis results
30     boolean  $hazardFound = false$ 
31     foreach  $res \in fha(actD^{fr})$  {
32         if  $res = haz$  {  $hazardFound = true$  }
33     }
34     --- if hazard wasn't found, Hazard Relation Diagram cannot be created
35     if  $hazardFound = false$  {
36         throw error
37         ("Mitigated Hazard is not part of Hazard Analysis Result Set. Aborting.");
38         return  $hazRD = \emptyset$ ;
39     }
40     --- check if hazard-mitigating requirements are referenced
41     --- in some mitigation template.
42     foreach  $actD_i^{hmr} \in AD^{hmr}$  {
43         --- assume activity diagram is not referenced in mitigation template
44         boolean  $adReferenced = false$ 
45         foreach  $pm_i^h \in CM^h$  {
46             if  $actD_i^{hmr} \in_t pm_i^h$  {
47                  $adReferenced = true$ 
48             }
49         }
50         if  $adReferenced = false$  {
51             throw error("No Mitigation Template available for Hazard-Mitigating
52             Requirements. Cannot create mitigation partition. Aborting.");
53             return  $hazRD = \emptyset$ ;
54         }
55     }

```

Listing 6 Pseudo-Code Signature sig of the QVTo Script q^{hrd} .

Similarly, in lines 40 to 55, it is checked whether or not the activity diagrams containing hazard-mitigating requirements have at least one equivalent mitigation template. This is necessary in order to ensure a mitigation partition can be added to the Hazard Relation Diagram subsuming modeling elements that are part of some activity diagram (see Section 5.4). If at least one activity diagram has no referencing mitigation template, creation of the Hazard Relation Diagram result in an error.

Example

Listing 6 hence checks the prerequisites in order to create a Hazard Relation Diagram. For example, in order to create the Hazard Relation Diagram from Figure 5-2, it is checked whether all partial mitigations reference the same hazard, and whether that hazard is part of the Functional Hazard Analysis result. According to (6.47), $CM^{Mitigation\ for\ Hazard\ H1}$ contains only one single partial mitigation $pm_1^{Mitigation\ for\ Hazard\ H1}$.

Appending Hazard Relation Diagram Extensions

The partial mitigation references a hazard that has been identified by subjecting the same activity diagram containing hazard-inducing requirements (i.e. $actD^{pm}$, see (6.35)) to functional hazard analyses (see (6.22) and (6.23)) that is also referenced in the partial mitigation (see (6.54)). Therefore, the pseudo-code will allow for performing the append operations in the following, as shown in the following Listing 7. In this listing, pseudo-code to create the components of Hazard Relation Diagrams by performing the appending actions $append^{AD}$, $append^h$, $append^{sg}$, $append^{tc}$, $append^{hr}$, and $append^{CM}$ from (6.75). As can be seen, in line 4, the hazard-mitigating requirements are added from the input parameters from Listing 6. Since this must be a set of activity diagrams, the hazard-mitigating requirements are depicted using the notational elements of UML activity diagrams such that Well-Formedness Rule 8 is satisfied. Subsequently, in line 6, the hazard that is referenced in the partial mitigations is added to the Hazard Relation Diagram. Since Listing 6 has ensured that all hazards referenced in the partial mitigations are the same, Listing 7 line 4 simply takes the hazard referenced by the first partial mitigation in CM^h , thereby satisfying Well-Formedness Rule 1. In Lines 5 to 13, the safety goals and trigger conditions specific to the hazard that was appended to the Hazard Relation Diagram in line 4 are appended as well, thereby fulfilling Well-Formedness Rule 2 to Well-Formedness Rule 4. This is done by looking up the hazard from line 4 in the hazard analysis results passed as an argument to Listing 6, retrieving the safety goal and the trigger condi-

tions, respectively, and storing them in a local variable. Moreover, since for the trigger conditions in $fha(actD^{fr})$, (6.20) holds, thereby fulfilling Well-Formedness Rule 5 to Well-Formedness Rule 7. After all contextual information about the hazard from line 4 are added to the Hazard Relation Diagram, the pseudo-code in Listing 7, lines 14 to 19 creates a mitigation partition for each partial mitigation in CM^h such that all inserted or substituted elements are subsumed by the mitigation partition. This way, Well-Formedness Rule 12 and Well-Formedness Rule 15 are satisfied. Lastly, exactly one Hazard Relation is created in Listing 7, line 21, thereby fulfilling Well-Formedness Rule 16.

```

1  --- assign the hazard-mitigating requirements
2  let  $AD^{hazRD} = AD^{hmr}$ 
3  --- assign the mitigated hazard from  $CM^h$  from Listing 6
4  let  $h^{hazRD} = haz$ 
5  --- assign safety goal and trigger conditions specific to  $h^{hrd^h}$ 
6  let  $tc^h = \varepsilon$ 
7  let  $sg^h = \varepsilon$ 
8  foreach  $res \in fha(actD^{fr})$  {
9      if  $res = haz$  {
10          $sg^h = sg^{haz} | sg^{haz} \in_t res$ 
11          $tc^h = tc^{haz} | tc^{haz} \in_t res$ 
12     }
13 }
14 --- create mitigation partitions for each partial mitigation
15 let  $CM^{hazRD} = \emptyset$ 
16 foreach  $pm_i^h \in CM^h$  {
17     let  $part = pm_i^h \setminus R^{actD} | R^{actD} \in_t pm_i^h \wedge actD \in_t pm_i^h s$ 
18      $CM^{hazRD} = CM^{hazRD} \cup (part)$ 
19 }
20 --- create Hazard Relation
21 let  $hr^{hazRD}$ 
22

```

Listing 7 Pseudo-Code Signature $append^X$ of the QVTo Script q^{hrd} .

Example

The result of applying Listing 7 to the example of hazard H1 from Table 2-1 is shown in Figure 6-8. As can be seen, the hazard from (6.23), its safety goal and the trigger conditions (see (6.21)) have been appended to the hazard-mitigating requirements from Figure 6-7. The Hazard Relation has been inserted and a mitigation partition was created for the partial mitigation in (6.47).

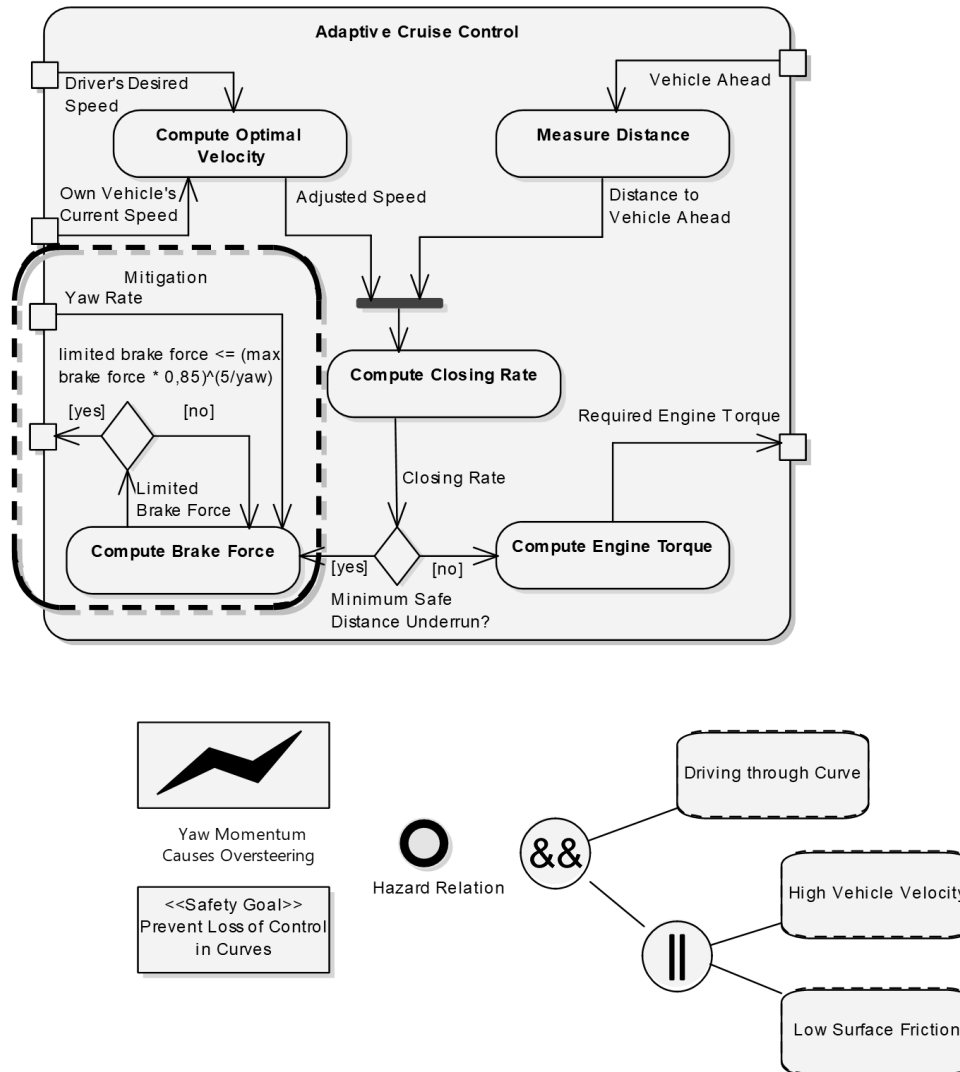


Figure 6-8 Preliminary Hazard Relation Diagram without Appended Hazard Associations.

Appending Hazard Associations

When Listing 7 completes execution, all Hazard Relation Diagram components of have been created with the exception of the set of hazard association HA^{hrd^h} . This is done in the below pseudo-code Listing 8. According to Well-Formedness Rule 17, there must be at least four hazard associations: Between the hazard, the safety goal, the trigger conditions, and at least one mitigation partition, respectively, and the Hazard Relation. The hazard associations between the hazard, the safety goal, and the trigger conditions are created in Listing 8 in lines 4, 6, and 9, respectively. It is to note, however, that the hazard association between the trigger conditions and the Hazard Relation must be made with the top-most element of the binary trigger condition tree, which is enforced in line 9 as well. Since Hazard Relation Diagrams can contain any number of mitigation partitions (however, at least one, see Well-Formedness Rule 12), all mitigation partitions must be associated with the Hazard Relation. This is done in lines 12 to 16, where for

each mitigation partition (see Listing 7), a hazard association to the Hazard Relation is created and appended to the set of all hazard associations. When line 16 completes execution, Well-Formedness Rule 17 is satisfied. All components are then appended into a tuples denoting the complete Hazard Relation Diagram (line 18), which is subsequently returned by the script.

```

1  --- create hazard associations from each Hazard Relation Diagram
2  --- component to the Hazard Relation.
3  --- first, create hazard association between hazard and Hazard Relation
4  let  $ha_1^{hazRD} = (h^{hazRD}, hr^{hazRD})$ 
5  --- second, create hazard association between safety goal and Hazard Relation
6  let  $ha_2^{hazRD} = (sg^h, hr^{hazRD})$ 
7  --- third, create hazard association between top-most
8  --- trigger condition and Hazard Relation
9  let  $ha_3^{hazRD} = (tc^h, hr^{hazRD})$ 
10 --- subsume all three hazard association into one set of Hazard Relations
11 let  $HA^{hazRD} = \{ha_1^{hazRD}, ha_2^{hazRD}, ha_3^{hazRD}\}$ 
12 --- fourth (to n-th), create hazard association between every mitigation
13 --- partition and the Hazard Relation and add to set of Hazard Associations
14 foreach  $pm_i^h \in CM^{hazRD}$  {
15     let  $HA^{hazRD} = HA^{hazRD} \cup ha_{tmp}^{hazRD} | ha_{tmp}^{hazRD} = (pm_i^h, hr^{hazRD})$ 
16 }
17 --- create a Hazard Relation Diagram tuple from all components
18 let  $hrd = "HRD for Hazard" + h^{hazRD}$ 
19  $hazRD = (hrd, AD^{hazRD}, h^{hazRD}, T^h, sg^h, CM^{hazRD}, hr^{hazRD}, HA^{hazRD})$ 
20 return  $hazRD$ 
21 } .

```

Listing 8 Pseudo-Code Signature $append^{HA}$ of the QVTo Script q^{hrd} .

Result of QVTo Script Application

Just like with q^{hmr} , Listing 6 to Listing 8 form the complete pseudo-code script for q^{hrd} , which yields a Hazard Relation Diagram fulfilling all well-formedness rules. For example, when passing the hazard-mitigating requirements from Figure 6-7, the mitigation template from Table 6-3, as well as the Functional Hazard Analysis results from Table 2-1 as an input, the result is the Hazard Relation Diagram from Figure 5-2, which corresponds to (6.53).

6.4. Summary of the Satisfaction of Well-Formedness Rules

If the approach presented in Section 6.3 is used to create Hazard Relation Diagrams all well-formedness rules defined in Section 5.5 are satisfied. In Section 6.3, satisfaction of individual well-formedness rules have already been explained. In the following Table 6-4, we summarize the satisfaction of these rules. In this table, the “line” refers to the line in the respective listing, in which the pseudo-code specific for the respective well-formedness rule can be found.

Table 6-4 Summary of Well-Formedness Rules and their Satisfaction.

Rule No.	Description	Listing, Line(s)
1	A Hazard Relation Diagram contains exactly one hazard documented in hazard analysis worksheets.	Listing 7, Line 4
2	A Hazard Relation Diagram contains exactly one safety goal documented in hazard analysis worksheets.	Listing 7, Lines 5 to 13
3	The safety goal in a Hazard Relation Diagram is specific to the conceptual mitigation of the hazard depicted in the Hazard Relation Diagram.	
4	A Hazard Relation Diagram contains the trigger condition identified during hazard analyses that are document the Boolean conditions from the operational context in which the hazard occurs.	
5	The trigger conditions are represented in a Hazard Relation Diagram in a tree structure.	Listing 7, Line 11
6	The nodes of the binary trigger condition tree must be either a Trigger Condition Conjunction or a Trigger Condition Disjunction and there must be at most two leafs.	
7	The leafs of the binary trigger condition tree must be atomic states from the operational context of the system under development identified during hazard analyses or the root for a subtree.	
8	The hazard-mitigating requirements depicted in a Hazard Relation Diagram are documented using the notational elements of UML activity diagrams.	Listing 7, Line 2
9	There are no unconnected (“dangling”) activity edges in a Hazard Relation Diagram.	Listing 5, Lines 1-6.
10	There are no unconnected (“orphaned”) activities, control nodes, or pins in a Hazard Relation Diagram.	Listing 5, Lines 22-35; Lines 36-49; Lines 49-63
11	There are no cliques of circularly connected, but otherwise unconnected activities and control nodes in a Hazard Relation Diagram.	Listing 5, Lines 7-20,
12	A Hazard Relation Diagrams contains exactly one conceptual mitigation documented in at least one mitigation partition.	Listing 7, Lines 14-19
13	A Hazard Relation Diagrams contains all mitigation partitions pertaining to the same conceptual mitigation.	Listing 1, Lines 14-20
14	The conceptual mitigation depicted in a Hazard Relation Diagram is specific to the hazard depicted in the Hazard Relation Diagram.	Listing 6, Lines 19-24
15	The mitigation partitions depicted in a Hazard Relation Diagram subsume all hazard-mitigating requirements specific to the hazard depicted in the Hazard Relation Diagram.	Listing 7, Lines 17-18
16	A Hazard Relation Diagram contains exactly one Hazard Relation.	Listing 7, Line 21
17	The Hazard Relation contained in a Hazard Relation Diagram is associated with the hazard, the safety goal, the top-most element of the binary trigger condition tree, and all mitigation partitions depicted in the Hazard Relation Diagram.	Listing 8, Lines 1-16

It is to note, that the manual activity of conceiving a mitigating and the automatic activity of creating hazard-mitigating requirements could in principle be done simultaneously by manually modeling a new hazard-mitigating activity diagram directly. However, in that case, Well-Formedness Rule 13 and Well-Formedness Rule 15 must be ensured manually.

7. Tool Support for Hazard Relation Diagrams

In this chapter, we discuss the tool support for Hazard Relation Diagrams. Section 7.1 presents an overview over the purpose of the developed tools and rationales for the chosen technologies. Section 7.2 presents the profile for Enterprise Architect and Section 7.3 presents the tool prototype for the creation of Hazard Relation Diagrams.

7.1. Purpose and Rationales of Technology Choices

The purpose of the tool support for Hazard Relation Diagrams is to provide stakeholders with the ability to create Hazard Relation Diagrams during development. This can in principle be done in two different ways: manually, using a suitable modeling tool, and automatically, using the approach from Chapter 5.5. In both cases, it is necessary to implement the modeling concepts of Hazard Relation Diagrams from Chapter 5. To do so, the following two tool support components were implemented:

- **Hazard Relation Diagram Profile for Enterprise Architect.** Enterprise Architect (see [Enterprise Architect]) is a UML-compatible modeling tool. It was chosen to implement a Hazard Relation Diagram profile due to its profile extension mechanism. The extension mechanism allows implementing conceptual UML profiles, like the ontological foundation of Hazard Relation Diagrams from Section 5.1, into technical profiles. This allows simple implementation of Hazard Relation Diagrams into Enterprise Architect such that Hazard Relation Diagrams can be easily modeled manually by stakeholders during development. The Hazard Relation Diagram Profile for Enterprise Architect is described in Section 7.2.
- **A Tool Prototype to Create Hazard Relation Diagrams for Eclipse.** The tool prototype was implemented using Eclipse UML2 tools [Eclipse UML2 Tools]. Eclipse UML2 tools provides an Eclipse Modeling Framework based implementation of the UML2 specification [OMG 2015b] as well as the Meta Object Facility [OMG 2015a]. It does not only provide the graphical representations of UML modeling components, it also allows enforcing syntactic and semantic rules of UML. Furthermore, Eclipse UML2 tools can be used alongside other Eclipse plugins, such as Eclipse Papyrus [Eclipse Papyrus]. For Eclipse Papyrus, Eclipse an implementation of OMG's Query/View/Transformation Operational Mappings language [OMG 2016] is available [Eclipse QVT v3.5.0], which provides an environment to execute implemented QVTo scripts. Section 7.3 describes the tool prototype to create Hazard Relation Diagrams.

7.2. Hazard Relation Diagram Profile

In this section, the Hazard Relation Diagram Profile for Enterprise Architect is described. Section 7.2.1 describes the systematic approach for creating the profile. Section 7.2.2 describes the resulting profile⁵.

7.2.1. Systematic Definition of the Conceptual Profile

In the context of UML/SysML, the term “profile” describes a conceptual extension of the language features defined in the Meta Object Facility [OMG 2015a]. In order to implement conceptual UML/SysML profiles into tools, the conceptual UML/SysML profile must be implemented in a technical profile using the mechanisms provided by the implementation tool. In order to achieve this, the systematic approach proposed in [Lagarde et al. 2007; Lagarde et al. 2008] was tailored and applied (see [Tenbergen et al. 2013] for a more detailed discussion of the tailored approach). The approach consists of the following steps:

- 1. Step 1: Identification and Documentation of the relevant Modeling Concepts.** In this step, the modeling concepts of Hazard Relation Diagrams that must be implemented in the profile are identified and the relationships with one another are documented. The purpose is to ensure that the profile is going to be complete and consistent with regard to the ontological foundations of Hazard Relation Diagrams from Section 5.1.
- 2. Step 2: Quality Assurance and Completion of the Profile Ontology.** This step entails refining associations between the profile components are refined into those relationships specified by the well-formedness rules of Hazard Relation Diagrams (see Section 5.5). The purpose of this step is to ensure that the profile is adequate with regard to the visual notation and the syntactic relationships of Hazard Relation Diagrams.
- 3. Step 3: Transposing the Profile Ontology into a Preliminary Profile Metamodel.** In this step, the profile ontology further refined into a preliminary profile metamodel for the conceptual Hazard Relation Diagram profile. This step entails importing existing UML modeling concepts and UML packages that is reused in the profile.
- 4. Step 4: Refining the Preliminary Profile Metamodel into the final Conceptual Profile Metamodel.** In the final step, the preliminary profile metamodel is completed by allocat-

⁵ The Hazard Relation Diagram Profile for Enterprise Architect as well as a guide to install and use the profile is available at <http://goo.gl/MdxJie>.

ing appropriate Meta Object Facility metaclasses to the Hazard Relation Diagram modeling elements, thereby creating the profile’s stereotypes. This entails that for every profile element that denotes a visual notational element in Hazard Relation Diagrams and that cannot make use of imported UML packages, appropriate metaclasses from the Meta Object Facility must be found and inherited from in accordance with the ontological foundations of Hazard Relation Diagrams (see Section 5.1).

The resulting conceptual profile for the technical implementation is shown in Figure 7-1.

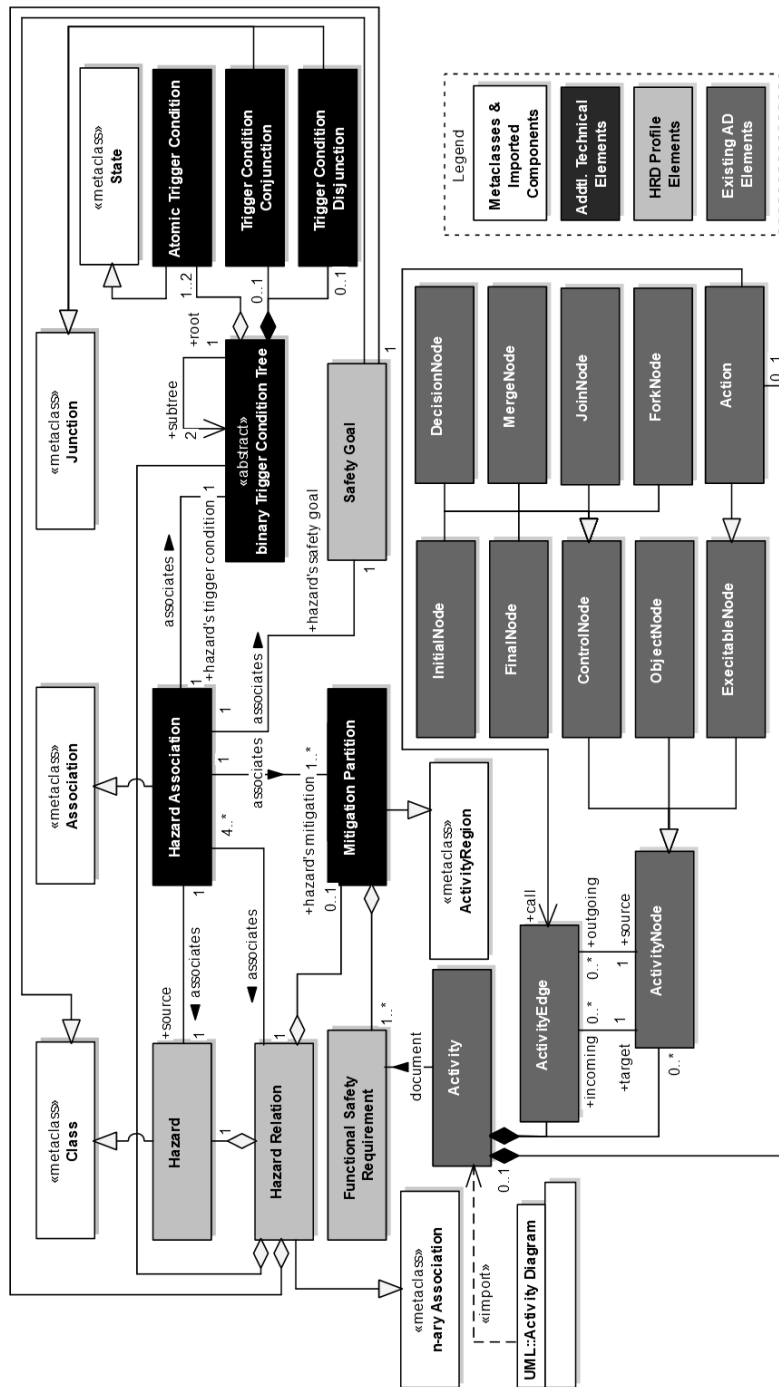


Figure 7-1 Conceptual Metamodel of the Hazard Relation Diagram Profile.

As can be seen by the white boxes, the stereotype “Hazard Relation” specializes the metaclass “n-ary Association.” Similarly, the stereotype “Hazard Association” now inherits from the UML metaclass “Association.” The stereotypes “Hazard” and “Safety Goal” specialize the metaclass “Class,” while atomic trigger conditions inherit from the UML metaclass “State.” Since there is no direct correspondence between trigger condition disjunctions, trigger condition conjunctions and some UML metaclass, the metaclass “Junction” has been used as a close approximation with regard to allowable static and dynamic associations [OMG 2015b]. The stereotype “Mitigation Partition” inherits from the UML metaclass “ActivityRegion” due to the similar visual representation.

7.2.2. Description of the Technical Profile for Enterprise Architect

Based on the conceptual profile from Section 7.2.1, the technical profile for Hazard Relation Diagrams was implemented into Enterprise Architect. Specifically, each implemented conceptual profile in Enterprise Architect consists of the following three packages.

The “diagram profile” package trivially defines a new technical Enterprise Architect metaclass representing Hazard Relation Diagrams as a new diagram type. This metaclass inherits from Enterprise Architect’s built-in metaclass representing the diagram type “Diagram_Dynamic,” which represents UML activity diagrams. This way, Hazard Relation Diagrams are established as an extension of UML activity diagrams within Enterprise Architect.

The “toolbox profile” package defines the user interface to allow users of Enterprise Architect to place Hazard Relation Diagram modeling elements onto the newly defined diagram type. For this purpose, element types and association types of UML activity diagrams and Hazard Relation Diagrams, respectively, are represented by technical metaclasses within the package. These metaclasses are then aggregated to the built-in metaclass “ToolboxPage.”

The “Hazard Relation Diagram Profile” package defines the stereotypes, association types, and dependencies from the conceptual UML/SysML profile from Section 7.2.1 into Enterprise Architect (see Figure 7-1). In order to integrate them into Enterprise Architect, it was necessary to define new classes for each modeling concept and inherit from the appropriate built-in Enterprise Architect metaclass representation of the Meta Object Facility element. This yields the technical profile for Hazard Relation Diagrams. In contrast to Figure 7-1, where the stereotype “Hazard Relation” only inherits from the metaclass “n-ary Association,” implementing this relationship in the technical profile is insufficient, as Enterprise Architect strictly enforces the separation between static and dynamic UML diagram types [SparxSystems 2014]. To accommodate this limitation, a second generalization was added for the stereotype “Hazard Relation”

such that it also inherits from the metaclass “MergeNode,” thereby implementing the semantics of the n-ary association (see Section 5.1) with the syntax of dynamic UML diagrams [Rumbaugh et al. 2004]. Furthermore, each stereotype defines the attributes “_image” and “_metatype.” These attributes are built-in tagged values to be added to the custom stereotypes [OMG 2015a] of Enterprise Architect [SparxSystems 2014]. The attribute “_metatype” can be used to define the human-readable name of the stereotype, while the attribute “_image” can be used to define the specific visual appearance of the stereotype (see Section 5.3) using built-in Enterprise Architect shape scripts.

7.3. Tool Prototype to Create Hazard Relation Diagrams

In Section 6.3, we have presented the approach to create Hazard Relation Diagrams. The approach consists of two steps:

1. **Create hazard-mitigating requirements** from activity diagrams containing functional requirements. In this step, the functional requirements of the system under development are changed by executing transformation steps documented in mitigation templates. This is done for every activity diagram, in which the conceptual mitigation must be integrated (cf. Cases 2 and 3 from Section 5.4). The result is a collection of activity diagrams containing hazard-mitigating requirements making up a partial mitigation for a hazard.
2. **Create Hazard Relation Diagrams** by appending contextual information about the hazard (i.e. the hazard, its trigger conditions, and its safety goal) to the hazard-mitigating requirements from the previous step. The necessary dependencies to achieve this are specified in the mitigation template. Since the mitigation templates correspond to the mitigation partitions to be included in the Hazard Relation Diagram, in this step, the Hazard Relation as well as all hazard associations are appended as well. The result is a completed Hazard Relation Diagram for one conceptual mitigation specific for one hazard.

In order to implement the tool prototype, it was necessary to implement the QVTo scripts from Section 6.3, as transformations can be executed within the runtime environment provided by the Eclipse QVT plugin. To be able to execute these scripts, several other components were also necessary. An overview over their logical structure and the functional interplay is given in Figure 7-2. As can be seen, four types of components that are part of the tool prototype: existing components provided by the Eclipse platform (dark grey classes in Figure 7-2), implemented artifact types (light grey classes in Figure 7-2), instantiated artifact types (white classes in Figure 7-2), and implemented QVTo scripts (black classes in Figure 7-2). The implemented artifact

types represent the formalized artifact types from Section 6.2. In particular, Hazard Analysis Results (see Section 6.2.2) as well as mitigation templates (see Section 6.2.3) have been implemented by as an Eclipse Ecore model by extending the Eclipse Ecore Metamodel. Furthermore, a profile for Hazard Relation Diagrams (see Section 6.2.4) has been implemented based on Eclipse Ecore as well. As can be seen, the functional requirements (see Section 6.2.1) have not been implemented. Since functional requirements are depicted as UML activity diagrams (see Section 5.1), the respective activity diagram metamodel provided by Eclipse UML2 was reused. Reused Eclipse components also include PapyrusUML, and the QVT Operational Runtime. The QVTo scripts (black classes in Figure 7-2) represent the implementation of the scripts from Section 6.3. The QVTo scripts consume the instantiated artifacts (white classes in Figure 7-2).

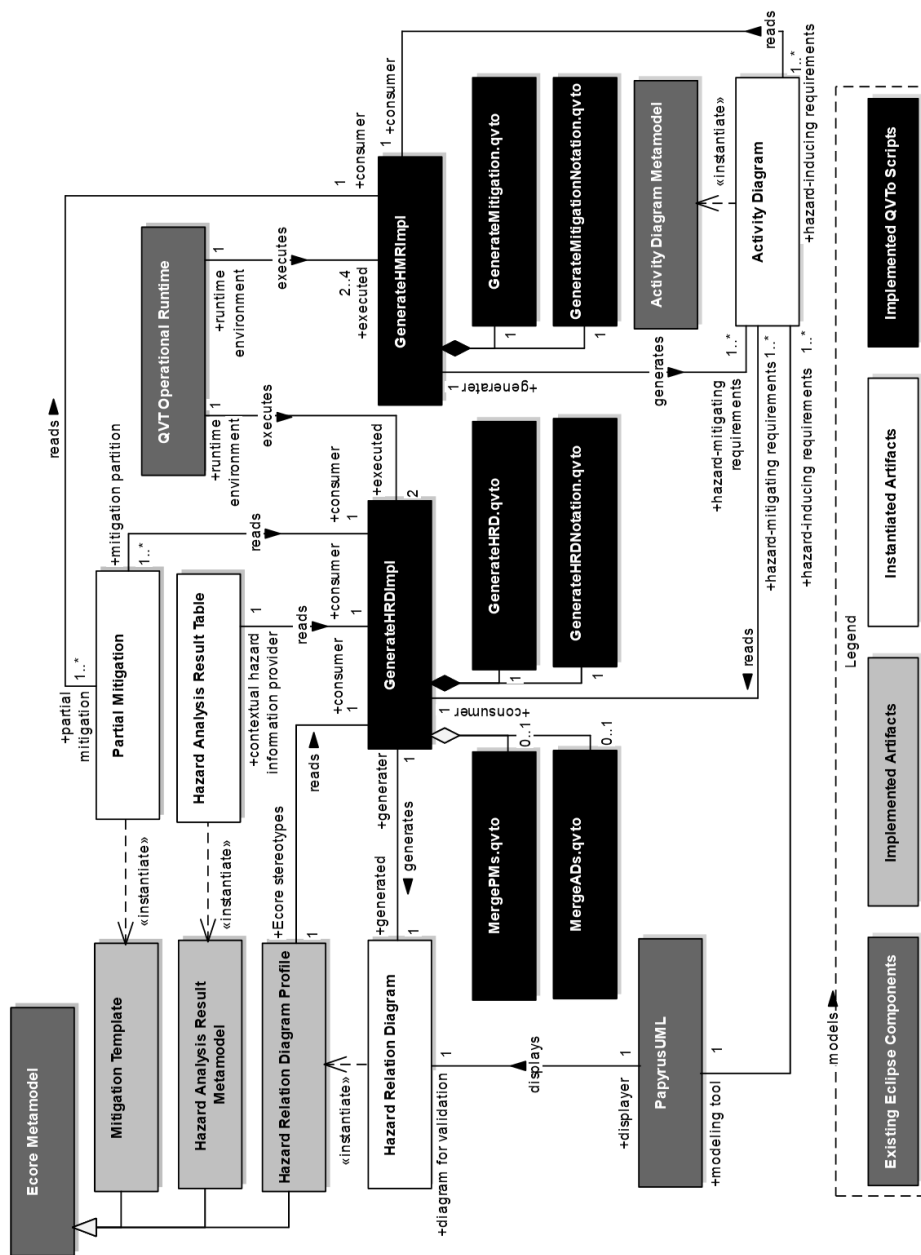


Figure 7-2 Structure and Technical Interplay of the Tool Prototype Components.

In the following Section 7.3.1 the principles of how the QVTo scripts from Section 6.3 must be implemented is explained. Section 7.3.2 presents the functional interplay of the implementation of the QVTo scripts in Section 6.3.1. Section 7.3.3 then presents the functional interplay of the implementation of the QVTo scripts in Section 6.3.2. Section 7.3.4 discusses additional support that was necessary in order to be able to create Hazard Relation Diagrams with multiple Mitigation Partitions, as described in Section 5.4⁶.

7.3.1. Principles of Implementation of the Transformation Scripts

Due to the manner in which Eclipse Ecore maintains UML diagrams, it was necessary to implement the creation scripts from Section 6.3 using a collection of six QVTo scripts (black elements in Figure 7-2). This is because in Eclipse, a graphical diagram is part of a file structure representing a “model.” An Eclipse model consists of three parts:

- a **.uml file** describing the conceptual elements, relationships, and stereotypes in the model,
- a **.notation file** describing the visual representation of the conceptual model elements, and
- a **.di file** which associates the visual representation and the conceptual elements to a common diagram.

It is hence sufficient to apply the scripts outlined in Section 6.3 to only the .uml file. This would not yield a corresponding visual representation. Therefore, each QVTo transformation must be applied twice, once on the .uml file and once equivalently on the .notation file, which requires individual implementations of the script for the .uml file and not .notation file as well.

7.3.2. Implementation of the QVTo Scripts to Create Hazard-Mitigating Requirements

Figure 7-3 shows the functional interplay between the implementation of the approach to create hazard-mitigating requirements (see Section 6.3.1) and the required input and output artifacts. As can be seen, the first QVTo script to be executed is “GenerateMitigation.qvto.” In accordance with in Section 6.3.1, the script implementation takes two inputs: a partial mitigation documented using a mitigation template (see Section 6.2.3) as well as the .uml file of some activity diagram containing hazard-inducing requirements. The script will execute all transformation steps specified in the partial mitigation.

⁶ Implemented and executable copies of the QVTo scripts are available at <http://goo.gl/yfjMy9>.
The tool to automatically create Hazard Relation Diagrams is available at <http://goo.gl/MdxJie>.

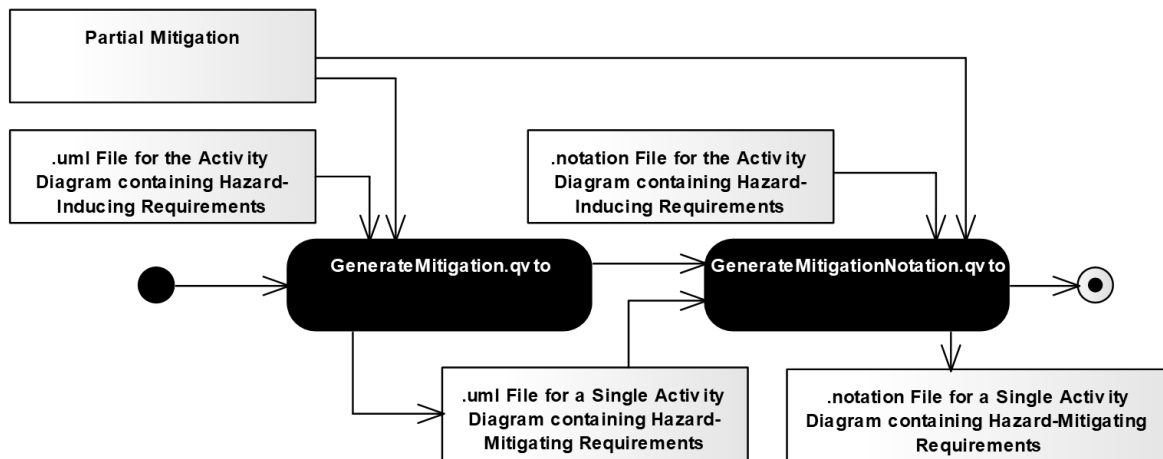


Figure 7-3 Functional Interplay of “GenerateMitigationImpl” from Figure 7-2.

The output of “GenerateMitigation.qvto” is an .uml file containing the hazard-mitigating requirements specified in the partial mitigation. Once the .uml file is created, the script implementation “GenerateMitigationNotation.qvto” is applied using the same partial mitigation as well as .notation file for the same activity diagram containing hazard-inducing requirements that were input for “GenerateMitigation.qvto.” As outlined in Section 7.3.1, this scripts enacts the same transformations specified in the partial mitigation on the visual representation of the diagram. The output of “GenerateMitigationNotation.qvto” is a .notation file corresponding to the .uml file for the activity diagram containing hazard-mitigating requirements.

7.3.3. Implementation of the QVTo Scripts to Create Hazard Relation Diagrams

In Figure 7-4 the functional interplay between QVTo scripts and artifacts to create a Hazard Relation Diagram with one mitigation partition is shown. The process in Figure 7-4 creates Hazard Relation Diagrams that correspond to Case 1 in Table 5-3.

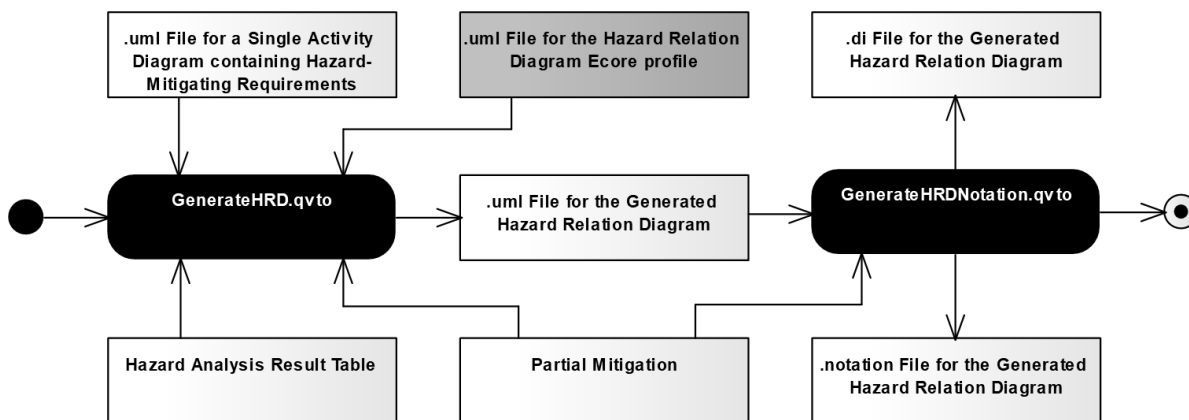


Figure 7-4 Functional Interplay of “GenerateHRDImpl” from Figure 7-2.

The QVTo script to create the Hazard Relation Diagram’s .uml file “GenerateHRD.qvto” takes three inputs: the .uml file of the activity diagram containing the hazard-mitigating requirements

that was created using the process depicted in Figure 7-3, a partial mitigation according to which the activity diagram containing the hazard-mitigating requirements was created, and a hazard analysis result table. However, since this the output to be created must conform to the Hazard Relation Diagram stereotypes, “GenerateHRD.qvto” requires a reference to the .uml file for the Hazard Relation Diagram Ecore profile as a fourth input. The QVTo script will append the contextual hazard information from the hazard analysis result table to the activity diagram and create a mitigation partition for the hazard-mitigating requirements specified in the partial mitigation. The output is an .uml file for the created Hazard Relation Diagram. The notational elements must subsequently be defined in correspondence with the stereotypes in the .uml file for the Hazard Relation Diagram. Therefore, the QVTo script “GenerateHRDNotation.qvto” will take the .uml file for the Hazard Relation Diagram as well as the partial mitigation as input and produce the corresponding .notation file. However, in contrast to the process depicted in Figure 7-3, the input diagram is not changed, but a new diagram (including .di file) is created.

7.3.4. Creating Hazard Relation Diagrams with Multiple Mitigation Partitions

Using the process depicted in Figure 7-3, Hazard Relation Diagrams which contain one mitigation partition and one activity diagram can be created. Such Hazard Relation Diagrams correspond to Case 1 in Table 5-3. In order to create Hazard Relation Diagrams with two or more mitigation partitions and one activity diagram (Case 2 in Table 5-3), or two or more mitigation partitions and two or more activity diagrams (Case 3 in Table 5-3), two additional QVTo scripts had to be implemented:

- **mergePMs.qvto** creates one mitigation partition within a Hazard Relation Diagram for each mitigation template that was specified, and
- **mergeADs.qvto** merges multiple individual activity diagrams into the same Hazard Relation Diagram.

To create a Hazard Relation Diagram according to Case 3 in Table 5-3, these two scripts take the .uml files of two activity diagrams containing hazard-mitigating requirements as input and merges the activity diagrams specified therein into a new .uml file for the merged activity diagrams. This new .uml file for the merged activity diagrams is used as input for the process shown in Figure 7-4. In order to create a Hazard Relation Diagram according to Case 2 in Table 5-3, the QVTo script “mergeADs.qvto” can simply be left out. If more than two activity diagrams shall be merged, the process must be repeated.

8. Discussion of the Solution Approach

In Part II, we have presented Hazard Relation Diagrams and their creation. In this section, we discuss the our solution

Execution Order of Transformation Steps

As illustrated in the example in Figure 6-5, the execution order of transformation steps directly impacts the well-formedness of activity diagrams. For example, if an activity edge is inserted, but the target activity hasn't been inserted yet, the insertion of the activity edge might fail. The script proposed in Section 6.3.1 does not prescribe or restrict the order in which transformation steps have to be specified or executed. As a consequence, it is possible for the transformation steps to be defined in the mitigation templates in such an order, that their execution results in syntactically invalid activity diagram and/or incomplete hazard-mitigating requirements.

Since we currently do not provide any support to check the correct order of the transformation steps, it is the responsibility of the human stakeholder to define a correct order, which yields well-formed activity diagrams and complete hazard-mitigating requirements (with regard to the conceptual mitigation). The stakeholder can define the order of the execution of the steps by using the ID attribute of the mitigation template depicted in Table 6-2.

Conflicts within Transformation Steps

The script presented in Section 6.3.1 does not consider conflicts within the specified transformation steps. For example, if the stakeholder specifies an insertion operation of an input pin and, at a later point in the execution sequence, specifies a removal operation of the previously inserted input pin, the two transformation steps effectively cancel each other out. This obviously results in semantically incorrect hazard-mitigating requirements for a conceptual mitigation. Albeit such conflicts might be detected during validation, it remains the responsibility of the stakeholders to specify conflict-free transformation steps.

Optimization

When realizing the tool prototype we did not put in any effort in optimizing the execution of the scripts defined in Section 7.3.

Well-Formedness of Activity Diagrams

The script presented in Section 6.3.1 does not enforce syntactically valid activity diagrams. Albeit Well-Formedness Rule 9 through Well-Formedness Rule 11 provide some consideration of syntactic validity of the activity diagrams containing hazard-mitigating requirements, these well-formedness rules merely pertain to errors in notation stemming from insertion, removal, or substitution of modeling elements. For example, it is still possible that the stakeholder specifies the insertion of a join node, but neglects to specify any incoming activity edges for the inserted join node. The implication is a syntactically incorrect activity diagram containing semantically incorrect hazard-mitigating requirements. Albeit such issues might be detected during validation, it is the responsibility of the stakeholder to specify transformation steps yielding syntactically valid activity diagrams and the responsibility of the implementation to conduct syntactic validity checks (see also Section 6.3.1) and involve the stakeholder in order to remedy syntactic incorrectness.

Layouting of Activity Diagrams and Hazard Relation Diagrams

The tool support does not support layouting of the created diagrams. However, the stakeholder can change the layout of the diagram using our prototype tool.

Part III: Empirical Evaluation

9. Overview

In this part, Hazard Relation Diagrams are empirically evaluated. To foster reproducibility and comparability to other studies, we adhere to the structure proposed in [Jedlitschka et al. 2008], as outlined in the following subsections.

9.1. Experimental Design

The empirical evaluation consists of four experiments, which were designed in order to investigate the impact of Hazard Relation Diagrams on the challenges described in Section 1.2. Research questions are derived and systematically refined into an evaluation strategy, hypotheses, and metrics. Chapter 10 outlines the experimental design in detail. In particular, purpose and differences between the four experiments are explained in order to increase comparability between the experiments.

9.2. Experimental Results

The experimental results are presented in Chapter 11. The acceptance and rejection criteria for each hypotheses in each experiment are stated and the results of the data analyses are presented. The implications of the results are discussed with regard to each hypotheses.

9.3. Threats to Validity

The chosen evaluation strategy, experimental design, as well as data preparation and analysis procedure may have impacted the results of the experiments presented in Chapter 11. In order to be able to draw valid conclusions from the data, Chapter 13 discusses the threats to validity of the experimental design, the measures taken to mitigate these threats, and remaining threats.

9.4. Discussion of the Results

Based on the threats to validity, we critically discuss the results of the empirical evaluation. We summarize the results for each hypotheses across the experiments and draw conclusions for each research question with regard to the impact of Hazard Relation Diagrams on the challenges from Section 1.2.

10. Experimental Design

In Section 10.1 we define research questions based on the challenges outlined in Section 1.2 and define a suitable evaluation strategy consisting of four experiments (see Section 10.2.). Section 10.3 presents the stimuli used in all experiments. Section 10.4 outlines the hypotheses derived from the research questions and discusses differences in their investigation across the four experiments. Sections 10.5 through 10.8 outline the experimental procedure for each experiment. These procedures were partly reported in [Tenbergen et al. 2017]. The data preparation procedure is explained in Section 10.9. Section 10.10 discusses participant experience levels in all experiments to support the comparability of the experimental results.

10.1. Research Questions

It was argued throughout this dissertation that using Hazard Relation Diagrams during validation of hazard-mitigating requirements improves stakeholders' ability to:

- ascertain how specific hazards have been mitigated (Challenge 1),
- ascertain how hazard-mitigating requirements avoid, control, or reduce the occurrence of trigger conditions for a specific hazard (Challenge 2), and
- differentiate hazard-mitigating requirements from other functional requirements that are not part of the conceptual mitigation for a specific hazard (Challenge 3).

We define the following research questions:

- **RQ1:** What is the impact on the adequacy judgements when validating hazard-mitigating requirements using Hazard Relation Diagrams?
- **RQ2:** What is the impact on validation efficiency and effectiveness when using Hazard Relation Diagrams?

Albeit manual reviews are the most frequently applied requirements validation technique [Flynn and Warhurst 1994], inspection-like group reviews have received a considerable amount of attention in the literature (see, e.g., [Aurum et al. 2002]). We therefore define the following research question:

- **RQ3:** What is the impact on the validation technique when validating hazard-mitigating requirements using Hazard Relation Diagrams?

To investigate research questions RQ1 and RQ2, a number of experimental hypotheses have been defined. These are discussed in detail in Section 10.4. To investigate research question RQ3, the empirical evaluation consists of four experiments that differ in validation technique

as well as experimental design type was conducted. The following Section 10.1 outlines the experimental strategy in this respect.

10.2. Evaluation Strategy

The experimental evaluation to investigate the research questions from Section 10.1 consisted of two between-subjects experiments and two within-subjects experiments for a total of four experiments. Table 10-1 summarizes the experimental configurations.

Table 10-1 Summary of the Experimental Configuration in the Four Experiments.

Property	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Validation Technique	Individual Reviews	Fagan Inspections	Individual Reviews	Fagan Inspections
Design Type	Between-Subjects		Within-Subjects	
Groups	2		1	2
Conditions	1 per Group		2 per Participant	
No. of Participants (see Section 10)	137	40	31	52
Relative Experience	3rd Semester Undergraduate	Graduate Students	5th Semester Undergraduate	5th Semester Undergraduate and Graduate
Experimental Stimuli	10 conceptual mitigations in either 10 activity diagrams with FHA excerpts (control group/condition) or 10 Hazard Relation Diagrams (control group/condition)			

In order to investigate research question RQ3, Experiment 1 investigates RQ1 and RQ2 using reviews as the validation technique, while Experiment 2 investigates RQ1 and RQ2 using Fagan Inspections. Both experiments were conducted first and a between-subjects experiments was chosen in an effort to minimize training overhead. In order to increase confidence in and generalizability of the results, both experiments were repeated using a within-subjects design, i.e. in Experiment 3 and 4. In consequence, Experiments 1 and 3 investigate the impact of Hazard Relation Diagrams in the sense outlined in RQ1 and RQ2 using reviews as the validation technique, while Experiments 2 and 4 investigate the impact of Hazard Relation Diagrams in the sense outlined in RQ1 and RQ2 using Fagan Inspections.

To maintain comparability of results regarding RQ1 and RQ2 and to be able to draw conclusions regarding RQ3, the same experimental stimuli are used in all experiments and an experimental design was pursued that only differed with regard to the validation technique. Furthermore, the data preparation and analysis procedure was kept consistent across all experiments, albeit differences in experimental design required minor deviations.

10.3. Experimental Stimuli

In all experiments, we used an excerpt of a requirements specification for the ACC system shown in Figure 2-1 as the basis for the experimental material. This entailed one activity diagram comprising five hazard-inducing requirements, for which a Functional Hazard Analysis was conducted and documented in a hazard analysis worksheet (see Table 2-1 for an excerpt). A total of ten hazards were identified during the Functional Hazard Analysis, five of which were randomly selected and *adequately* mitigated. To do so, for each hazard, a variation of the activity diagram from Figure 2-1 was derived containing hazard-mitigating requirements that mitigate the hazard during operation. This yielded five activity diagrams containing adequate hazard-mitigating requirements. The other five hazards were *inadequately* mitigated. To do so, for each hazard, a variation of the activity diagram from Figure 2-1 was derived containing hazard-mitigating requirements with semantic mistakes allowing the hazard to still occur during operation. This yielded five activity diagrams containing inadequate hazard-mitigating requirements. All ten activity diagrams were used for the control condition of the experiments (see Sections 10.5.3, 10.6.3, 10.7.3, and 10.8.3, respectively). For the treatment condition, each adequate and inadequate activity diagram was extended into a corresponding Hazard Relation Diagram. Deliberate semantic mistakes, which would not influence the safety of the ACC in operation, were included into the Hazard Relation Diagrams and activity diagrams, respectively, and some FHA safety goals for inadequately mitigated hazards were replaced with non-sense safety goals. This was done to allow for decoy defects such that for every activity diagram and Hazard Relation Diagram, at least one correct rationale would exist which pertains to diagram semantics, diagram syntax, trigger conditions, safety goal, and the conceptual mitigation itself. The introduced decoy defects were the same for treatment and control conditions. In summary, the experimental material consisted hence of ten activity diagrams for the control condition and ten Hazard Relation Diagrams for the treatment condition. The control condition furthermore included hazard analysis worksheets⁷.

10.4. Hypotheses and General Metrics

The primary aim of the experimental evaluation of Hazard Relation Diagrams is to investigate the influence of explicitly documented dependencies between hazard-mitigating requirements,

⁷ For researchers interested in replicating our experiments, the experimental material can be found at <https://goo.gl/XwJJQu>.

hazard, trigger conditions, and safety goals on validation judgement (RQ1 in Section 10.1). We assume that explicitly documented dependencies in Hazard Relation Diagrams improve validation judgments insofar that the rationales of adequacy judgements are more concerned with contextual hazard information than with other diagram properties. In the following, if a rationale mentions the conceptual mitigation, the trigger conditions, of the safety goal, we say that this rationale mentions “*contextual hazard information.*” If it mentions any other diagram properties (i.e. the diagram’s semantics or syntax), we say that this rationale mentions “*other diagram properties.*” We defined the following two-tailed hypothesis to investigate RQ1:

Hypothesis 1:

There is a difference in rationale between treatment condition and control condition regarding adequacy judgments when validating hazard-mitigating requirements.

Since Hazard Relation Diagrams integrate contextual hazard information with the functional requirements within the activity diagram, we predict that the rationale is more likely to be concerned with the hazard-mitigating requirements, the trigger conditions, or the safety goal, in the treatment condition than with diagram semantics or diagram syntax.

Challenge 2 in Section 1.2 indicates that without Hazard Relation Diagrams stakeholders’ ability to ascertain how hazard-mitigating requirements impact the trigger conditions of a hazard is impaired. It is therefore necessary to consider the impact of Hazard Relation Diagrams with regard to their ability to increase effectiveness (RQ2 in Section 10.1). Effectiveness in this sense means that stakeholders are more capable to make correct judgements with regard to the hazard-mitigating requirements’ ability to mitigate a hazard’s trigger conditions. We hence define the following two-tailed hypothesis for RQ2:

Hypothesis 2:

There is a difference in effectiveness between treatment condition and control condition when validating hazard-mitigating requirements.

We predict that using Hazard Relation Diagrams for validation leads to more correct judgements and therefore to higher effectiveness than using conventional activity diagrams and hazard analysis worksheets. For each diagram depicting an *adequate* conceptual mitigation, we recorded whether or not the participants indicated “adequate” during review. This was considered a *true positive* answer (i.e. the answer was supposed to be “adequate” and the participants

answered correctly). If participants indicated “adequate” during review of a diagram that depicted an *inadequate* conceptual mitigation, this was considered a *false positive* answer (i.e. the answer was supposed to be “inadequate” and the participants answered incorrectly). Equivalently, we recorded *true negative* answers (i.e. the answer was supposed to be “inadequate” and the participants answered correctly) and *false negative* answers (i.e. the answer was supposed to be “adequate” and the participant answered incorrectly) as well. The participants’ effectiveness can hence be determined by the amount of true positive, false positive, true negative, and false negative answers. We specifically opted not to determine effectiveness by means of the number of identified defects in the rationales, as giving at least one rationale was mandatory (see Sections 10.5.3, 10.6.3, 10.7.3, and 10.8.3, respectively) in order to correctly measure hypothesis H1, which could have led to skewed effectiveness measurements.

However, Hazard Relation Diagrams must also be evaluated with regard to their efficiency. This means that validation using Hazard Relation Diagrams ought to require at least as much time to complete the validation task as validation using conventional activity diagrams and hazard analysis worksheets (also see RQ2 in Section 10.1). If validation using Hazard Relation Diagrams takes more time than validation using conventional activity diagrams, a possible improvement in terms of rationale. To investigate these relationships, we defined the following two-tailed hypotheses to further investigate RQ2:

Hypothesis 3:

There is a difference in efficiency between treatment condition and control condition when validating hazard-mitigating requirements.

We predict that that using Hazard Relation Diagrams for validation requires comparably much times as validation using conventional activity diagrams and hazard analysis worksheets. In order to measure efficiency of reviews (i.e. Experiments 1 and 3), we measured the time needed for true positive, true negative, false positive, and false negative answers (see Sections 10.5.1 and 10.7.1). However, since in Experiment 2 and 4, multiple individuals were involved, we computed the effort in person-minutes in inspection-based experiments (see Section 10.6.1 and Section 10.8.1).

In addition to the rationales, effectiveness, and efficiency, we also investigated the individual stakeholders’ confidence in their own review judgments, as confidence further impacts the validation performance (RQ2 in Section 10.1). To account for this, we defined the following two-tailed hypotheses:

Hypothesis 4:

There is a difference in self-reported confidence between treatment condition and control condition when validating hazard-mitigating requirements.

To measure the validators' confidences, we applied the Goal/Question/Metric Method [van Solingen and Berhout 1999] in order to systematically select suitable items from the technology acceptance model version 3 (TAM3, see [Venkatesh and Bala 2008]) and the task-technology fit model (TTF, see [Goodhue 1998]), as summarized in Table 10-2.

Table 10-2 TAM3 and TTF Item Selection using a Template based on [van Solingen and Berhout 1999].

Object	Purpose	Metric for	Role	Development Situation
Hazard Relation Diagram	Evaluate	Subjective Confidence	Requirements Engineer	Validation of the adequacy of hazard-mitigating requirements.
Quality Focus	Source	Adapted TAM3 / TTF Definitions		
Perceived Usefulness	TAM3	The degree to which someone believes that using Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets to validate hazard-mitigating requirements adequacy enhances her performance.		
Self-Efficacy	TAM3	The degree to which an individual believes that she has the ability to validate hazard-mitigating requirements adequacy using Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets.		
Result Demonstrability	TAM3	The degree to which an individual believes that the validation results using Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets are tangible, observable, and communicable.		
“The Right Data”	TTF	The degree to which the needed information to validate hazard-mitigating requirements adequacy is maintained when using Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets.		
Meaning	TTF	The ease of determining what a modeling element in a Hazard Relation Diagram, activity diagram, or hazard analysis worksheet means, or what information is depicted therein.		
Presentation	TTF	The degree to which a Hazard Relation Diagram or an activity diagram with an hazard analysis worksheet is understandable.		
Training	TTF	The amount of available training in order to use Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets to validate hazard-mitigating requirements adequacy.		
Confusion	TTF	The degree to which the user is confused in using Hazard Relation Diagrams or activity diagrams with hazard analysis worksheets to validate hazard-mitigating requirements adequacy.		

To measure the quality foci of subjective confidence, we compiled a post-hoc questionnaire⁸, randomized the questions pertaining to each selected TAM3 and TTF item, and measured participant responses on a 5-point-Likert scale [1: “disagree”; 2: “somewhat disagree”; 3: “neither agree nor disagree”; 4: “somewhat agree”; 5: “agree”]. We adapted questionnaire questions to be neutral with regard to the used notations in order to minimize the validity threat of hypothesis guessing [Wohlin et al. 2012]. We calculated the degree of agreement for each quality focus by the sum of the answers for each question pertaining to the respective quality focus (adding the inverse ordinate for negated questions) and dividing the sum by the number of questions in the respective quality focus.

⁸The post-hoc questionnaire is available at <https://goo.gl/XwJJQu>

Table 10-3 lists the independent variable (IV) as well as the dependent variables (DVs) that result from the measurements pertaining to the hypotheses stated above. Please note, that the dependent variable IDs contain a placeholder “x”, which is replaced in Chapter 11 by the respective experiment number in order to be able to compare the results from each experiment more easily.

Table 10-3 General Metrics for each Hypothesis.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
-	IV	Independent	Presentation Mode used during Validation. Two Levels: Using Hazard Relation Diagram (treatment) vs. Using activity diagrams and hazard analysis worksheets (control)	-
H1.x	DV1.x		Number of rationales mentioning semantics	Ratio
	DV2.x		Number of rationales mentioning syntax	
	DV3.x		Number of rationales mentioning trigger conditions	
	DV4.x		Number of rationales mentioning safety goals	
	DV5.x	Number of rationales mentioning conceptual mitigations		
H2.x	DV6.x	Dependent	Number of true positive answers	Interval
	DV7.x		Number of true negative answers	
	DV8.x		Number of false positive answers	
	DV9.x		Number of false negative answers	
H3.x	DV10.x	time needed for true positive answers	Ratio	
	DV11.x	time needed for true negative answers		
	DV12.x	time needed for false positive answers		
	DV13.x	time needed for false negative answers		
H4.x	DV14.x	degree of agreement of questions pertaining to “Perceived Usefulness”	Ratio	
	DV15.x	degree of agreement of questions pertaining to “Self-Efficacy”		
	DV16.x	degree of agreement of questions pertaining to “Result Demonstrability”		
	DV17.x	degree of agreement of questions pertaining to “The Right Data”		
	DV18.x	degree of agreement of questions pertaining to “Meaning”		
	DV19.x	degree of agreement of questions pertaining to “Presentation”		
	DV20.x	degree of agreement of questions pertaining to “Training”		
	DV21.x	degree of agreement of questions pertaining to “Confusion”		

In the following, we will discuss the experimental designs of each individual experiment.

10.5. Experiment 1: Individual Reviews (Between-Subjects)

Experiment 1 was designed as a one-way between-subjects experiment [Wohlin et al. 2012] in order to investigate the impact of Hazard Relation Diagrams on validation results when using individual reviews to validate hazard-mitigating requirements. This experimental design was initially favored over a repeated measures design since most participants required pre-hoc introduction into the fundamentals of safety engineering, hazard analyses, and Hazard Relation Diagrams. This way, the participants were separated into a treatment group (reviews using Hazard Relation Diagrams) and a control group (reviews using conventional activity diagrams and hazard analysis worksheets). Hence, participants only had to be instructed in one, rather than

two representation forms. This study design has been validated by means of a pilot test (see [Tenbergen et al. 2015]).

10.5.1. Specific Metrics

In Experiment 1, every participant either reviewed ten Hazard Relation Diagrams or ten Activity Diagrams with respective hazard analysis results. Review judgments hence dependent on each participant's understanding of the respective diagrams. We measured the hypotheses from Section 10.4 as follows:

- **Hypothesis H1: Rationales.** In order to measure rationale objectivity, we asked participants to judge whether or not a specific hazard was either *adequately mitigated* (i.e. the hazard no longer occurs during operation), or *inadequately mitigated* (i.e. the hazard might still occur during operation). In addition, participants were asked to provide a rationale by means of a simple written statement detailing why they judged that a hazard was adequately or inadequately mitigated. During post-experimental analyses, this rationale was qualitatively analyzed and a differentiation was made whether a rationale pertained to contextual hazard information or not (see Section 10.9 for details on the qualitative analysis).
- **Hypothesis H2: Effectiveness.** In order to determine effectiveness in Experiment 1, we measured the number of correctly judged adequate conceptual mitigations (true positive answers), the number of correctly judged inadequate conceptual mitigations (true negative answers), the number of incorrectly judged adequate conceptual mitigations (false positive answers), and the number of incorrectly judged inadequate conceptual mitigations (false negative answers). Since every participant was subjected to five adequate and five inadequate conceptual mitigations, a perfect score was hence five true positive and five true negative answers.
- **Hypothesis H3: Efficiency.** Each participant's review efficiency was measured by determining the time needed in seconds. Depending if a diagram was judged correctly or incorrectly, and depending in whether or not the respective diagram contained an adequate or inadequate conceptual mitigation, the response time was aggregated into true positive, false positive, true negative, and false negative response times as well as the total time needed for all reviews. Efficiency measurements considered the time needed to review a diagram and indicate either "adequate" or "inadequate," not the time needed to write the review rationale.
- **Hypothesis H4: Self-Reported Confidence.** Since we aimed to compare subjective confidence of the treatment group with that of the control group in Experiment 1, we adapted

the definition of the original instrument items and questions pertaining thereto to reflect the notation used by the respective group.

10.5.2. Participants

Participants for Experiment 1 were recruited from an undergraduate requirements engineering course at the University of Duisburg-Essen in fall 2014. The author of this thesis was responsible for conducting the lab sections in this course. Among others, the course instructed the fundamentals of conceptual modeling in preparation of a grade-determining final exam. This experiment was part of the course and participation (in the sense that it was immediately relevant for the students' individual learning success and exam grade, see [Hart et al. 2000; Carver et al. 2008]) was mandatory. However, students were informed that albeit participation is mandatory for their own benefit, they had the option to discontinue at any point without penalty (other than the self-inflicted penalty of missing out on exam preparation).

A total of 123 students participated in Experiment 1. These were undergraduate ($n = 110$) or graduate students ($n = 13$) enrolled in Systems Engineering or Business Information Systems degree programs. Given the pilot study presented in [Tenbergen et al. 2015] yielded a sound methodology and only small changes (e.g., typos) were undertaken prior to onset of Experiment 1, pilot study participants were included in this sample of Experiment 1 (see [Thabane et al. 2010]). These were mainly members of the author's research group. The total population of Experiment 1 was hence 133 participants ($n_{\text{treatment}} = 67$, $n_{\text{control}} = 65$). Although gender and age were not assumed to impact the results of the experiment, we recorded a total of 117 male participants, sixteen female participants, and a participants' age between 18 and 36 years ($\mu = 23.9$, $\sigma = 3.66$).

10.5.3. Experimental Procedure

The experiment was implemented using the survey website SoSci Survey [SoSci Survey]. A short briefing was administered during a class session, which instructed participants on the fundamentals of safety engineering including hazard analyses, embedded software, and function modeling. The unaltered ACC specification excerpt was used during these pre-experimental briefings. In addition, the task of how to perform the reviews was briefly illustrated by means of two example hazards, which were different from the hazards depicted in the experimental material. After all students were briefed, a link to the survey website was made available and the students were given five days to complete the experiment. In order to minimize interaction

effects between participants, data collection was scheduled to take place between course meetings. The experimental procedure consisted of the following steps, as shown in Figure 10-1.

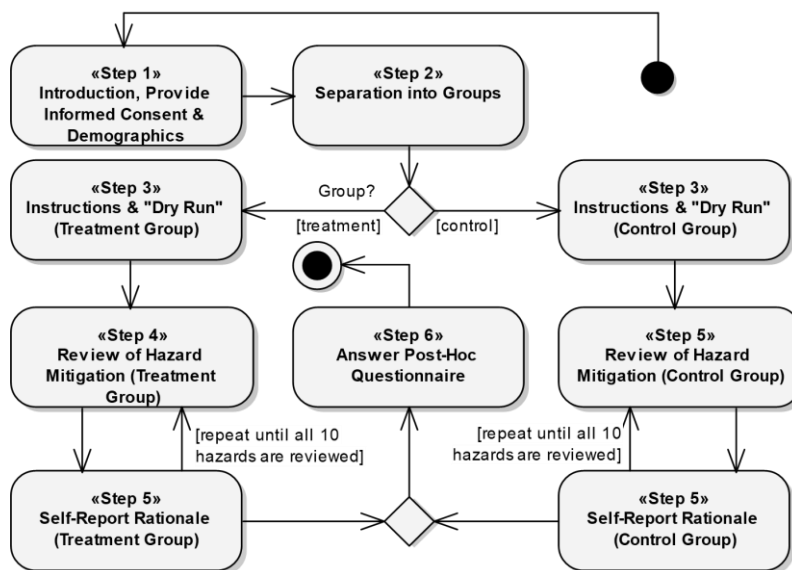


Figure 10-1 Experimental Procedure for Experiment 1.

Step 1: Introduction, Provide Informed Consent, & Demographics. Experiment 1 began with a short introduction, where informed consent as well as demographic data were collected. In this step, students were informed again of their right to discontinue participation and assured that data would be treated anonymously.

Step 2: Separation into Groups. Participants were randomly assigned to the *treatment group*, which conducted reviews using Hazard Relation Diagrams, and to the *control group*, which conducted reviews using conventional activity diagrams and hazard analysis result tables. Assignment took into consideration the participants' self-reported experience levels, such that both groups contained an approximately equal number of participants with corresponding experience levels.

Step 3: Instructions & "Dry Run." Both groups were presented with written instructions on how to review the experimental material. These instructions summarized the information given in the pre-experimental briefing. Furthermore, two example runs were performed in which the participants could rehearse the review task.

Step 4: Review of Hazard Mitigation. In this step, participants were asked to review conceptual mitigations pertaining to one randomly selected hazard. In order to reduce primacy, to recency, and carry-over effects [Wohlin et al. 2012], the sequence in which the ten activity diagrams or Hazard Relation Diagrams were presented was randomized for each participant. To ensure that both participant groups reviewed approximately equally many information items,

the control group was only shown the one row from the hazard analysis result table relevant to the hazard being mitigated in the activity diagram currently being displayed. Participants could review each activity diagram of Hazard Relation Diagram for an indeterminate amount of time. Nevertheless, response times were recorded for this step. Participants were asked to indicate “yes” if they are of the opinion that the hazard may still occur during operation and “no” otherwise. Participants were able to change their assessment as often as they wished before advancing to the next step. We recorded the reviewer confidence for this step.

Step 5: Self-Report Rationale. Participants were asked to state a brief reason why they chose “yes” or “no” in the previous step. Due to technical reasons, this rationale could not be recorded in the step as adequacy judgments and confidence of Step 4. Therefore, participants were given the opportunity to return to the previous step and change their answer if thinking about the rationale made them change their mind. The experimental material along with their decision was shown for reference.

Step 6: Answer Post-Hoc Questionnaire. After all ten conceptual mitigations were reviewed, both groups were presented with the post-hoc questionnaire in order to record participants’ subjective confidence when conducting reviews using the groups’ respective notation (see Section 10.1). The order in which questionnaire items were displayed was randomized to avoid that similar questions pertaining to the same quality focus were adjacent to one another.

10.6. Experiment 2: Fagan Inspections (Between-Subjects)

Experiment 2 focused on Fagan inspections [Fagan 1976; Fagan 1986] rather than reviews. We aimed to keep the experimental design as close to Experiment 1 as possible. We used a one-way between-subjects design and reused the experimental material from Experiment 1. However, since inspections are a group activity requiring all participants to be present at the same time, this yielded differences with regard to measurements, recruited participants, and the experimental procedure.

10.6.1. Specific Metrics in Experiment 2

For Experiment 2, we defined the following measurements:

- **Hypothesis H1: Rationales.** Like in Experiment 1, we measured differences in rationales by recording written rationales for each adequacy judgment. Rather than asking for at least one rationale from each participant, we recorded group consensus, i.e. rationales given by at least one participant that other participants agreed with (see Section 10.6.3). Like in

Experiment 1, we have qualitatively analyzed the rationales in order to determine whether a rationale pertained to contextual hazard information or not (see Section 10.5.1).

- **Hypothesis H2: Effectiveness.** We used the same measure as in Experiment 1.
- **Hypothesis H3: Efficiency.** Since the efficiency in Experiment 2 was a group effort rather than an individual effort, it was necessary to measure the time invested by the group as a whole. We did so by dividing the amount of true positive, true negative, false positive, and false negative answers for the measurements for hypothesis H2 by the effort invested by the group in person-minutes. We multiplied the result with an appropriate constant in order to map the result onto a more easily comparable interval. Albeit this is in accordance with commonly applied efficiency metrics for inspections, which, however, is sensitive to unequal group sizes [Liggesmeyer 2002]. Table 10-4 shows the differences for the dependent variables for hypothesis H3 in Experiment 2.

Table 10-4 Specific Metrics for Hypothesis H2.3 in Experiment 2.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H2.3	DV2.10	Dependent	1000 * (number of true positive answers / (number of group members * time))	Ratio
	DV2.11		1000 * (number of true negative answers / (number of group members * time))	
	DV2.12		1000 * (number of false positive answers / (number of group members * time))	
	DV2.13		1000 * (number of false negative answers / (number of group members * time))	

- **Hypothesis H4: Self-Reported Confidence.** Just like in Experiment 1, subjective confidence was measured using the TAM3 and TTF items from Table 10-2 using a post-hoc questionnaire. Thus, the questionnaire was filled out by each participant individually, rather than by the group. Thereby we were able to assess individual participants' subjective confidence to reason about the adequacy during the group inspection process.

10.6.2. Participants

Participants were recruited from a graduate software quality assurance course taught by members of the authors' research group at the University of Duisburg-Essen in winter 2014. The course covers a wide range of software quality assurance topics, including testing, monitoring, and static quality assurance techniques. In particular, inspections are covered during the lecture, explaining the phases of Fagan inspections and different reading techniques. Since 2007, the lab section of the course has included a live inspection session using an excerpt of a requirements specification of a simple system (e.g., a parking garage access system or video rental system) with seeded defects. For the purpose of Experiment 2, the requirements specification of the past years was replaced with the experimental material from Section 10.1. While attendance in the lab sessions, including the inspection session, was mandatory, participation in the

experiment was voluntary: Students were allowed to opt out of data collection without penalty. One such request was received, however the student later withdrew her decision on her own volition.

A total of 40 students enrolled in Systems Engineering or Business Information Systems degree programs participated; 35 were graduate students and the remaining five were undergraduate students preliminarily admitted to their respective Master's program. Since due to administrative reasons, group size could not be controlled for (see Section 10.6.3), the treatment group had 15 participants, while the control group had 25 participants. In this experiment, neither gender nor age were assumed to impact the results, yet a total of 31 male and nine female participants were recorded, aged between 22 and 32 years ($\mu = 24.83$, $\sigma = 2.41$).

10.6.3. Experimental Procedure

In the lecture, students were informed that in the lab sessions, a live inspection would be conducted in order to rehearse the principles of inspections in preparation for the exam. They were furthermore informed that in contrast to the previous years, results of the inspection session would serve an additional research purpose and that albeit participation in the inspection was mandatory (as instructional success depended on participation), participation in the experiment was voluntary. Another inspection session using the requirements specifications of the previous years was scheduled for students who opted out or had a scheduling conflict and could not attend the experimental inspections; data collected therein were not used in this experiment in order to honor participatory consent. All students opted to participate voluntarily, hence no non-experimental inspection was conducted. The experimental procedure outlined in Figure 10-2 roughly corresponds to the inspection phases proposed in [Fagan 1976]. Specifically, Step 1 corresponds to Fagan's "Overview" phase, Step 3 corresponds to the "Preparation" phase and Step 4 corresponds to the "Inspection" phase. Fagan's "Rework" and "Follow-up" phases were beyond the scope of Experiment 2.

Step 1: Introduction, Provide Informed Consent, & Overview. In the next lab after the lecture on static quality assurance, Experiment 2 began with a short introduction and summary of the previous lecture. Informed consent was collected and participants were introduced into the unaltered ACC specification excerpt, into the principles of safety assessment and Functional Hazard Analyses, and were given a list of hazards identified by means of a Functional Hazard Analysis (see Section 10.1). Participants were informed that the purpose of the inspection session was to judge whether or not certain changes made to the ACC specification would result in these hazards possibly occurring during operation or not. This step took place for all students

in plenum at the same time. Each student participant was assigned the role of the inspector. In the inspection sessions, the role of the moderator and reader was carried out by the experimenters. In addition, another researcher employed with the experimenters’ research group was asked to serve as the scribe.

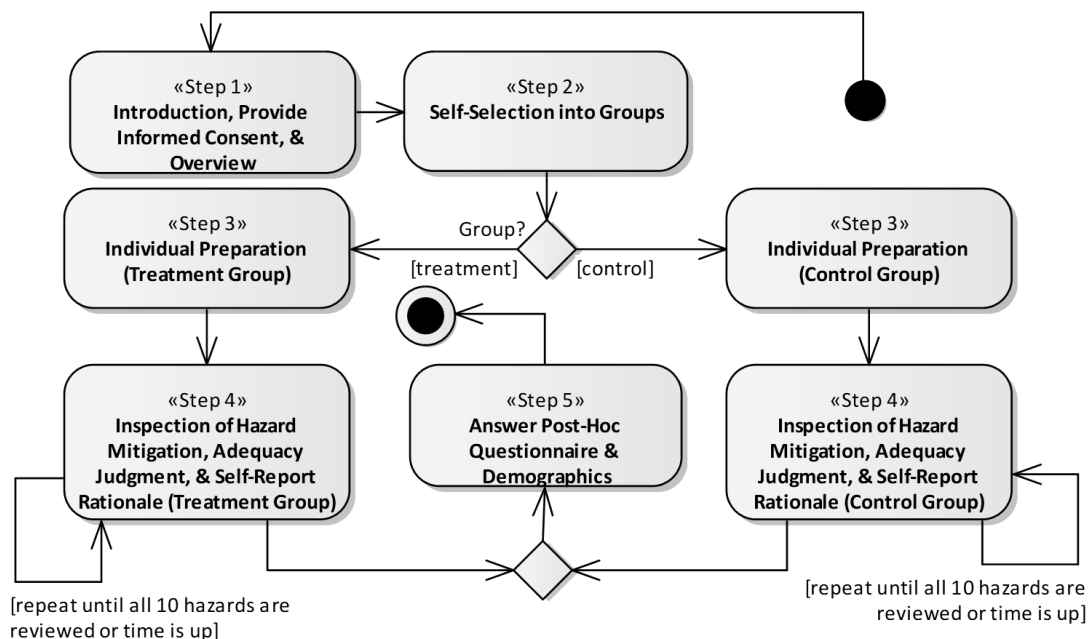


Figure 10-2 Experimental Procedure for Experiment 2.

Step 2: Self-Selection into Groups. Earlier in the semester, the lab has been separated into two sections with about 25 students each. The two sections did not take place simultaneously and students were free to attend either section. For Experiment 2, students were asked to self-select which lab section they would like to attend. This was done through an online poll using the university’s online campus system. It was necessary in order to assign the right experimental material to the participants. We decided by coin toss to conduct the treatment condition inspection session in the earlier lab section and to conduct the control condition inspection one day later during the other lab section. This happened before the students were given the opportunity to select which one they would like to attend. Due to student availability and the voluntary nature of participation, it was not possible to balance the number of participants in each group. This caused a larger number of participants in the control condition ($n_{\text{control}} = 25$, $n_{\text{treatment}} = 15$).

Step 3: Individual Preparation. After participants self-selected into the treatment or control condition, the respective experimental material was made available to them via the university’s online campus system and participants were asked to prepare for the inspection session, which was scheduled for the next lab section meeting. Preparation included studying material on haz-

ard analyses and safety assessment and familiarizing oneself in detail with the ACC specification and the hazard analysis results, which were already introduced during Step 1. During this time period, interaction between students (and groups, for that matter) regarding the experimental material and possible defects therein was not controlled for.

Step 4: Inspection of Hazard Mitigation, Adequacy Judgment, & Self-Report Rationale.

Approximately one week after the introductory session (Step 1), the inspection sessions were conducted for both the treatment group and the control group, respectively. The inspection sessions started with a brief welcome. The participants were thanked for their willingness to participate in the experiment. The moderator showed the original, unaltered specification once again and read the diagrams' meaning out loud. The inspectors were given the opportunity to ask questions about the functionality of the ACC, the diagram syntax, etc. After all questions were answered, the first conceptual mitigation was shown and read out loud by the moderator. The moderator focused on the changed hazard-mitigating requirements in the diagram and compared the changes to how it was in the original specification excerpt. The groups were asked if they were of the opinion that the depicted hazard could still occur during operation and why. Like in Experiment 1, it was ensured that both participant groups inspected approximately equally many information items for each conceptual mitigation by only presenting the one row from the hazard analysis worksheet that was relevant to the hazard being mitigated in the activity diagram. The moderator encouraged open discussion about the depicted Hazard Relation Diagram or activity diagram with hazard analysis result, respectively, explicitly allowing any adequacy judgment, rationale, or questions that were issued by the participants. The moderator guided the resulting discussion by summarizing points, asking controversial questions, and ensuring that every participant could contribute his or her opinion. Whenever discussion died down, the moderator summarized the discussion once again and asked for a vote among the inspectors, whether or not the hazard is adequately mitigated. If this vote motivated further discussion, the moderator allowed the discussion to continue. If not, the majority vote was recorded by the scribe as the adequacy judgment for that hazard. The moderator then proceeded to ask for the reason why inspectors voted the way they did, which was recorded as rationale. A rationale was recorded, if at least one participant agreed, no other participant objected to, or where after a brief discussion consensus about the rationale was reached. In case, there was no immediate consensus, it was decided by voting among the participants if some rationale was recorded.

The author of this dissertation was the moderator of the inspection. This obviously gives rise to the risk of experimenter bias. To reduce this threat to validity, the moderator resorted to

passively observing and neutrally guiding the discussion in both experimental conditions. In particular, for the control condition, it was emphasized to read out loud the contents of the hazard analysis worksheet and to emphasize the changed hazard-mitigating requirements. Furthermore, the scribe acted as a neutral observer and was asked to document if he thought the moderator influenced the discussion. In case influence was detected, the respective result was excluded from further analysis.

Step 4 continued until all conceptual mitigations were inspected or the allotted lab session time of 90 minutes was up. Since most students had other class obligations, the 90 minutes duration was a hard time limit. To increase comparability, the order in which conceptual mitigations were inspected was not randomized, such that the same conceptual mitigations would be inspected by both groups. This was deemed appropriate, in contrast to Experiment 1, as result skewing due to primacy, recency, or carry-over effects is not an issue in group activities.

Step 5: Answer Post-Hoc Questionnaire & Demographics. After Step 4 concluded, participants were given a paper-based version of the demographic and post-hoc questionnaires already used in Experiment 1. The questionnaires were answered by each participant individually, since the aim was to measure the self-reported confidence of individuals rather than of the group as a whole.

10.7. Experiment 3: Individual Reviews (Within-Subjects)

To substantiate the empirical evaluation of Hazard Relation Diagrams, we designed Experiment 3 as a repetition study to Experiment 1. While Experiment 1 was designed as a between-subjects experiment in order to reduce the need for training overhead, Experiment 3 was designed as a within-subjects repeated measures experiment [Wohlin et al. 2012]. Since we expected a smaller amount of participants, a repeated measures design was favored over a between-subjects design, as repeated measures experiments are more robust against small number of participants. In Experiment 3, all participants performed the same task as in Experiment 1, but participated in both treatment condition (reviews using Hazard Relation Diagrams) as well as the control condition (reviews using conventional activity diagrams and hazard analysis results).

10.7.1. Specific Metrics

Every participant reviewed ten diagrams, but all participants were subjected to both treatment and control condition. The measurements for the hypotheses in Experiment 3 is very similar to those outlined in Experiment 1:

- **Hypothesis H1: Rationales.** Equivalently to Experiment 1, we asked participants to judge whether a hazard was adequately or inadequately mitigated (regardless of presentation mode) and asked participants to provide a reason for their decision. These rationales were treated the same way during post-experimental analyses as the rationales provided during Experiment 1 (see Section 10.5.1 for details).
- **Hypothesis H2: Effectiveness.** Just like in in Experiment 1, we measured the number of correctly judged adequate (i.e. true positive), correctly judged inadequate (i.e. true negative), incorrectly judged adequate (i.e. false positive), and incorrectly judged inadequate (i.e. false negative) conceptual mitigations. In contrast to Experiment 1, the number of adequate and inadequate conceptual mitigations was not even across both presentation modes because participants were shown five Hazard Relation Diagrams and five Activity Diagrams and corresponding hazard analysis results. A participant could, for instance, review two adequate Hazard Relation Diagrams, three inadequate Hazard Relation Diagrams, three adequate Activity Diagrams, and two inadequate Activity Diagrams, for a total of ten diagrams. Therefore, rather than measuring the total amount of true positive, true negative, false positive, and false negative answers, we calculated the respective ratio for each participant. Table 10-5 depicts the differences for the dependent variables for hypothesis H2 in Experiment 3.

Table 10-5 Specific Metrics for Hypotheses H3.2 in Experiment 3.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H3.2	DV3.6	Dependent	ratio of true positive answers	Ratio
	DV3.7		ratio of true negative answers	
	DV3.8		ratio of false positive answers	
	DV3.9		ratio of false negative answers	

- **Hypothesis H3: Efficiency.** Like in Experiment 1, each participant’s review efficiency was measured by determining the time needed in seconds.
- **Hypothesis H4: Self-Reported Confidence.** In contrast to Experiment 1, the level of agreement with TAM3 and TTF questionnaire items could not be measured on a Likert-scale. The key reason for this is that in a repeated measures design, participants’ answers are compared to their own answers in different conditions. Therefore, we applied the semantic differential technique (see [Osgood et al. 1957; Verhagen et al. 2015]), which polarizes ordinates on measurement scales. We therefore adapted the definition of the original TAM3 and TTF questions to reflect both notations. The consequence is that this allowed assessing participants’ related relative preference of either Hazard Relation Diagrams or Activity Diagrams and hazard analysis worksheets. We used Likert-style 5-point ordinates

ranging from “1: true for activity diagrams” to “5: true for Hazard Relation Diagrams.” A median of “3” hence indicated indifference between both presentation modes. The dependent variables for hypothesis H4 in Experiment 3 are shown in Table 10-6.

Table 10-6 Specific Metrics for Hypotheses H3.4 in Experiment 3.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H3.4	DV3.14	Dependent	relative preference with regard to “Perceived Usefulness”	Ratio
	DV3.15		relative preference with regard to “Self-Efficacy”	
	DV3.16		relative preference with regard to “Result Demonstrability”	
	DV3.17		relative preference with regard to “The Right Data”	
	DV3.18		relative preference with regard to “Meaning”	
	DV3.19		relative preference with regard to “Presentation”	
	DV3.20		relative preference with regard to “Training”	
	DV3.21		relative preference with regard to “Confusion”	

10.7.2. Participants

Participants for Experiment 3 were recruited from an undergraduate software engineering course instructed by members the author’s department at the Oswego State University in fall 2015. Prior to recruitment in the course, approval of the institution’s research ethics board was sought and granted. The software engineering course is aimed towards upper classmen, i.e. at least third year undergraduate students or, in exceptional cases, advanced second semester with excellent grade point average. The course featured all aspects of software engineering in industry projects, with a key focus on process models, project phases, and formal as well as semi-formal languages. Specifically, students were taught UML activity diagrams as a central description language for system specification and were introduced with the aims of software quality assurance and safety engineering. The study was administered at a time in the course when students would benefit from participation, as the study’s review task would rehearse some of the instructed material.

Student participation was voluntary and a total of 31 students participated, all of which were undergraduates enrolled in Electrical Engineering, Software Engineering, or Computer Science degree programs. The majority of eighteen participants were male with two female participants, eleven participants declined to provide gender information. Participants’ age was between 18 and 45 years ($\mu = 22.95$, $\sigma = 5.995$). We assumed that gender and age has no impact on the results of the Experiment 3.

10.7.3. Experimental Procedure

Like Experiment 1, Experiment 3 was implemented using the survey website SoSci Survey [SoSci Survey]. For this purpose, the implementation was Experiment 1 was copied and adapted

for a repeated measures design. The class met three times a week for 55 minutes. An introductory briefing session was administered during one of those meetings, in which the principles of software validation and safety engineering, including hazards analysis of embedded systems, as well as UML activity diagrams was briefly reviewed. Then participants were informed that during the next class meeting, an experiment would take place, participation in which would help them study for an upcoming midterm exam, as the study rehearsed some of the principles taught in class. However, students were informed that participation was voluntary and they were free to remain absent from the next class meeting for all participants without penalty. Afterwards, the unaltered ACC specification excerpt from Experiment 1 was used to introduce how to perform the reviews. The review task was illustrated by means of the two example hazards from Experiment 1. After the briefing concluded, students were dismissed and the briefing material as well as the study link was made available to all students through an online campus system in case students wanted to participate remotely, rather than in the next class meeting. No such remote participation was recorded. During the next class meeting, data collection for Experiment 3 commenced. It consisted of the steps depicted in Figure 10-3.

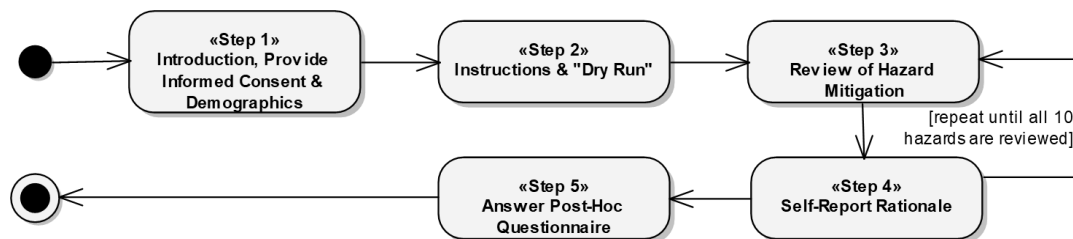


Figure 10-3 Experimental Procedure for Experiment 3.

Step 1: Introduction, Provide Informed Consent, & Demographics. Like Experiment 1, Experiment 3 began with a short introduction, the retrieval of informed consent as well as the collection of demographic data. Students were again informed that participation is voluntary, anonymous, and that they are free to discontinue and leave at any point.

Step 2: Instructions & “Dry Run.” In contrast to Experiment 1, participants were not separated into groups, as the review task would be conducted for both treatment and control condition. For this purpose, the review task was explained once again by summarizing the information from the briefing session. Two example runs were conducted using both treatment and control condition.

Step 3: Review of Hazard Mitigation. Like in Step 4 of Experiment 1 (see Section 10.5.3), this step entailed the review of conceptual mitigations pertaining to one randomly selected hazard. The sequence in which the five activity diagrams and the five Hazard Relation Diagrams

were presented was randomized for each participant in order to reduce primacy, recency, and carry-over effects [Wohlin et al. 2012]. For each conceptual mitigation, an activity diagram or a Hazard Relation Diagram was available (i.e. a total of twenty stimulus diagrams, two for each hazard, see Section 10.1). Whether a conceptual mitigation was shown as an activity diagram or as a Hazard Relation Diagram was randomized. Randomization ensured that each participant would be presented with exactly five activity diagrams and five Hazard Relation Diagrams and that each participant would be presented with exactly five adequate conceptual mitigations and five inadequate conceptual mitigations (regardless if these conceptual mitigations were presented as an activity diagram or Hazard Relation Diagram). However, an effort was made to balance adequate and inadequate conceptual mitigations across presentation mode, such that no participant would, for example, review only adequate conceptual mitigations as Hazard Relation Diagrams. Thus every participant reviewed either two or three adequate or inadequate conceptual mitigations, respectively, as activity diagrams or Hazard Relation Diagrams, respectively. This step was repeated ten times until all conceptual mitigations from Section 10.1 were reviewed. As in Experiment 1, we ensured that both conditions reviewed approximately equally many information items. Furthermore, like in Experiment 1, participants could review each conceptual mitigation for an indeterminate amount of time while response times were recorded and participants were asked to indicate “yes” if they are of the opinion that the hazard may still occur during operation and “no” otherwise, allowing for an indeterminate amount of time to change such an assessment. Moreover, we recorded the confidence of the participants.

Step 4: Self-Report Rationale. Like in Experiment 1, participants provided a brief reason why they chose “yes” or “no” in the previous step. Participants were able to return to the previous step and change their answer if they changed their mind while reasoning about the conceptual mitigation that was shown to them for reference along with their decision from the previous step.

Step 5: Answer Post-Hoc Questionnaire. After all ten conceptual mitigations were reviewed, participants were presented with the post-hoc questionnaire. The order in which questionnaire items were displayed was randomized to avoid that similar questions pertaining to the same quality focus were adjacent to one another.

10.8. Experiment 4: Fagan Inspections (Within-Subjects)

Experiment 4 was designed to investigate the impact of using Hazard Relation Diagrams during Fagan inspections in within-subjects conditions. We kept the experimental design as

close to Experiment 2 as possible. However, due to the within-subjects design, there are differences, as described in the following.

10.8.1. Specific Metrics

We measure the hypotheses from Section 10.4 as follows:

- **Hypothesis H1: Rationales.** Like in Experiment 2, we record the rationales achieved through group consensus.
- **Hypothesis H2: Effectiveness.** We used the same measures as in Experiment 3. These are shown in Table 10-7.

Table 10-7 Specific Metrics for Hypotheses H4.2 in Experiment 4.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H4.2	DV4.6	Dependent	ratio of true positive answers	Ratio
	DV4.7		ratio of true negative answers	
	DV4.8		ratio of false positive answers	
	DV4.9		ratio of false negative answers	

- **Hypothesis H3: Efficiency.** Like in Experiment 2, we measured the group’s overall efficiency for each conceptual mitigation, for both the treatment stimuli as well as the control stimuli. We counted the respective true positive, true negative, false positive, and false negative answers for both stimulus types and dividing that number by the time allotted for the inspection session times the amount of inspectors present. Again, we mapped the result to a more human-readable interval and, as for Experiment 2, this metric is very sensitive to unequal group sizes. Table 10-8 shows the measures for hypothesis H4.3.

Table 10-8 Specific Metrics for Hypotheses H4.3 in Experiment 4.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H4.3	DV4.10	Dependent	$1000 * (\text{number of true positive answers} / (\text{number of group members} * \text{time}))$	Ratio
	DV4.11		$1000 * (\text{number of true negative answers} / (\text{number of group members} * \text{time}))$	
	DV4.12		$1000 * (\text{number of false positive answers} / (\text{number of group members} * \text{time}))$	
	DV4.13		$1000 * (\text{number of false negative answers} / (\text{number of group members} * \text{time}))$	

- **Hypothesis H4: Self-Reported Confidence.** For self-reported confidence, we adopted the metric from Experiment 3 (see Section 10.7.1) and applied the TAM3 and TTF items. As in Experiment 2, a questionnaire with these items was filled out by each participant, rather than by the group as a whole. The specific metrics used for hypothesis H4 in Experiment 4 are summarized in Table 10-9.

Table 10-9 Specific Metrics for Hypotheses H4.4 in Experiment 4.

Assoc Hypo.	Variable			Scale
	ID	Type	Definition	
H4.4	DV4.14	Dependent	relative preference with regard to “Perceived Usefulness”	Ratio
	DV4.15		relative preference with regard to “Self-Efficacy”	
	DV4.16		relative preference with regard to “Result Demonstrability”	
	DV4.17		relative preference with regard to “The Right Data”	
	DV4.18		relative preference with regard to “Meaning”	
	DV4.19		relative preference with regard to “Presentation”	
	DV4.20		relative preference with regard to “Training”	
	DV4.21		relative preference with regard to “Confusion”	

10.8.2. Participants

The participants for Experiment 4 were recruited from an undergraduate software quality assurance course instructed by the author at the Oswego State University in fall 2015. The course covered a wide range of quality assurance techniques, such as verification strategies, code-based testing, and static quality assurance. The lecture of that course has been extended to cover Fagan inspections in detail before the experiment sessions took place. During that time, a department colleague offered to recruit participants from her graduate level human factors course for the experiment as well. User requirements validation and usability inspections were part of the curriculum in her course. Participation in the experiment would thus offer immediate benefit for the students. Hence another inspection session was conducted for the human factors course.

A total of two inspection sessions took place. One with participants recruited from the software quality course (called Session 1), and one with participating students from the human factors course (called Session 2). Before recruitment approval of the institution’s research ethics board was obtained. In both cases, the inspection session was part of the curriculum. Participation in the inspection session was hence mandatory, albeit not strictly enforced. Students were informed that participation in data collection was voluntary. Moreover, students were free to decline surrendering data without penalty. In such a case, a secondary inspection session would have been scheduled for students opting out. No such opt-out request was received.

In Session 1, a total of 24 undergraduate students participated. All students were enrolled in Electrical Engineering, Software Engineering, or Computer Science degree programs. Albeit age and gender is not assumed to influence the performance in the inspection session, we recorded 22 male participants, two female participants, aged between 19 and 34 years ($\mu = 22.45$, $\sigma = 4.228$).

In Session 2, a total of 27 students participated, eight were graduate students, eighteen undergraduates, and one participant self-identified as a part time student working in the industry.

Graduate participants were exclusively enrolled in a Masters-level Human Computer Interaction degree program. Undergraduate participants were provisionally accepted to the Human Computer Interaction degree program. The graduate level human factors course was co-listed as an upper division undergraduate psychology course. Therefore, the inspection session encompassed a considerable portion of undergraduate students enrolled in Psychology or Human Development undergraduate degree programs. In Session 2, six participants were male and the remainder was female. Participant's age ranged between 19 and 53 years ($\mu = 26.82$, $\sigma = 9.548$).

10.8.3. Experimental Procedure

In both sessions, Experiment 4 began with a lecture on Fagan inspections as part of the regularly scheduled curriculum. Students were informed that in the next class meeting, a live inspection would be conducted in order to rehearse the principles of inspections in preparation for the exam, similarly to Experiment 2. Students were furthermore informed that data collection would take place to serve a research purpose. Mandatory participation in the inspection was stressed, but it was made clear that participation in the experiment was voluntary. Figure 10-4 depicts the experimental procedure.

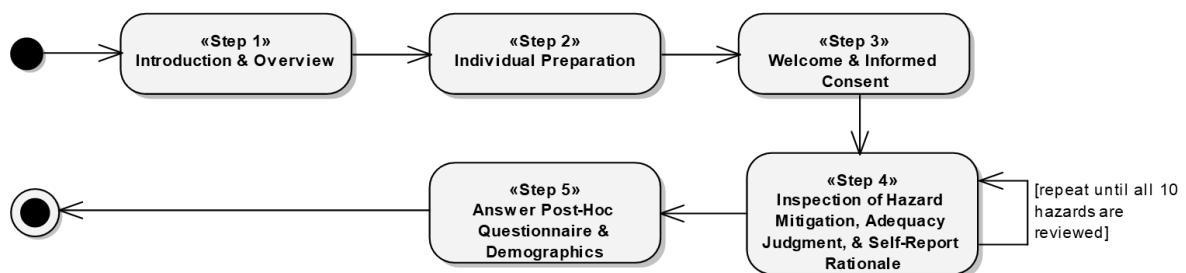


Figure 10-4 Experimental Procedure for Experiment 4.

Step 1: Introduction & Overview. In the class meeting after the lecture on Fagan inspections, an introductory session was administered, which featured a brief summary of the lecture and detailed further information about the validation objective. Participants were introduced into the unaltered ACC specification excerpt, received an overview over the principles of safety assessment and hazard analysis, and were given a hazard analysis worksheet. Like in Experiment 2, participants were informed that the purpose of the inspection session was to judge whether or not conceptual mitigations incorporated into the ACC specification would render the system safe with respect to specific hazards. Volunteers for the role of the scribe were sought. The role of the moderator and the reader was assigned to the course instructors.

Step 2: Individual Preparation. The experimental material (also discussed during Step 1) was made available through an online campus system. Participants were asked to familiarize themselves with the diagrams until the next class meeting. During that time, interaction between participants was not controlled for, but participants were encouraged to take notes about questions or findings they encounter.

Step 3: Welcome & Informed Consent. About four days later, the next class meetings in both courses commenced the actual inspection session with Step 3. After a brief welcome, participants were thanked for their participation. Informed consent was collected and the unaltered ACC specification was shown and explained one more time. Afterwards, the moderator asked if there were any questions for clarification, which were discussed in plenum.

Step 4: Inspection of Hazard Mitigation, Adequacy Judgment, & Self-Report Rationale. The first conceptual mitigation was shown and read out loud by the moderator. Like in Experiment 2, the moderator focused on the changed hazard-mitigating requirements in the diagram and compared the changes to the unaltered ACC specification excerpt. The participants were asked if they were of the opinion that the depicted hazard could still occur during operation and why. The moderator guided the discussion like in Experiment 2. When the discussion came to a natural end, the moderator summarized the discussion once again and asked for a vote among the inspectors, whether or not the system is now considered safe with respect to the depicted hazard. The moderator allowed the discussion to continue if this triggered further opinions. If summarization did not prompt additional discussion, the majority vote for the adequacy judgment for that hazard was recorded by the scribe and the moderator ask the participants about their rationales for their decision. Like in Experiment 2, every rationale was recorded, where at least one participant agreed, no other participant objected to, or where after a brief discussion consensus about the rationale was reached. If there was no consensus, the majority vote was recorded.

The order of the hazards was randomized. This included randomizing the order in which specific conceptual mitigation were shown and whether they would be presented as a Hazard Relation Diagram (treatment condition) or as an activity diagram with corresponding hazard analysis results (control condition). Randomization was done such that in both sessions, an equal number of adequate and inadequate as well as treatment stimuli and control stimuli were inspected. In both sessions, Step 4 continued until all conceptual mitigations were inspected or the allotted class time was up (55 minutes for Session 1 and 70 minutes for Session 2). Due to the participants' other class obligations, this was a hard time limit.

Like in Experiment 2, experimenter bias was a considerable threat to validity, because the author of this dissertation acted as the moderator in both sessions. The same measures to mitigate this threat was applied as in Experiment 2.

Step 5: Answer Post-Hoc Questionnaire & Demographics. Participants were given a paper-based version of the demographic and post-hoc questionnaires that were used in Experiment 3. The questionnaires were answered by each participant individually.

10.9. Data Preparation

After the review experiments (i.e. Experiment 1 and Experiment 3) concluded, the data of all fully completed data sets were downloaded from the survey website, thereby discarding incomplete data sets (e.g., due to participants who discontinued participation). All data was transcoded into a data file for the statistical analysis tool SPSS and prepared for statistical analysis. Similarly, the inspection protocol for all inspection sessions from Experiment 2 and Experiment 4 was transferred into the SPSS data file as well. Since the order in which diagrams was randomized in the experiments, a common order of diagrams was established using a unique ID.

For each adequacy judgment (i.e. yes/no indication concerning the adequacy of the conceptual mitigation in each diagram), it was recorded whether this was a correct or incorrect responses, thereby determining true positive, false positive, true negative, and false negative answers for Experiment 1 and 3.

Rationales were qualitatively analyzed and prepared for statistical analysis by means of constant comparison [Corbin and Strauss 2008]. The constant comparison technique entails reading every statement and categorizing the statements according to constant concepts mentioned in the statement. This was done by counting the number of statements in each rationale (i.e. when a participant in Experiments 1 and 3 used conjunctions such as “and” or “furthermore” or when multiple rationales were given by the inspectors in Experiments 2 and 4). If in Experiments 1 and 3, a rationale included phrases such as “don’t know,” “just guessed,” or other indications that the participant did not provide a proper reason (e.g., placeholders such as “...,” “bla,” or if the text field was simply left empty), this was seen as an invalid rationale and not counted. For all valid rationales, the rationales were categorized into exactly one of the categories, as outlined in Section 10.4:

- **Semantics:** The rationale was based on the individuals’ understanding of the semantics of the subject matter, e.g., functionality of the system, etc.
- **Syntax:** The rationale was based on the individuals’ understanding of the diagram notation, the diagram’s layout, etc.

- **Trigger Condition:** The rationale indicated that one or more trigger conditions are or are not rendered sufficiently unlikely to occur during operation in order to mitigate the hazard.
- **Safety Goal:** The rationale indicated that the safety goal was or was not fulfilled or is semantically wrong.
- **Conceptual Mitigation:** The rationale explained how and why the hazard-mitigating requirements do or do not lead to resolution of the hazard.

Rationales that could not be coded into either category were counted as invalid and discarded. Initial qualitative analysis and categorization was done by the author, where each rationale was categorized independently from all other rationales. Moreover, in order to reduce researcher bias, each rationale was categorized without knowledge about what condition (i.e. treatment or control) the rationale in question belonged to (i.e. which group the participant was in Experiment 1, by which inspection team the rationale was given in Experiment 2, or whether the associated diagram was a Hazard Relation Diagram or an activity diagram in Experiments 3 and 4). This was done by temporarily removing the information pertaining to group membership and experimental condition from the data file during rationale categorization and re-introducing it based on participant IDs and inspection protocols after qualitative analysis was complete. In addition, rationale categorization and transcoding was verified by a total of three independent researchers (two researchers at the University of Duisburg-Essen and one additional researcher at the State University of New York at Oswego) in order to identify erroneously coded data. Each derivation between the authors' and the verifiers' categorization was individually reviewed, bilaterally discussed, and rectified. In case of unresolvable conflicts, the rationale was discarded. This was the case only in very few rationales. In only five cases, there were conflicting categorizations were identified, all of which were easily resolved. Hence, due to the low number of conflicts, we refrained from computing interrater reliability.

After data preparation was concluded, descriptive statistics and frequencies were computed using SPSS Version 22.0 [IBM SPSS 22] and the data file was manually reviewed in order to identify outliers and irregular responses. Irregular responses entailed all participant data sets which contained three or fewer valid rationales in total for all hazards, where participants answered in patterns (e.g., alternating yes/no answers for each adequacy judgment or using always the same answer on the post-hoc questionnaire). Such irregular responses were discarded. In addition, since in Experiment 1, participants were able to run the experiment at home, controlling for participants "pausing" participation (e.g., by leaving the survey open in the browser and leaving the computer) or not fully concentrating on the experiment and doing other things while participating (see Section 10.5.3) was not possible. This behavior likely resulted in largely

skewed reaction times. Therefore, data sets from Experiment 1 and 3 were counted as irregular responses, if participants took less time than 5.5 minutes (since this was the minimal time needed to view each stimulus and questionnaire page once) or more than 40 minutes (about twice the mean of total response times, which is a standard exclusion criterion for multivariate statistics [Tabachnik and Fidell 2010]). These cases were excluded from all analyses as well. In total, thirteen participant data sets from both groups were excluded from Experiment 1, yielding a total of 120 valid cases, with 59 participants in the treatment group and 61 participants in the control group. For Experiments 2, 3, and 4, the same criteria (with the exception of time limits) was used to review and clean the data file from outliers and irregular responses. No participant data set nor the inspection results met any of these criteria, so no data set was discarded from the batch collected during these experiments.

Since the original TAM3 and TTF items and questionnaire questions were adapted to suit the needs of these experiments, it was necessary to validate questionnaire cohesion. For this purpose, Cronbach's α [Cronbach 1951] was computed on the post-hoc questionnaire items for the quality foci from Table 10-2. For further analysis, only those quality foci where cohesion was at least high (i.e., $\alpha > 0.7$, which is the standard criterion, cf. [Cronbach 1951]) were retained. The results are shown for each experiment in Table 10-10 with retained quality foci printed in bold.

Table 10-10 Cronbach's α for all quality foci from Table 10-2 and all experiments.

TAM3 / TTF item		Cronbach's α			
DV ID	Quality Focus	Experiment 1	Experiment 2	Experiment 3	Experiment 4
DV14.x	Perceived Usefulness	0.906	0.819	0.930	0.927
DV15.x	Self-Efficacy	0.796	0.770	0.405	0.416
DV16.x	Result Demonstrability	0.793	0.783	0.774	0.809
DV17.x	The Right Data	0.629	0.516	0.439	0.311
DV18.x	Meaning	0.747	0.737	0.740	0.527
DV19.x	Presentation	0.502	0.515	0.928	0.840
DV20.x	Training	0.641	0.348	< 0.01	0.333
DV21.x	Confusion	0.736	0.784	0.786	0.547
Overall α of retained quality foci		0.914	0.785	0.869	0.949

10.10. Participant Experience Levels across Experiments

The post-hoc questionnaire administered in all experiments pertained, among other things, to the participants' levels of experience with automotive software engineering (since the case example was from the automotive domain), requirements engineering in general, modeling using activity diagrams (since this is the foundation of Hazard Relation Diagrams), static requirements quality assurance using reviews or inspections (since this was the experimental task), dynamic quality assurance (i.e., testing and verification), and functional design and system architecture (since experience in this area may increase participants ability to think abstractly

about diagrams in general). Like the questions pertaining to participant confidence (see Section 10.4), each experience level question was measured on a 5-point-Likert scale [1: “no experience”; 2: “experience from academic homework”; 3: “experience from one or more academic projects”; 4: “experience from one industry project”; 5: “experience from multiple industry projects”]. We have deliberately chosen this scale, as these ordinates yield more objective categorizations than ordinates that, for instance, ask for “very little experience” to “very much experience.” Table 10-11 shows the levels of experience of participants in all four experiments. For each ordinate, Table 10-11 shows the total number as well as the relative amount of participants. It is to note that there were participants who agreed to participate in the experiments, yet declined to provide demographic information. The total n for demographics in Experiment 1 is = 130, in Experiment 2, n = 40, in Experiment 3, n = 30, and in Experiment 4, n=25.

Table 10-11 Relative Levels of Experience in Participants for all Experiments.

Experience Level in...	Exp.	no experience		academic homework		academic projects		industry project		mult. industry projects	
Automotive Software Engineering	1	62	47.3%	54	41.2%	11	8.4%	2	1.5%	2	1.5%
	2	23	57.5%	7	17.5%	4	10.0%	1	2.5%	3	7.5%
	3	23	76.7%	2	6.7%	3	10.0%	0	0.0%	2	6.7%
	4	20	80.0%	5	20.0%	0	0.0%	0	0.0%	0	0.0%
Requirements Engineering	1	12	9.2%	69	52.7%	39	29.8%	8	6.1%	3	2.3%
	2	4	10.0%	17	42.5%	10	25.0%	3	7.5%	4	10.0%
	3	4	13.3%	11	36.7%	8	26.7%	3	10.0%	4	13.3%
	4	7	28.0%	9	36.0%	6	24.0%	1	4.0%	2	8.0%
Modeling using Activity Diagrams	1	20	15.3%	61	46.6%	39	29.8%	10	7.3%	1	0.8%
	2	1	2.5%	16	40.0%	13	32.5%	5	12.5%	3	7.5%
	3	4	13.3%	12	40.0%	10	33.3%	2	6.7%	2	6.7%
	4	4	16.0%	10	40.0%	9	36.0%	0	0.0%	2	8.0%
Reviews and Inspections	1	45	34.4%	68	51.9%	14	10.7%	4	3.1%	0	0.0%
	2	17	42.5%	11	27.5%	6	15.0%	3	7.5%	1	2.5%
	3	8	26.7%	8	26.7%	7	23.3%	4	13.3%	3	10.0%
	4	6	24.0%	12	48.0%	5	20.0%	0	0.0%	2	8.0%
Req. Test and Software Test	1	38	29.0%	62	47.3%	25	19.1%	6	4.3%	0	0.0%
	2	8	20.0%	16	40.0%	9	22.5%	3	7.5%	2	5.0%
	3	8	26.7%	10	33.3%	5	16.7%	4	13.3%	3	10.0%
	4	10	40.0%	8	32.0%	4	16.0%	1	4.0%	2	8.0%
Design and System Architecture	1	46	35.1%	52	36.7%	27	20.6%	6	4.3%	0	0.0%
	2	5	12.5%	12	30.0%	11	27.5%	6	15.0%	4	10.0%
	3	6	20.0%	10	33.3%	8	26.7%	3	10.0%	3	10.0%
	4	9	36.0%	9	36.0%	4	16.0%	2	8.8%	1	4.0%

In order to increase comparability between the experimental results, the experience levels of the experimental populations ought to be homogenous. To investigate differences in experimental populations, we computed the mean experience level (μ) and standard deviations (σ) by summing up the numeric value of the ordinates. To test the significance of differences in means between groups we used independent group T-Tests [Corbin and Strauss 2008] for all experiments. We deliberately computed two independent group T-Tests as opposed to one One-way ANOVA [Fisher 1918] in order to highlight the comparison between experiments using the

same validation technique. For each T-Test result, we calculated Cohen's d [Cohen 1992] for statistical power and effect size.

Table 10-12 summarizes the means (μ), standard deviations (σ), T-Test results (t, dF), significance (p), effect size (d), and statistical power (η^2) of participants' levels of experience between Experiment 1 and 3. Significant T-Test results for higher means in Table 10-12 are printed in bold.

Table 10-12 Means, Std. Dev., T-Test, and Power for Participants' Levels of Experience in Exp. 1 & 3.

Experience Level in...	Exp.	μ	σ	t	dF	p	d	η^2																																																						
Automotive Software Engineering	1	1.69	0.814	-0.701	31.104	0.488	0.162	0.609																																																						
	3	1.53	1.137						Requirements Engineering	1	2.40	0.829	1.426	32.265	0.163	0.315	0.563	3	2.73	1.230	Modeling using Activity Diagrams	1	2.32	0.853	1.041	38.838	0.304	0.221	0.541	3	2.53	1.042	Reviews and Inspections	1	1.82	0.739	2.870	33.366	0.007	0.669	0.716	3	2.53	1.306	Req. Test and Software Test	1	1.99	0.818	1.905	34.383	0.065	0.441	0.625	3	2.47	1.306	Design and System Architecture	1	1.95	0.862	2.632	35.874
Requirements Engineering	1	2.40	0.829	1.426	32.265	0.163	0.315	0.563																																																						
	3	2.73	1.230						Modeling using Activity Diagrams	1	2.32	0.853	1.041	38.838	0.304	0.221	0.541	3	2.53	1.042	Reviews and Inspections	1	1.82	0.739	2.870	33.366	0.007	0.669	0.716	3	2.53	1.306	Req. Test and Software Test	1	1.99	0.818	1.905	34.383	0.065	0.441	0.625	3	2.47	1.306	Design and System Architecture	1	1.95	0.862	2.632	35.874	0.012	0.586	0.638	3	2.57	1.223						
Modeling using Activity Diagrams	1	2.32	0.853	1.041	38.838	0.304	0.221	0.541																																																						
	3	2.53	1.042						Reviews and Inspections	1	1.82	0.739	2.870	33.366	0.007	0.669	0.716	3	2.53	1.306	Req. Test and Software Test	1	1.99	0.818	1.905	34.383	0.065	0.441	0.625	3	2.47	1.306	Design and System Architecture	1	1.95	0.862	2.632	35.874	0.012	0.586	0.638	3	2.57	1.223																		
Reviews and Inspections	1	1.82	0.739	2.870	33.366	0.007	0.669	0.716																																																						
	3	2.53	1.306						Req. Test and Software Test	1	1.99	0.818	1.905	34.383	0.065	0.441	0.625	3	2.47	1.306	Design and System Architecture	1	1.95	0.862	2.632	35.874	0.012	0.586	0.638	3	2.57	1.223																														
Req. Test and Software Test	1	1.99	0.818	1.905	34.383	0.065	0.441	0.625																																																						
	3	2.47	1.306						Design and System Architecture	1	1.95	0.862	2.632	35.874	0.012	0.586	0.638	3	2.57	1.223																																										
Design and System Architecture	1	1.95	0.862	2.632	35.874	0.012	0.586	0.638																																																						
	3	2.57	1.223																																																											

The results depicted in Table 10-11 and Table 10-12 show that the participant populations in Experiment 1 and 3 are largely homogeneous. This is indicated by the fact that the majority of participants in both experiments claimed either no experience from the respective areas or experience from academic work (see Table 10-12). A notable exception is the claimed industrial experience (zero participants in Experiment 1; three participants in Experiment 3). These three individuals in Experiment 3 who claimed industrial experience were students, who had a side-job in local software development companies. Overall, the only experience areas with significant differences between experimental populations are “reviews and inspections” and “design and system architecture” (see Table 10-12). The assumption that this difference had little impact on the experimental results is supported by the fact that effect size and statistical power was medium ($0.3 < d, \eta^2 < 0.8$) for both significant T-Test results in Table 10-12. In principle, independent samples T-Tests are sensitive to largely unequal n. The difference in population ($n_{\text{Experiment 1}} = 130, n_{\text{Experiment 2}} = 30$) size has thus likely exaggerated significance.

Table 10-13 summarizes the means (μ), standard deviations (σ), T-Test results (t, dF), significance (p), effect size (d), and statistical power (η^2) of participants' levels of experience between Experiment 2 and 4.

Table 10-13 Means, Std. Dev., T-Test, and Power for Participants' Levels of Experience in Exp. 2 & 4.

Experience Level in...	Exp.	μ	σ	t	dF	p	d	η^2																																																						
Automotive Software Engineering	2	1.25	2.667	0.116	41.882	0.908	0.026	0.052																																																						
	4	1.20	0.408						Requirements Engineering	2	2.05	2.791	-0.460	56.791	0.647	0.107	0.069	4	2.28	1.173	Modeling using Activity Diagrams	2	2.23	2.778	-0.442	54.127	0.660	0.100	0.067	4	2.44	1.044	Reviews and Inspections	2	1.40	2.639	-1.703	56.153	0.094	0.397	0.335	4	2.20	1.080	Req. Test and Software Test	2	1.78	2.713	-0.618	58.429	0.539	0.143	0.085	4	2.08	1.222	Design and System Architecture	2	2.20	2.848	0.239	55.089
Requirements Engineering	2	2.05	2.791	-0.460	56.791	0.647	0.107	0.069																																																						
	4	2.28	1.173						Modeling using Activity Diagrams	2	2.23	2.778	-0.442	54.127	0.660	0.100	0.067	4	2.44	1.044	Reviews and Inspections	2	1.40	2.639	-1.703	56.153	0.094	0.397	0.335	4	2.20	1.080	Req. Test and Software Test	2	1.78	2.713	-0.618	58.429	0.539	0.143	0.085	4	2.08	1.222	Design and System Architecture	2	2.20	2.848	0.239	55.089	0.812	0.055	0.055	4	2.08	1.115						
Modeling using Activity Diagrams	2	2.23	2.778	-0.442	54.127	0.660	0.100	0.067																																																						
	4	2.44	1.044						Reviews and Inspections	2	1.40	2.639	-1.703	56.153	0.094	0.397	0.335	4	2.20	1.080	Req. Test and Software Test	2	1.78	2.713	-0.618	58.429	0.539	0.143	0.085	4	2.08	1.222	Design and System Architecture	2	2.20	2.848	0.239	55.089	0.812	0.055	0.055	4	2.08	1.115																		
Reviews and Inspections	2	1.40	2.639	-1.703	56.153	0.094	0.397	0.335																																																						
	4	2.20	1.080						Req. Test and Software Test	2	1.78	2.713	-0.618	58.429	0.539	0.143	0.085	4	2.08	1.222	Design and System Architecture	2	2.20	2.848	0.239	55.089	0.812	0.055	0.055	4	2.08	1.115																														
Req. Test and Software Test	2	1.78	2.713	-0.618	58.429	0.539	0.143	0.085																																																						
	4	2.08	1.222						Design and System Architecture	2	2.20	2.848	0.239	55.089	0.812	0.055	0.055	4	2.08	1.115																																										
Design and System Architecture	2	2.20	2.848	0.239	55.089	0.812	0.055	0.055																																																						
	4	2.08	1.115																																																											

Table 10-11 and Table 10-13 show that the experimental populations in Experiment 2 and 4 were homogenous. Like in Experiments 1 and 3, the majority of participants indicated no experience or experience from academic homework in the respective areas shown in Table 10-11. Independent group T-Test confirm homogeneity in experimental populations, as none of the by-group comparisons shown in Table 10-13 revealed significant results. Moreover, for all T-Test results effect size and statistical power was small ($0.3 < d, \eta^2$) or, in case of “reviews and inspections” approaching small ($d = 0.397, \eta^2 = 0.335$, see Table 10-13).

Overall, there is no statistically significant difference in experience levels between groups and experimental populations. The groups are therefore comparable.

11. Experimental Results

In this chapter, the results of all four experiments are presented⁹. In order to improve comparability of results for each hypotheses, we present the results ordered by hypotheses, rather than ordered by experiments. Using the data prepared according to Section 10.9 we

1. tested for normal distribution of measurements;
2. determined the mean difference between treatment and control groups using independent group T-Tests [Student 1908] for all dependent variables (in the following: DVs) from Table 10-3; and
3. computed Cohen's d [Cohen 1992] to calculate effect size and the achieved statistical power in the T-Tests

for all hypothesis in which statistical hypothesis testing was possible. Since all hypotheses in Section 10.4 are two-tailed, the aim of T-Tests was to accept or reject the corresponding alternative hypotheses (i.e. there is no difference between groups). In accordance with common statistical practice [Tabachnik and Fidell 2010], in this chapter we use the following symbols whenever referring to statistical tests: " μ " is used to refer to a variable's mean, " σ " to refer to standard deviations, " t " and " df " to refer to the results and degrees of freedom of a statistical test, " p " to denote the significance level, as well as " d " and " η^2 " to refer to effect size and statistical power, respectively. Moreover, we assume a strict p -level of 0.05 to consider a result significant and assume small ($0.3 < d, \eta^2$), medium ($0.3 \leq d, \eta^2 < 0.8$), and large ($0.8 \leq d, \eta^2$) effect sizes and statistical power [Cohen 1992].

In Experiment 2 and 4, for some hypotheses, statistical hypothesis testing was not possible. This is due to the fact that in inspections, group responses were recorded, which do not produce a mean and no standard deviation. In these cases, comparative analyses were conducted on the group scores.

In the following subsections 11.1, 11.2, 11.3, and 11.4, we first state under which circumstance we accept results for each hypothesis as evidence in favor of Hazard Relation Diagrams. We then present the results from each experiment and, for each experiment, state if the hypothesis was accepted.

⁹ The results of Experiments 1 and 3 have been previously reported in [Tenbergen et al. 2017].

11.1. Hypothesis H1: Rationale Objectivity

We accept hypothesis H1 if participants more often mention contextual hazard information (i.e. “mitigation,” DV3.x, “trigger conditions,” DV4.x, or “safety goal,” DV5.x) in their rationales than diagram properties (i.e. “semantics,” DV1.x or “syntax,” DV2.x) in the treatment condition and vice versa in the control condition. There must be a significant difference between treatment and control conditions in the combination of DV1.x and DV2.x (in the following H1.xa) and the treatment group has a lower mean for H1.xa. At the same time, there must be a significant difference between in the combination of DV3.x, DV4.x, and DV5.x (in the following H1.xb) and the treatment condition has a higher mean for H1.xb.

11.1.1. Experiment 1

Table 11-1 summarizes means, standard deviations, T-Test, and power analysis results for all dependent variables pertaining to H1.1. Better performing groups with regard to the means in the dependent variables as well as significant T-Test results are printed in bold.

Table 11-1 Means, Std. Deviations, T-Test, and Power for H1.1.

Variable	Group	μ	σ	t	dF	p	d	η^2
DV1.1	treatment control	1.44 4.79	1.393 2.751	-8.36	118	< 0.001	1.536	1.000
DV2.1	treatment control	0.29 0.39	0.720 0.900	-0.71	118	0.481	0.123	0.172
DV3.1	treatment control	1.10 0.57	1.410 0.694	2.614	118	0.10	0.477	0.860
DV4.1	treatment Control	1.44 1.07	1.755 1.559	1.239	118	0.218	0.223	0.355
DV5.1	treatment Control	5.37 2.69	2.304 1.867	7.024	118	< 0.001	1.278	1.000
H1.1a`	treatment control	1.73 5.18	1.799 2.742	-8.13	118	< 0.001	1.488	1.000
H1.1b	treatment control	7.92 4.33	2.967 2.593	7,059	118	< 0.001	1.288	1.000

The significant differences between treatment and control group for rationales pertaining to diagram properties (H1.1a) and contextual hazard information (H1.1b) show that the alternative hypothesis must be rejected and hypothesis H1.1 can be accepted.

It is established that when using Hazard Relation Diagrams, adequacy judgments are more often based on contextual information about hazard. This is due to the fact that the mean for H1.1a was lower and the mean for H1.1b was higher in the treatment group than in the control group. In other words, whether or not the hazard was adequately mitigated is more likely to be judged based on objective information made available during hazard analyses, and less likely to be judged based on the subjective understanding of the diagram. Specifically, considering the effect sizes, Hazard Relation Diagram have a definitive positive impact mostly with regard

of conceptual mitigation (DV5.1) and completeness of trigger conditions (DV3.1). The effect of Hazard Relation Diagrams regarding the fulfillment of safety goals (DV4.1) is also positive, yet the effect is smaller. This indicates that from the concepts by which Hazard Relation Diagrams extend activity diagrams (see Section 5.1), the mitigation partition and trigger conditions are most effective to increase rationale objectivity. Since the purpose of the mitigation partitions is to highlight the affected hazard-mitigating requirements and since the purpose of trigger conditions is to act as a validation reference for operational conditions to be rendered sufficiently unlikely such that the hazard is less likely to be triggered during operation, we conclude that Hazard Relation Diagrams successfully highlight dependencies between hazard-mitigating requirements, hazards, their trigger conditions, and their safety goals and improve review results.

11.1.2. Experiment 2

A summary of the total scores in rationales provided by the treatment group and the control group is given in Table 11-2. Better performing groups with regard to hypothesis H1.2 are printed in bold and percentages from the total rationales given by each group are provided for the combined variables H1.2a and H1.2b.

Table 11-2 Rationales Categorization for each Inspected Conceptual Mitigation for H1.2.

Variable	Group	Adequately Mitigated			Inadequately Mitigated		Σ	
		Hazard 1	Hazard 2	Hazard 3	Hazard 4	Hazard 5		
DV1.2	treatment	1	1	1	2	3	8	
	control	2	0	3	4	4	13	
DV2.2	treatment	1	0	0	0	1	2	
	control	0	0	0	0	0	0	
DV3.2	treatment	4	1	2	0	0	7	
	control	0	0	0	0	0	0	
DV4.2	treatment	1	2	2	2	1	8	
	control	0	2	1	1	1	5	
DV5.2	treatment	1	1	3	1	1	7	
	control	1	1	1	0	0	3	
H1.2a	treatment	2	1	1	2	4	10	31.25%
	control	2	0	3	4	4	13	61.9%
H1.2b	treatment	6	4	7	3	2	22	68.75%
	control	1	3	2	1	1	8	38.1%

As can be seen, with the exception of Hazard 3, the treatment group provided less rationales mentioning diagram semantics for all conceptual mitigations. Furthermore, while the control group provided no rationales mentioning diagram syntax (DV2.2), the treatment group scored two such rationales. In total, the treatment group provided less rationales mentioning non-contextual information about the hazard than the control group ($H1.2a_{\text{treatment}} = 10$, $H1.2a_{\text{control}} = 13$), which can be seen as evidence in favor of hypothesis H1.2. Similarly, while no rationale was provided by the control group mentioning trigger conditions (DV4.2), the treatment group

provided seven such rationales. Yet, none of these rationales were given for inadequate conceptual mitigations (i.e. Hazard 4 and Hazard 5). For DV4.2, the treatment group uniformly provided more rationales mentioning the safety goal than controls did, scoring eight such rationales as opposed to five in the control group. Together with the larger number of rationales mentioning the conceptual mitigation in the treatment condition ($DV5.2_{\text{treatment}} = 8$, $DV5.2_{\text{control}} = 5$), this yielded a total number of 22 rationales mentioning contextual hazard information (H1.2b) for the treatment group and eight such rationales provided by the control group.

Despite the lack of statistical hypothesis testing like in Experiment 1, the results concerning hypothesis H1.2 can be seen as indications in favor of the ability of Hazard Relation Diagrams to increase validation objectivity in Fagan inspections. Since the treatment group collectively based their judgment more often on contextual information about the hazard and the control group inversely based their collective judgment more often on diagram semantics and diagram syntax. This effect becomes even more apparent when considering the percentages: The treatment group based 31.25% of their rationales on diagram semantics or diagram syntax and the majority of 68.75% on contextual information about the hazard. For the control condition, this relationship is approximately reversed. In conclusion, these results provide evidence that Hazard Relation Diagrams succeed in focusing the inspectors' rationales on the information uncovered during hazard analyses, thereby increasing objectivity when inspecting hazard-mitigating requirements.

11.1.3. Experiment 3

Table 11-3 summarizes means, standard deviations, T-Test, and power analysis results for all dependent variables pertaining to H1.3, with better performing conditions (with regard to the hypothesis) and significant T-Test results are printed in bold.

Table 11-3 Means, Std. Deviations, T-Test, and Power for H1.3.

Variable	Condition	μ	σ	t	dF	p	d	η^2
DV1.3	treatment control	1.52 2.74	1.087 1.318	-5.151	29	< 0.001	1.010	0.978
DV2.3	treatment control	0.00 0.07	0.000 .267	-1.795	29	0.161	n/a	n/a
DV3.3	treatment control	0.74 0.26	0.944 0.447	-2.359	29	0.009	0.660	0.761
DV4.3	treatment control	1.04 0.44	1.160 0.892	3.275	29	0.005	0.580	0.677
DV5.3	treatment control	1.96 1.52	1.285 0.975	2.249	29	0.063	0.386	0.403
H1.3a	treatment control	1.52 2.81	1.087 1.302	-5.677	29	< 0.001	1.076	0.988
H1.3b	treatment control	3.74 2.22	1.723 1.502	5.673	29	< 0.001	0.940	0.961

Results show that the means in the treatment condition for DV1.3, DV2.3, and the combined variable H1.3a are lower than in the control condition, indicating that review rationales were less often based on diagram semantics and diagram syntax whenever participants used Hazard Relation Diagrams to review conceptual mitigations as opposed to activity diagrams and hazard analysis worksheets. A pairwise T-Test shows that these differences between treatment condition and control condition was significant for DV1.3, DV3.3, DV4.3, H1.3a, and H1.3b. For DV5.3, the differences in means approaches significance.

The significant differences require hypothesis H1.3 to be accepted: there is a difference in rationales within participants when reviewing conceptual mitigations. Specifically, since in the treatment condition, more rationales were based on objective contextual information about the hazard, it is established that Hazard Relation Diagrams lead to adequacy judgements being based on contextual information about the hazard. This effect was large of very large especially with respect to diagram semantics and diagram syntax, where the treatment condition produced fewer such rationales, as well as trigger condition completeness and safety goal appropriateness. With regard to conceptual mitigation adequacy, the effect was less pronounced, but measurable. It can hence be concluded that integrating the contextual information about the hazard from the hazard analysis worksheets into activity diagrams, thereby creating Hazard Relation Diagrams is quite effective in focusing the reviewers' judgements on hazard analyses results as opposed to non-safety related information.

11.1.4. Experiment 4

Table 11-4 summarizes the total amount of rationales provided in the treatment condition and the control condition in both inspection sessions for each respective hazard. In contrast to Experiment 2, the order of presented stimuli was randomized for each session. Table 11-4 therefore shows the number of rationales provided for each hazard stimulus, referred to by its unique ID that was assigned prior to experimental onset (see Section 10.1). Hazard 1, 2, and 8 were stimuli containing adequate conceptual mitigations, and Hazard 4 through 6 containing inadequate conceptual mitigations. Furthermore, Table 11-4 sums up the rationales given for each stimulus and provides the percentages from the total rationales given in each condition in both sessions for the combined variables H1.4a and H1.4b. Better performing conditions with regard to the hypothesis are printed in bold.

Table 11-4 Rationales Categorization for each Inspected Conceptual Mitigation for H1.4.

Variable	Session	Condition	Adequately Mitigated			Inadequately Mitigated			Σ	
			Hazard 1	Hazard 2	Hazard 8	Hazard 4	Hazard 5	Hazard 6		
D1.4	1	treatment control	0	4			1		0	5
	2	treatment control	1	2		4	4	1	2	10
DV2.4	1	treatment control	0	0			0		0	0
	2	treatment control	0	0	0	0	0	0	0	0
DV3.4	1	treatment control	0	1			1		0	2
	2	treatment control	0	0	1	0	0	1	2	0
DV4.4	1	treatment control	0	0			0		0	0
	2	treatment control	1	0	1	0	0	0	2	0
DV5.4	1	treatment control	2	1			0		2	1
	2	treatment control	3	0	1	1	1	2	6	2
H1.4a		treatment control	1	6	0	4	5	1	2	14.28%
H1.4b		treatment control	6	2	3	1	1	3	12	85.71%
									4	21.05%

As can be seen, neither inspection session provided any rationales mentioning diagram syntax. Moreover, the total number of rationales mentioning diagram semantics was lower in the treatment condition than it was in the control condition for both inspection sessions (DV1.4). In consequence, the number of rationales mentioning diagram properties (H1.4a) is lower in the treatment condition than in the control condition. Regarding rationales mentioning trigger conditions (DV3.4), there is no clear superiority between conditions. While in Session 1, inspectors mentioned trigger conditions in two rationales in the control condition and zero in the treatment condition, the opposite was the case in Session 2. In Session 1, no rationale was provided mentioning safety goals in either condition (DV4.4), yet two such rationales were provided in the treatment condition in Session 2. For DV5.4, it can be seen from Table 11-4 that in both sessions, there were more rationales provided mentioning the conceptual mitigation in the treatment condition. Although in Session 1, there was only one more such rationale, there are three times as many rationales mentioning the conceptual mitigation in Session 2 in the treatment condition. In summary, the total number of rationales mentioning contextual information about the hazard (H1.4b) is considerably higher in the treatment condition with over 85% of all rationales provided belonging to that category.

Even though the absence of a standard deviation rendered hypothesis testing impossible, the result discussed above can be seen in favor of Hazard Relation Diagrams for hypothesis H1.4: Hazard Relation Diagrams appear to increase validation objectivity in Fagan inspections.

In the treatment condition, considerably more rationales mentioned contextual information about the hazard, indicating a higher degree of objectivity when judging the adequacy of conceptual mitigations. This effect was particularly pronounced with the conceptual mitigation: the majority of rationales in the treatment condition were provided for the conceptual mitigation, as opposed to trigger conditions or the safety goal. In summary, these results provide strong indications that Hazard Relation Diagrams foster adequacy judgements to be based on contextual information about the hazard as opposed to diagram properties.

11.2. Hypothesis H2: Effectiveness

In order for Hazard Relation Diagrams to have a positive influence on reviews, the treatment condition should score more correct answers (DV6.x and DV7.x) and fewer incorrect answers (DV8.x and DV9.x). Therefore, we accept hypothesis H2, if there is a significant difference between DV6.x through DV9.x, the mean in the treatment condition is higher for DV6.x and DV7.x than in the control condition (i.e. treatment condition scored more correct answers), and the mean in the treatment condition is lower in DV8.x and DV9.x than in the control condition (i.e. treatment condition scored fewer wrong answers).

11.2.1. Experiment 1

Table 11-5 summarizes means, standard deviations, T-Test, and power analysis results. Better performing groups are printed in bold.

Table 11-5 Means, Std. Deviations, T-Test, and Power for H2.1.

Variable	Group	μ	σ	t	dF	p	d	η^2
DV6.1	treatment	3.03	1.299	-1.23	118	0.222	0.226	0.362
	control	3.31	1.177					
DV7.1	treatment	3.17	1.289	-0.19	118	0.850	0.032	0.071
	control	3.21	1.226					
DV8.1	treatment	1.93	1.298	1.214	118	0.227	0.216	0.341
	control	1.66	1.196					
DV9.1	treatment	1.78	1.260	0.187	118	0.852	0.033	0.072
	control	1.74	1.196					

The control group scored more correct answers (DV6.1, DV7.1) and fewer wrong answers (DV8.1, DV9.1) than the treatment group. However, results from the T-Test indicate that none of the means in DV6.1 through DV9.1 were significant and effect sizes as well as statistical power was small for all tests except DV8.1.

The lack of significance for all dependent variable pertaining to hypothesis H2 in Experiment 1 shows that the alternative hypothesis must be accepted: There is no difference between groups with regard to the number of correctly or incorrectly identified adequate or inadequate

conceptual mitigations between both groups. On the one hand, this means that using Hazard Relation Diagrams for reviews does not improve effectiveness. On the other hand, this also means that the positive effect of Hazard Relation Diagrams on rationales is not achieved at the expense of reduced effectiveness, either.

11.2.2. Experiment 2

Table 11-6 shows the adequacy judgments for each conceptual mitigation for both inspection groups. Each conceptual mitigation was either judged as adequately or as inadequately mitigated. Depending on whether this judgment was correct for the specific conceptual mitigation (Hazards 1, 2, and 3 were in fact adequately mitigated, whereas Hazards 4 and 5 were in fact inadequately mitigated), a mark was placed into the appropriate cells for DV6.2 through DV9.2. For example, the “X” for the treatment group in DV6.2 for Hazard 2 means that this was an adequate conceptual mitigation and it was correctly assessed as such by the treatment group, hence rendering a true positive response. For another example, the “X” for the control group in DV9.2 for Hazard 4 means that this was an inadequate conceptual mitigation, which was incorrectly judged as adequate by the control group, hence rendering a false negative response. The rightmost column sums the number of marks per variable, better performing groups indicated in bold.

Table 11-6 Judgement Effectiveness for H2.2.

Variable	Group	Adequately Mitigated			Inadequately Mitigated		Σ
		Hazard 1	Hazard 2	Hazard 3	Hazard 4	Hazard 5	
DV6.2	treatment		X	X			2
	control	X	X	X			3
DV7.2	treatment					X	1
	control					X	1
DV8.2	treatment	X					1
	control						0
DV9.2	treatment				X		1
	control				X		1

Results show that both groups performed about equally, with a slight advantage for the control group: The control group managed to correctly assess all adequate conceptual mitigations, whereas the treatment group incorrectly judged Hazard 1 as inadequately mitigated, yielding one more false positive answer (and therefore, inversely, one less true positive answer). However, both treatment group and control group correctly judged one inadequate conceptual mitigation (Hazard 5) and missed the other (Hazard 4). Unfavorable for hypothesis H2.2, the control group performed better than the treatment group. This might be explained by diagram semantics: The functional requirements for Hazard 1 contained a deliberate semantic mistake in the decision node (see Figure 2-1), which was noticed by both the treatment group and the control

group. However, the treatment group decided that this semantic mistake would render the entire ACC unfit for operation, regardless if it is safe or not. While the control group ascertained the same semantic mistake, they decided to continue under the assumption that this was an oversight during modeling and judged that it had no impact on safety. Similarly, both treatment and control group noticed a nonsense safety goal in Hazard 4 and decided that albeit the safety goal was semantically wrong, the conceptual mitigation is adequate (with regard to the safety goal), hence incorrectly arriving at the conclusion that the hazard was adequately mitigated. In consequence, it can be concluded that using Hazard Relation Diagrams for inspections only minimally impacts effectiveness.

11.2.3. Experiment 3

Table 11-7 summarizes means, standard deviations, T-Test results, and power analysis results and depicts better performing conditions with respect to H2.3 and significant results in bold.

Table 11-7 Means, Std. Deviations, T-Test, and Power for H2.3.

Variable	Condition	μ	σ	t	dF	p	d	η^2
DV6.3	treatment control	0.70 0.51	0.318 0.325	3.077	56	0.011	0.614	0.720
DV7.3	treatment control	0.70 0.43	0.282 0.359	2.718	56	< 0.001	0.860	0.929
DV8.3	treatment control	0.30 0.49	0.318 0.325	-3.112	56	0.011	1.547	0.999
DV9.3	treatment control	0.30 0.57	0.282 0.359	-3.225	56	< 0.001	0.860	0.929

Results show that for adequately mitigated hazards (i.e. DV6.3 and DV8.3), participants correctly judged conceptual mitigations more often when using Hazard Relation Diagrams, while making incorrect judgments less often. This performance is superior to the control condition. The same relationship holds for inadequately mitigated hazards. Interestingly, participants averaged at the same success and failure rate in the treatment condition, but were slightly more likely to make incorrect judgment in the control condition. T-Tests show significance for all these differences. Moreover, Cohen's d shows large and very large effect size and power.

These results provide compelling evidence for accepting hypothesis H2 in Experiment 3: there is a significant difference in effectiveness within participants between the treatment condition and the control condition when reviewing conceptual mitigations and using Hazard Relation Diagrams for review results in higher effectiveness.

11.2.4. Experiment 4

Table 11-8 summarizes the adequacy judgments for each unique conceptual mitigation for both conditions and both sessions. Like in Experiment 2 (see Section 11.2.2), an “X” in a respective field indicates an answer. For example, the “X” for Hazard 2 in the control condition for Session 2 in DV9.4 means that Hazard 2 was adequately mitigated, but the inspectors in Session 2 considered it to be an inadequate conceptual mitigation, thereby rendering a false negative answer. The rightmost column sums up the total amount of answers per variable, condition, and session; better performing conditions are printed in bold.

Table 11-8 Judgement Effectiveness for H2.4.

Variable	Session	Condition	Adequately Mitigated			Inadequately Mitigated			Σ
			Hazard 1	Hazard 2	Hazard 8	Hazard 4	Hazard 5	Hazard 6	
DV6.4	1	treatment	X						1
		control							0
	2	treatment	X		X			X	3
		control							0
DV7.4	1	treatment							0
		control					X		1
	2	treatment							0
		control							0
DV8.4	1	treatment							0
		control							0
	2	treatment							0
		control					X		1
DV9.4	1	treatment							0
		control		X					1
	2	treatment							0
		control		X		X			2

As can be seen, inspectors performed better across the board in the treatment condition: In both sessions, using Hazard Relation Diagrams to inspect conceptual mitigation adequacy lead to more correct answers (DV6.4 and DV7.4) than when using conventional activity diagrams and hazard analysis worksheets. Moreover, it is noteworthy, that the treatment condition, neither inspection session made a single mistake, leading to a total score of zero for both sessions in (DV8.4 and DV9.4). On the other hand, in the control condition, participants in Session 1 incorrectly judged one hazard, while in Session 2, half the conceptual mitigations were misjudged in the control condition (i.e. three out of six inspected conceptual mitigations).

Results show that both sessions performed about equally. However, due to the fact that in the treatment condition, neither inspection group made a single mistake, these results indicate that using Hazard Relation Diagrams to inspect conceptual mitigations leads to higher effectiveness, which is evidence in support for hypothesis H2.4. These results are supported by the fact that in Session 2, half of the judgements were incorrect (i.e. false positives or false nega-

tives). Albeit in Session 1, there was a higher amount of overall correct answers, it is conceivable that using conventional activity diagrams and hazard analysis worksheets for untrained participants (considering that participants in Session 2 were recruited from a human factors class as opposed to a software quality class, see Section 10.8.2) results in adequacy judgment correctness to be determined by random chance. Based on these results, Hazard Relation Diagrams seem to have a positive impact on inspection effectiveness for untrained participants.

11.3. Hypothesis H3: Efficiency

We accept hypothesis H3, if the means of treatment condition are significantly lower than the means of the control condition in the response times (for Experiment 1 and 3) or the group effort (Experiment 2 and 4) for true positive answers (DV10.x), true negative answers (DV11.x), false positive answers (DV12.x), and false negative answers (DV13.x) as well as the overall efficiency (in the following H3.xtotal).

11.3.1. Experiment 1

Table 11-9 summarizes the results pertaining to hypothesis H3.1. Superior group performance and significant T-Test results depicted using bold font.

Table 11-9 Means, Std. Deviations, T-Test, and Power for H3.1.

Variable	Group	μ	σ	T	dF	p	d	η^2
DV10.1	treatment	248.17	157.124	-1.92	116	0.05	0.353	0.646
	control	305.65	168.001					
DV11.1	treatment	326.38	224.416	-0.97	116	0.336	0.178	0.266
	control	373.48	298.171					
DV12.1	treatment	176.36	163.282	-0.66	116	0.508	0.123	0.172
	control	199.20	206.925					
DV13.1	treatment	187.24	212.735	-0.72	116	0.641	0.133	0.118
	control	214.93	203.705					
H3.1total	treatment	938.16	418.790	-1.89	116	0.281	0.348	0.635
	control	1093.27	470.493					

The mean response time for all dependent variables DV10.1 through DV11.3 was lower in the treatment group than in controls. This means the treatment group was able to correctly detect adequate (true positive answers, DV10.1) and inadequate (true negative answers, DV11.1) conceptual mitigations more quickly than the control group. Furthermore, the treatment group had a lower response time for incorrect answers (DV12.1 and DV13.1). T-Tests only revealed difference in means for true positive answers (DV10.1). Post-hoc power analyses on the T-Test results revealed a medium effect size and statistical power for DV10.1 and H3.1total as well as a small effect size and low statistical power for all other variables.

In light of the absence of significance in the differences between the treatment and control groups for variable H3.1total, the hypothesis for H3 must be rejected for Experiment 1. In other words, there is no significant difference between the two groups with regard to the time needed to review hazard-mitigating requirements using Hazard Relation Diagrams. Nevertheless, there is a clear trend for Hazard Relation Diagrams to positively influence efficiency, given that the response times were consistently faster in the treatment group for all dependent variables.

11.3.2. Experiment 2

Table 11-10 compares the efficiency of both groups for DV10.2, DV11.2, DV12.2, and DV13.2, with more efficient groups depicted in bold.

Table 11-10. Judgement Efficiency for H3.2.

Variable	Group	No. of participants	Time	No. of Answers	Efficiency
DV10.2	treatment	15	90	2	1.481
	control	25		3	1.333
DV11.2	treatment	15	90	1	0.741
	control	25		1	0.444
DV12.2	treatment	15	90	1	0.741
	control	25		0	-
DV13.2	treatment	15	90	1	0.741
	control	25		1	0.444
H3.2total	treatment	15	90	5	3.704
	control	25		5	2.222

As can be seen, despite the treatment group having identified one true positive conceptual mitigation less than the control group (DV10.2), their efficiency remained superior due to the fact that the treatment group consisted of fifteen participants as compared to the 25 controls. In consequence, the efficiency for true negative and false negative answers (DV11.2 and DV13.2, respectively) is higher in the treatment condition than in the control condition as well. For false positive answers in DV12.2, no comparison can be made since control group gave no such response.

Albeit these results are in favor of hypothesis H3.2 and thereby in favor of Hazard Relation Diagrams, it must be noted that these results are impacted the unequal group sizes: Since the group size could not be controlled for due to the self-selection process into treatment and control conditions, which was administratively necessary in order to avoid scheduling conflicts, the treatment group had less participants than in the control condition, thereby increasing efficiency in favor of Hazard Relation Diagrams. In a within-subjects design, such an issue would not have impacted the results, as the same number of participants would inspect treatment stimuli as well as control stimuli. Nevertheless, inspection efficiency using Hazard Relation Diagrams did not decrease as compared to activity diagrams.

11.3.3. Experiment 3

Table 11-11 summarizes the results pertaining to hypothesis H3.3. Superior group performance is depicted using bold font.

Table 11-11 Means, Std. Deviations, T-Test, and Power for H3.3.

Variable	Condition	μ	σ	t	dF	p	d	η^2
DV10.3	treatment	68.95	60.171	-1.349	22	0.366	0.020	0.158
	control	78.56	30.871					
DV11.3	treatment	89.55	54.360	-0.870	18	0.349	0.277	0.203
	control	103.67	47.518					
DV12.3	treatment	76.50	60.078	1.043	13	0.406	0.242	0.148
	control	63.87	42.804					
DV13.3	treatment	85.31	66.861	0.188	15	0.781	0.103	0.085
	control	79.63	40.052					
H3.3total	treatment	86.34	38.485	0.497	28	0.374	0.156	0.140
	control	81.11	27.639					

As can be seen in Table 11-11, response times for correctly judged adequate (DV10.3) and inadequate (DV11.3) conceptual mitigations was faster in the treatment condition than in the control condition. However, response times in the control condition were faster than in the treatment condition for incorrectly judged conceptual mitigations, both adequate and inadequate (DV12.3 and DV13.3, respectively). Furthermore, the overall response time (H3.3total) was on average faster in the control condition as well. However, it can be seen, that response times only differed by five to fourteen seconds for all variables, indicating that the average time to review any stimulus was only marginally impacted by experimental condition or conceptual mitigation adequacy. This is supported by the relatively high standard deviations for all variables. In consequence, none of the pairwise T-Tests indicated any significance of the mean response times for the variables. Furthermore, effect size and statistical power was low for all variables as well.

Due the lack of significance in the difference of response times in both conditions for variable H3.3total, we must reject hypothesis H3.3. In other words, there is no significant difference with regard to the time needed to review hazard-mitigating requirements using Hazard Relation Diagrams as opposed to conventional activity diagrams and hazard analysis worksheets. However, it is noteworthy that when using Hazard Relation Diagrams for review, participants were faster for all correct judgements (i.e. DV10.3 and DV11.3), but slower when they incorrectly judged a conceptual mitigation (i.e. DV12.3 and DV13.3).

11.3.4. Experiment 4

Table 11-12 compares the efficiency of both inspection sessions for both the treatment condition and the control condition for true positive, true negative, false positive, and false negative adequacy judgments. More efficient conditions are depicted in bold.

Table 11-12 Judgement Efficiency for H3.4.

Variable	Session	Condition	No. of participants	Time	No. of Answers	Efficiency
DV10.4	1	treatment	24	50	1	0.833
		control			0	-
	2	treatment	27	75	3	1.481
		control			0	-
DV11.4	1	treatment	24	50	0	-
		control			1	0.833
	2	treatment	27	75	0	-
		control			0	-
DV12.4	1	treatment	24	50	0	-
		control			0	-
	2	treatment	27	75	0	-
		control			1	0.494
DV13.4	1	treatment	24	50	0	-
		control			1	0.833
	2	treatment	27	75	0	-
		control			2	0.988
H3.4correct		treatment	25.5	62.5	4	2.510
		control			1	0.627
H3.4incorrect		treatment	25.5	62.5	0	-
		control			4	2.510

Due to the fact that the order of inspected conceptual mitigations was randomized, there was an unequal amount of control and treatment stimuli, and, for inspection Session 1, also an unequal amount of adequate and inadequate stimuli. Therefore, efficiency measurements did not yield a result for all combinations of sessions, condition, and conceptual mitigation adequacy. Nevertheless, it can be seen that efficiency in the treatment condition ranges between 0.833 and 1.481, while efficiency in the control condition has a lower range between 0.494 and 0.988 over all variables and inspection sessions. This observation especially holds for correct judgments (i.e., true positive and true negative answers, see DV10.4 and DV11.4, respectively). As we have outlined in Section 11.2.4, neither inspection session has a false positive or false negative answer in the treatment condition. It was therefore not possible to compute the efficiency for DV12.4 and DV13.4 in the treatment condition, such that a comparison of efficiency between for incorrect judgements cannot be made. In consequence, a comparison of the efficiency for the combined variable H3.4incorrect can also not be made. However, it can be seen that with regard to correct judgements, higher efficiency was achieved in the treatment condition ($H3.4correct_{treatment} = 2.510$ and $H3.4correct_{control} = 0.627$).

Albeit efficiency for the combined variable H3.4correct is higher in the treatment condition than in the control condition, we hesitate to consider this evidence in favor of Hazard Relation Diagrams positively impacting inspection efficiency. Due to the within-subjects nature of Experiment 4, group size and available time for the inspection sessions were not a factor that impacted different results. Nevertheless, we believe that the one sided effectiveness measure in hypothesis H2.4 (see Section 11.2.4) may have caused a ceiling effect [Wohlin et al. 2012] that skewed results such that a more detailed analysis of differences in efficiency became moot. Instead, these results can at best be seen as indications, not as evidence in favor of H3.4, but we cannot definitively accept nor reject this hypothesis.

11.4. Hypothesis H4: Self-Reported Confidence

In Experiment 1 and 2, we expected a difference between treatment condition and control condition regarding the participant's self-reported confidence in their own adequacy judgment when validating conceptual mitigations using Hazard Relation Diagrams for hypothesis H4. We assumed that Hazard Relation Diagrams increase review confidence and lead to a higher mean for the treatment condition in the quality foci with high post-Cronbach cohesion, i.e. DV14.x, DV15.x, DV16.x, DV18.x, and DV21.x. Moreover, we assumed the overall degree of confidence (in the following H4.xtotal) is also higher in the treatment condition than in the control condition.

In contrast to Experiments 1 and 2, we measured self-reported confidence in Experiment 3 and 4 by means of a semantic differential (see Section 10.7.1). Since every participant answered based on their personal preference, answers cannot be differentiated by means of groups or conditions, like in Experiments 1 and 2. Therefore, it was neither possible to run T-Tests nor power analyses on the results pertaining to hypothesis H4.x. Instead, we accept results as indications in favor of Hazard Relation Diagrams, if the retained quality foci 3 (see Table 10-10) as well as the overall variable H4.xtotal indicate tendencies towards the ordinate "5: true for Hazard Relation Diagrams." We accept evidence in favor of activity diagrams and hazard analysis worksheets if these dependent variables indicate tendencies towards the ordinate "1: true for activity diagrams."

11.4.1. Experiment 1

Table 11-13 summarizes the results with higher rating groups and significant results printed in bold.

Table 11-13 Means, Std. Deviations, T-Test, and Power for H4.1.

Variable	Group	μ	σ	t	dF	p	d	η^2
DV14.1	treatment control	3.60 3.48	0.786 0.880	0.944	130	0.347	0.151	0.217
DV15.1	treatment control	3.76 3.52	0.629 0.824	2.001	130	0.047	0.323	0.580
DV16.1	treatment control	2.82 2.64	0.472 0.618	2.001	130	0.552	0.322	0.577
DV18.1	treatment control	3.50 3.19	0.878 0.876	2.228	130	0.027	0.356	0.651
DV21.1	treatment control	2.90 3.49	0.840 1.039	-3.881	130	< 0.001	0.625	0.973
H4.1total	treatment control	3.41 3.25	0.491 0.588	1.900	130	0.05	0.301	0.538

Results indicate that the means was higher in the treatment group for the quality foci “Perceived Usefulness,” (DV14.1) “Computer Self-Efficacy,” (DV15.1) “Results Demonstrability,” (DV16.1) and “Meaning” (DV18.1) while at the same time reporting less confusion (DV21.1). Moreover, the independent samples T-Test reveals that the differences in between-group means were significant for DV15.1, DV18.1, DV21.1, and the combined variable H4.1total, yet not significant for any other variable. The post-test power analysis revealed a medium effect size and very high statistical power for the significant difference in DV21.1. Medium effect size and statistical power became apparent for the other significant variables DV15.1, DV18.1, and H4.1total as well as for the non-significant variable DV16.1. A small effect size and low statistical power was revealed for DV14.1.

In light of these results, we can accept hypothesis H4 in Experiment 1: participants using Hazard Relation Diagrams to review hazard-mitigating requirements report higher subjective confidence compared with participants using conventional activity diagrams and hazard analysis worksheets.

11.4.2. Experiment 2

Table 11-14 summarizes means, standard deviations, T-Test, and power analysis results for hypotheses H4.2 with higher rating groups printed in bold.

Table 11-14 Means, Std. Deviations, T-Test, and Power for H4.2.

Variable	Group	μ	σ	T	dF	p	d	η^2
DV14.2	treatment	3.83	0.595	1.313	38	0.197	0.437	0.371
	control	3.55	0.696					
DV15.2	treatment	3.97	0.452	0.696	38	0.491	0.239	0.177
	control	3.81	0.708					
DV16.2	treatment	2.97	0.339	0.696	38	0.491	0.240	0.178
	control	2.86	0.531					
DV18.2	treatment	3.63	0.694	0.309	38	0.273	0.369	0.296
	control	3.36	0.784					
DV21.2	treatment	3.23	0.753	-0.985	38	0.331	0.332	0.259
	control	3.52	0.963					
H4.2total	treatment	3.64	0.406	1.212	38	0.233	0.400	0.330
	control	3.47	0.428					

A comparison of the means for these variables shows a higher mean in the treatment group than in the control group for DV14.2, DV15.2, DV16.2, DV18.2, and H4.2total and a lower mean DV21.2. Independent groups T-Tests did not indicate significance for these differences in either variable. The post-test power analysis revealed a medium effect size and medium statistical power for the differences in DV14.2 and H4.2total, and a medium effect size with low statistical power for DV21.2. The power analysis also indicated low effect sizes and statistical power for the remaining variables.

Given the T-Test results, we must reject hypothesis H4.2 in favor of the alternative hypothesis: There is no significant difference in self-reported confidence between participants using Hazard Relation Diagrams and participants using conventional activity diagrams and hazard analysis worksheets to inspect hazard-mitigating requirements. A possible explanation of the lack of significance may lie in the overall small number of participants for both groups and in the high sensitivity of Student's T to unequal group sizes [Tabachnik and Fidell 2010]: Due to the fact that participants were sampled at convenience from a graduate class, the voluntary nature of participation, and self-selection into treatment and control condition, equality of group sizes could not be controlled for, resulting in $n_{\text{treatment}} = 15$ and $n_{\text{control}} = 25$. Nevertheless, clear positive trend can be observed: Just like in Experiment 1, the participants in the treatment group reported higher perceived usefulness, self-efficacy, and result demonstrability, less confusion, and higher understandability of the information depicted in Hazard Relation Diagrams than control participants did for conventional activity diagrams and hazard analysis worksheets.

11.4.3. Experiment 3

Table 11-15 summarizes the means for each variable. In Figure 11-1, the frequencies of responses for each ordinate and each retained quality focus is shown. Darker bars indicate stronger preference for Hazard Relation Diagrams.

Table 11-15 Means and Std. Deviations H4.3

Variable	μ	σ
DV14.3	2.79	1.11
DV16.3	2.93	0.92
DV18.3	2.54	1.11
DV19.3	2.44	1.24
DV21.3	3.41	0.89
H4.3total	2.83	0.78

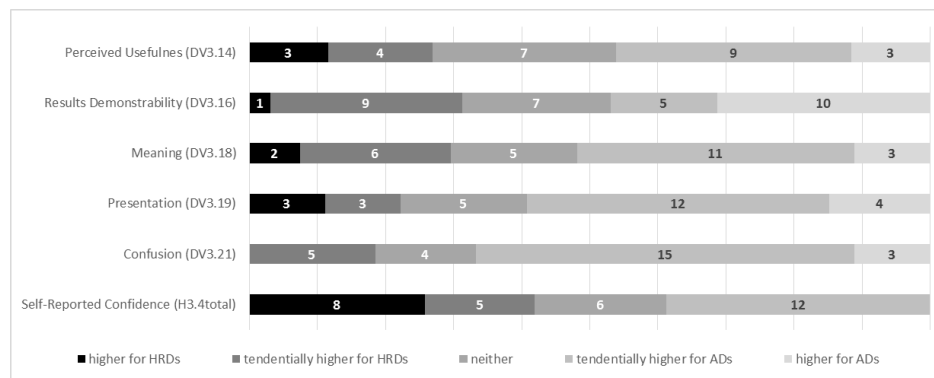


Figure 11-1 Distributions over all Ordinates of the Retained Quality Foci for H4.3.

As can be seen in Table 11-15, the means for “Perceived Usefulness,” “Result Demonstrability,” “Meaning,” “Presentation,” and “Confusion“ (DV14.3, DV16.3, DV18.3, DV19.3, and DV21.3, respectively) are around the neutral ordinate of “3: indifference.“ It can also be seen that the standard deviation was for all variables around one full ordinate, indicating that the mean difference between participants was between the ordinate “4: tendentially true for Hazard Relation Diagrams” and the ordinate “2: tendentially true for activity diagrams”. Results also show that participants report a slight preference for activity diagrams with regard to the variables “Perceived Usefulness” (DV14.3), “Result Demonstrability” (DV16.3), “Meaning” (DV18.3), and “Presentation” (DV19.3), and reported slightly more confusion with Hazard Relation Diagrams (DV21.3). With regard to the overall variable for self-reported subjective confidence H4.3total, participants indicate a preference for activity diagrams and hazard analysis worksheets.

The fact that the means for each variable were consistently around the neutral ordinate with a fairly large standard deviation suggest that we must reject hypothesis H4.3: there is no difference in subjective confidence within participants when using Hazard Relation Diagrams or when using conventional activity diagrams with hazard analysis worksheets to review conceptual mitigations.

11.4.4. Experiment 4

Table 11-16 summarizes the means for each retained quality focus. For each ordinate, the response frequency over both inspection sessions are shown in Figure 11-2, where darker bars indicate stronger preference for Hazard Relation Diagrams.

Table 11-16 Means and Std. Deviations H4.4.

Variable	μ	σ
DV14.4	3.75	1.23
DV16.4	3.54	1.08
DV19.4	3.76	1.35
H4.4total	3.65	1.13

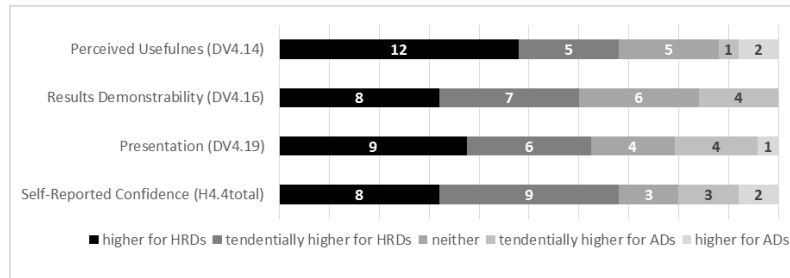


Figure 11-2 Distributions over all Ordinates of the Retained Quality Foci for H4.4.

For this hypothesis, answers from the post-hoc questionnaire from both inspection sessions have been combined. After excluding invalid questionnaire responses (i.e. pattern-like answers, only single-ordinate answers, or blank answers, see Section 10.9), this yielded a total of 25 completed questionnaires. Cronbach’s α was computed for all quality foci over the responses from valid questionnaires. Only three quality foci from Table 10-10 were retained for Experiment 4. This was due to the fact that “Perceived Usefulness,” “Result Demonstrability,” and “Presentation” were the only three TAM3 and TTF items that had a high questionnaire cohesion. From the means for the variables representing these quality foci, i.e. DV14.4, DV16.4, and DV19.4, respectively, are approaching, but do not reach the ordinate “4: tententially true for Hazard Relation Diagrams.” However, it can also be seen that the standard deviation for all variables was at least one full ordinate, indicating that the mean difference between participants was between the ordinates 2/3 and 4/5. These results show participants’ tentential preference for Hazard Relation Diagrams over conventional activity diagrams with hazard analysis worksheets when inspecting conceptual mitigations.

Considering the fact that questionnaire responses are strongly approaching ordinate “4: tententially true for Hazard Relation Diagrams,” the results suggest that participants tententially consider Hazard Relation Diagrams more useful during inspections than activity diagrams and hazard analysis worksheets and that participants’ inspection judgments were easier to demonstrate and present using Hazard Relation Diagrams as well. This is also reflected in the combined variable H4.4total, suggesting that participants felt subjectively more confident with their inspections when judging the adequacy of conceptual mitigations using Hazard Relation Diagrams. Yet, this is at best a tentential indication in favor of accepting hypothesis H4.4.

12. Discussion of the Results of the Empirical Evaluation

In this chapter, we critically discuss the results of the empirical evaluation. In Section 10.1, several research questions were defined and in Section 10.4, hypotheses were defined to investigate these research questions. These hypotheses are summarized in Table 12-1. In the following, the results with regard to these hypotheses are summarized.

Table 12-1 Summary of Experimental Results, Hypotheses and Research Questions

Hypothesis		Experiment 1	Experiment 2	Experiment 3	Experiment 4
		Hypothesis Accepted			
		<i>Superior Performance of HRD/AD</i>			
H1	There is a difference in rationales between treatment condition and control condition regarding adequacy judgments when validating hazard-mitigating requirements.	yes	yes	yes	yes
		<i>HRD</i>	<i>HRD</i>	<i>HRD</i>	<i>HRD</i>
H2	There is a difference in effectiveness between treatment condition and control condition when validating hazard-mitigating requirements.	no	(no hypothesis testing)	yes	(no hypothesis testing)
		<i>AD</i>	<i>AD</i>	<i>HRD</i>	<i>HRD</i>
H3	There is a difference in efficiency between treatment condition and control condition when validating hazard-mitigating requirements.	no	(no hypothesis testing)	no	(no hypothesis testing)
		<i>HRD</i>	<i>HRD</i>	<i>HRD</i>	<i>HRD</i>
H4	There is a difference in self-reported confidence between treatment condition and control condition when validating hazard-mitigating requirements.	yes	no	(no hypothesis testing)	(no hypothesis testing)
		<i>HRD</i>	<i>HRD</i>	<i>AD</i>	<i>HRD</i>

12.1. Hypothesis H1: Rationales Objectivity

Hypothesis H1 aimed at investigating the impact on validation judgments by measuring differences in rationale. In all experiments, the treatment condition provided considerably more rationales mentioning contextual information about the hazard as opposed to diagram properties. In Experiment 1 and Experiment 3, these results are significant for all combined variables (see Sections 11.1.1 and 11.1.3, respectively). In particular, the modeling concepts “mitigation partition” and “trigger condition” are the most pertinent to this effect, while there does not seem to be a significant effect on “safety goal” or “diagram syntax.” Yet, these results do not indicate whether or not this effect is due to the combination of modeling elements in Hazard Relation Diagrams or due to the mitigation partitions and trigger conditions alone. Further investigation is needed to ascertain whether this effect also holds when safety goals are not present in the diagram, i.e. whether or not surrounding the changed portions of the diagram with a dashed line is sufficient as long as trigger conditions are present. Nevertheless, these results suggest that the positive impact of Hazard Relation Diagrams on the rationales seems to be unaffected by the experience level of validators. Albeit experimental populations were roughly equivalent with regard to their levels of experience (see Section 10.10), repetition is necessary with, for

example, industry representatives. Nevertheless, it can be confidently concluded that Hazard Relation Diagrams improve the stakeholders' ability to judge the adequacy of hazard-mitigating requirements based on the information from hazard analysis worksheets when validating hazard-mitigating requirements.

12.2. Hypothesis H2: Effectiveness

Experiment 1 revealed negligibly better effectiveness in control participants during reviews, as they scored more correct answers while making fewer mistakes than the treatment group. However, these differences were insignificant and the measured effect was small in all instances (see Section 11.2.1). In Experiment 3, these results were reversed: when using Hazard Relation Diagrams for reviews, participants scored significantly more correct answers than in the control condition (see Section 11.2.3). These results can be explained by the experimental design: while in Experiment 1, participants were allowed to complete the experiment at home in an uncontrolled environment, we controlled the experimental conditions more strictly in Experiment 3 by inviting participants to complete the experiment in a controlled environment during a class meeting. Considering the lack of significance in all variables, we can assume that the mode of experimentation between Experiment 1 and Experiment 3 caused participants in Experiment 1 to be distracted, impacting their focus and motivation, and lead to equal effectiveness in between Hazard Relation Diagrams and activity diagrams Experiment 1. In light of this possibility, the significant results in Experiment 3 represent more compelling evidence in favor of Hazard Relation Diagrams. Moreover, effect sizes were high or very high for all variables, which increases confidence in these results.

In Experiment 2, effectiveness in both inspection sessions was about equal, however with a negligible advantage for the control groups due to the fact that the treatment groups in total had one more incorrect answer (see Section 11.2.2). Again, in Experiment 4, this effect was reversed and participants performed better in the treatment condition than in the control condition (see Section 11.2.4). In consequence, with regard to validation mode, we have evidence in favor and against Hazard Relation Diagrams to improve effectiveness in inspection sessions, albeit it is to note that evidence against Hazard Relation Diagrams lack significance in Experiment 1.

12.3. Hypothesis H3: Efficiency

Results regarding efficiency in Experiment 1 indicate that there is no significant difference between groups, albeit all response times were faster in the treatment group (see Section 11.3.1). Similar results were found in Experiment 3, where participants were faster when answering

correctly, but slower when answering incorrectly (see Section 11.3.3). These findings are in line with the results from Experiment 2 and 4, where the treatment condition outperformed the control condition in terms of efficiency in all cases (see Section 11.3.2 and Section 11.3.4, respectively). However, it must be noted that the efficiency in Experiment 2 was impacted by differences in group size between treatment group and control group, which may have resulted in better performance of the treatment group to a higher degree than the higher effectiveness in the control group could alleviate. Albeit in Experiment 4, differences in group size did not impact efficiency measures, the fact that in the treatment condition, not a single incorrect answer was given may have resulted in a ceiling effect. Considering the lack of significance for Experiment 1 and 3 and due to the impact on group size and effectiveness metrics in Experiments 2 and 4, we must conclude that there is no significant difference when validating hazard mitigating using Hazard Relation Diagrams. Nevertheless, it must be pointed out that in all experiments, albeit not statistically significant, an improvement in efficiency using Hazard Relation Diagrams became apparent.

12.4. Hypothesis H4: Self-Reported Confidence

The results of the post-hoc questionnaire from Experiment 1 show that the treatment group rated their confidence during reviews consistently higher than the control group, revealing an overall significant effect. In particular, participants in the treatment condition rated their ability to justify adequacy judgements and understand the modeling concepts contained in the diagrammatic representation significantly higher than controls and self-reported significantly less confusion about the review task than control participants (see Section 11.4.1). Results from individual participant answers on the same questionnaire in Experiment 2 indicate similar results, with treatment participants self-reporting consistently higher confidence for all quality foci than control participants (see Section 11.4.2). Results from Experiment 4 are in line with these findings: participants reported a preference towards Hazard Relation Diagrams when inspecting conceptual mitigations and perceived Hazard Relation Diagrams to be more useful and presentable and preferred demonstrability of inspection results using Hazard Relation Diagrams over conventional activity diagrams (see Section 11.4.4). Contrastingly, however, questionnaire results from Experiment 3 did not reveal such a strong effect. Instead, participants reported a slight preference for conventional activity diagrams (see Section 11.4.3). These results are surprising, as participants' self-reported lower confidence using Hazard Relation Diagrams for reviews contrasts with the results pertaining to rationales, effectiveness, and efficiency in Experiment 3 (see Sections 11.1.3, 11.2.3, and 11.3.3). This could be explained with the fact that

activity diagrams were a more recent topic in the course from which participants were recruited. The topic of activity diagrams was recently completed in the course just before Experiment 3 was conducted, such that participants preferred more familiar diagrammatic representations. In Experiment 1, activity diagrams were a topic earlier in the semester, and in Experiment 2 and 4, activity diagrams were introduced for the purpose of the experiment. Yet, the current experimental investigation nor the participants' self-reported levels of experience (which were comparable between all experiments, see Section 10.10) do not conclusively support such conjecture. The results from Experiment 3 notwithstanding, Experiment 1, 2, and 4 show a positive influence on participants' self-reported subjective confidence for validation at large when Hazard Relation Diagrams to validate the adequacy of conceptual mitigations, regardless of participant demographics.

13. Threats to Validity

Albeit utmost care was taken to design the experiments, several threats to validity remain. These are discussed in the order suggested by [Wohlin et al. 2012] in the following.

Internal Validity

One issue for the study at hand is the suitability of the between-subjects design and the experimental procedure. To gain confidence in the design and experimental set-up in general, we conducted a pilot test (reported in [Tenbergen et al. 2015]) that led to several improvements of the study. Another issue which may impair the internal validity is that participants were conveniently sampled from undergraduate and graduate university courses and had limited experience in the subject matter. “Conveniently sampled” in this sense means that students were recruited from courses where the authors had easiest access to and where students could be motivated to participate the easiest. Albeit the use of undergraduate students in Experiment 1 in this sense is somewhat controversial [Carver et al. 2003], we adhered to the best practices in this matter [Hart et al. 2000; Carver et al. 2008] to reduce this threat, while at the same time served to motivate students. This included integrating the experiments as a learning opportunity into the courses from which were sampled, allowing for student feedback on the effectiveness of experiments to facilitate learning, allowing for student introspection about the post-experimental learning progress post, and providing a detailed debriefing concerning the purpose of the study, including sharing preliminary data with participants to foster learning. Moreover, Experiments 2 and 4 were conducted using graduate student participants, which have previously been reported to be comparable to practitioners in regard to introspective measures (such as statement objectivity, response time, or self-reported metrics, see [Sjøberg et al. 2005] for an overview).

Another issue that impacts the internal validity of the empirical evaluation is the difference in the mode of experimentation between Experiment 1 and Experiment 3. In Experiment 1, students completed the experiment at home, in an uncontrolled setting, while in Experiment 3, students participated in the experiment in a class meeting. In consequence, there is the possibility that participants in Experiment 1 were distracted, did not entirely focus on participation, and may have had a lower level of motivation than in Experiment 3. This may have impacted their performance and skewed results. We accounted for this possibility during data preparation (see Section 10.9) and strictly excluded potentially skewed results. Furthermore, we took a very conservative approach to present and discuss the results in Chapter 11 in light of differences in

mean, significance, and statistical power. Nevertheless, a remaining influence of the uncontrolled factors in Experiment 1 as a threat to internal validity must be acknowledged. In fact, this issue was one of the key motivators for us to conduct Experiments 3.

Language ability may also have had a small influence on the results of Experiment 1 and Experiment 2, as the experimental stimuli were in English, but participants were mainly German speaking. However, we are confident that this impact is negligible due to the generally high proficiency of German university students in written and spoken English. Nevertheless, we combatted this threat by allowing for questions pertaining to technical terminology used in the experimental stimuli during the briefing session.

Construct Validity

The experimental material must be suitable to measure the desired effect. Errors or bias within the material must not influence the experimental results. In our experiments, we placed particular emphasis on the development of the experimental stimuli. To avoid bias, we specifically used a case example which is intuitively understandable whilst maintaining some degree of realism. We used the findings from the pilot test to improve the experimental material and the pre-experimental briefing. Participants were trained not only in the case example and in the intended review procedure, but also in safety engineering and in the syntax and semantics of Hazard Relation Diagrams, activity diagrams, and Functional Hazard Analysis. The same training material was used in both experiments. Moreover, through industry cooperation and pilot testing, we iteratively improved the experimental material, metrics, and questionnaires until they fit for experimentation in both conditions.

In addition, the mode of testing may have impaired the experimental results. While in Experiment 2, participation took place during a class meeting, scheduling conflicts prevented controlled participation conditions in Experiment 1. To combat this issue, we took great care in identifying irregular responses, by rigorously excluding incomplete participant datasets, data sets with pattern answers, or data sets showing skewed response times, etc. This resulted in the exclusion of several data sets as outlined in Section 10.9.

Conclusion Validity

Confirmation bias and low statistical power may impair conclusions drawn from results. To avoid confirmation bias, we have only accepted hypotheses based on a strict significance level of 0.05. We have taken a conservative approach in discussing results and aimed to illustrate tendencies that are in favor of and against Hazard Relation Diagrams. We have conducted post-

hoc power analyses on all T-Test results and have reported on effect sizes and power in order to increase credibility in analysis results. Due to the possible impact on internal and construct validity of Experiment 1 (see above), we furthermore conducted Experiment 2, 3, and 4 in order to minimize the impact on the “take-home” nature of Experiment 1 on the experimental results and in order to investigate the impact on reviews rigorously in thorough detail. We furthermore designed two empirical experiments investigating the impact on validation at large by means of two different validation techniques (i.e. reviews and inspections) and have conservatively analyzed the impact of Hazard Relation Diagrams on both techniques and have carefully drawn conclusions for validation at large. Findings from Experiment 2 in part confirm findings from Experiment 1, thereby increasing confidence in experimental results. In general, confirmatory, but also contrasting results were highlighted, thereby increasing confidence in the findings.

In addition, the results in both experiments may have been impacted by the participants’ levels of experience. We have reported on a detailed comparison of the differences in experience (Section 10.10) and have discussed the possible impact of experience on the findings (Chapter 12). We have found little differences between the experimental populations, which presumably had little impact on the results.

Considering the conservative approach and strict criteria in accepting evidence in favor of Hazard Relation Diagrams, and considering rigorous experimental design and discussion of findings, we have confidence in our results. Nevertheless, the results from both experiments warrant further investigation into the impact of Hazard Relation Diagrams, particularly with regard to effectiveness, subjective confidence, and more experienced participants (e.g., industry practitioners).

External Validity

Another relevant concern is how the results generalize to circumstances beyond the scope of the study at hand, i.e. if the effects found in this study hold for reviews of any hazards in any project. To ensure external validity, we used an industrial case example developed by industry partners as the basis for the experimental material. The experimental material was rigorously reviewed and quality assured. In addition, already during the development of Hazard Relation Diagrams, industrial practice of validating requirements by means of reviews and inspections was discussed intensively with industrial partners. The experimental procedures were hence developed to reflect industrial practice. Yet, since the industrial case example was reduced in complexity to fit this study’s scope, generalizability with real-world examples may have been lost. It must hence be noted that the experimental material reflects a realistic example, not a real

example. This means that the example used in the empirical evaluation is realistic in the sense that it meets industrial challenges and reflects industrial practice, however is not a real example taken directly from industrial practice.

Student Participant Representativity

The typical arguments regarding the employment of students in experimental evaluations can be brought forward [Carver et al. 2003], i.e. mainly that students may not be representative for industry experts and that the low experience level for undergraduate students from Experiment 1 and 3 may have impacted the results. Since the experimental design in all experiments balanced participants across groups and conditions to similar experience levels (see Section 10.10) and did not draw upon experience (with the exception of the material covered during pre-experimental training), we believe that the experimental designs in general alleviates this issue to some degree. Nevertheless, a repetition of these experiments with practitioners is desirable.

Researcher Bias

The qualitative analysis regarding the rationale categorization gives rise to the possibility of researcher bias during coding of the rationale statements. This is a serious issue that we took great care to avoid. For this purpose, the qualitative analysis was conducted as objectively as possible: We have removed group membership information from the data set while coding rationale statements in order to remove any possibility to influence categorization and categorization was verified by a total of three researchers from two different institutions. This should reduce the influence of researcher bias on categorization. In addition, categorization was verified by three independent researchers. Yet, due to the nature of the quality assurance process of the qualitative data categorizations (see Section 10.9), we were unable to compute interrater reliability, which is an acknowledgeable threat to validity. Furthermore, the fact that one of the experimenters also took on the role of moderator and reader in the inspection sessions, might have introduced some bias during execution of Experiments 2 and 4. Again, we took special care to avoid bias by asking an individual with no interest in the outcome of the study to act as the scribe as well as a control instance against bias during the inspection sessions. Because of this, we have taken a very conservative approach to data preparation and analysis, adhered to a strict protocol, which was reviewed by academic partners. Potentially biased results in both experiments have been rigorously reviewed, rectified, or discarded, where necessary. We are therefore confident that we took sufficient measures to minimize bias.

Part IV:

Conclusion and Outlook

14. Contribution, Limitations, and Future Work

In this chapter, we discuss the work undertaken in this dissertation. Section 14.1 summarizes the contributions this work presents. Section 14.2 discusses the limitations of the solution approach. Finally, Section 14.3 gives an outlook onto future work.

14.1. Contribution of this Dissertation

In Section 1.4, we have outlined the goal of this dissertation to *support stakeholders in the validation of hazard-mitigating requirements through an integrated diagrammatic representation which visualizes the dependencies between hazard-mitigating requirements, hazards, the hazard's trigger conditions, and the hazard's safety goal*. Throughout this dissertation, we have discussed Hazard Relation Diagrams as a fulfillment of this goal. In order to realize Hazard Relation Diagrams as a solution approach to the goal of this dissertation from Section 1.4, we have proposed a number solution components in Section 1.5 These are summarized in the following.

Modeling Concepts of Hazard Relation Diagrams

In this dissertation, we have proposed the modeling concepts of Hazard Relation Diagrams (Chapter 5), which visualize the dependencies between hazard-mitigating requirements and hazard analysis results (Section 5.1). These modeling concepts have been ontologically integrated with UML activity diagrams (Section 5.2). Moreover, a visual notation in accordance with [Moody 2009] has been defined (Section 5.3). We presented a detailed discussion of non-trivial n:m relationships between hazards, conceptual mitigations, and hazard-mitigating requirements distributed across several UML activity diagrams. The discussion showed how Hazard Relation Diagrams can be used to visualize such relationships (see Section 5.4). Based on the ontological foundations, well-formedness rules were defined to support the creation of Hazard Relation Diagrams (Section 5.5).

Formal Approach to Create Hazard Relation Diagrams

We presented an approach to create Hazard Relation Diagrams in Chapter 6. We proposed formalizations of Hazard Relation Diagrams, functional requirements, hazard analysis worksheets, and mitigation templates (Section 6.2). We presented OMG Query/View/Transformation scripts based on the formalized artifacts, which create Hazard Relation Diagrams in two steps: first, UML activity diagrams containing hazard-mitigating requirements are created based on

activity diagrams containing functional requirements and based on transformation steps specified in mitigation templates. Second, Hazard Relation Diagrams are created by appending hazard analysis results to the activity diagrams containing hazard-mitigating requirements from the previous step (Section 6.3). We have illustrated how the well-formedness rules for Hazard Relation Diagram are enforced by the scripts (Section 6.4).

Tool Support for Hazard Relation Diagrams

This dissertation furthermore demonstrated and discussed tool support for Hazard Relation Diagrams (Chapter 7). Tool support for Hazard Relation Diagrams consists of two components: a profile for the UML modeling tool Enterprise Architect (Section 7.2) and a tool prototype for Eclipse (Section 7.3). Tool support allows modeling and creating Hazard Relation Diagrams.

Empirical Evaluation of Hazard Relation Diagrams

We have presented the design and results of a detailed empirical evaluation of Hazard Relation Diagrams (Part III). The purpose of the empirical evaluation was to assess the impact of Hazard Relation Diagrams on the challenges from Section 1.2. Three research questions have been defined in accordance with these challenges. Four hypotheses have been defined to investigate the research questions. A total of four experiments using reviews and inspections as well as between-subjects as well as within-subjects designs were conducted to investigate all hypotheses. The results of the empirical investigation concerning the impact of Hazard Relation Diagrams on effectiveness were inconclusive (Hypothesis H3). However, the experiments show that Hazard Relation Diagrams significantly improve the rational objectivity when validating the adequacy of hazard-mitigating requirements (Hypothesis H1). Empirical results also show that using Hazard Relation Diagrams during validation leads to improved efficiency (Hypothesis H3). Furthermore, there are indications that using Hazard Relation Diagrams improve validators' confidence in their judgements (Hypothesis H4).

14.2. Discussion and Limitations

In this section, the contribution of this dissertation are critically evaluated.

Manual Specification of Hazard-Mitigating Requirements in Mitigation Templates

Albeit Hazard Relation Diagrams can be automatically created, the creation process requires manual specification of hazard-inducing requirements using mitigation templates. From a usability perspective, this manual specification might be tedious for developers and they might

prefer the manual creation of Hazard Relation Diagrams. Manual creation of Hazard Relation Diagrams still yields the advantages outlined above. However, additional effort to verify that the resulting Hazard Relation Diagrams enforce the well-formedness rules is required.

Sequential Validation of Hazards' Conceptual Mitigations

The use of Hazard Relation Diagrams might be constraint by the assumption that one hazard is validated at the time. This kind of validation might be quite time consuming [Flynn and Warhurst 1994], especially when there are large amounts of hazard-mitigating requirements and large amounts of hazards. However, safety standards (e.g., [ARP4761 1996] or [ISO26262 2011]) require that each conceptual mitigation has to be validated with regard to each hazard it is meant to mitigate. Thus, every hazard must be meticulously identified, documented, and mitigated, and there must be no doubt about the adequacy of the conceptual hazard mitigations.

However, the question remains if conceptual mitigation validation on a per-hazard basis is more or less efficient than bulk assessments of hazards. We did not validate this. Empirical evidence if individual hazards or bulk hazard validation using Hazard Relation Diagrams is more effective is an open issue.

Limitations of the Tool Prototype

The intention of the tool support was to show that Hazard Relation Diagrams can be created and visualized. This was illustrated in Chapter 7. Limitations of the tool prototype to create Hazard Relation Diagrams include graphical layouting of the created Hazard Relation Diagrams, execution order of transformation steps, and well-formedness of activity diagrams, as outlined in Chapter 8.

14.3. Future Work

The following related topics have been identified for continued and tangential work.

Continued Empirical Investigation of the Impact of Hazard Relation Diagrams

We have empirically evaluated the benefit of Hazard Relation Diagrams in reviews and inspections using within- and between-subjects designs. Our experiments were conducted with students. Future work might conduct repetition studies of our empirical investigations involving experienced students and/or practitioners. Furthermore, an investigation into which contextual information about a hazard are most beneficial to get objective validation results might be fruit-

ful. Moreover, investigations could focus on the upper limit in modeling elements and associations within a diagram that validators can be expected to be able to handle before the complexity of the visual representation outweighs the benefit of Hazard Relation Diagrams.

Reduction of Subjectivity in Validation and Confidence in Safety Arguments

The quality and objectivity of validation results largely depends on the involved stakeholders, their understanding of the relevant processes (see [Lisagor et al. 2010; Sun 2012]), their understanding of the system under development (see [Gacitua et al. 2009; Glinz and Fricker 2015]), and simply on the information available to them to make an informed judgment (see, e.g., [Flynn and Warhurst 1994; Sikora et al. 2011; Sikora et al. 2012]). Especially in early phases of development, there is thus a risk that those factors affect the confidence in the adequacy of conceptual mitigations, and thereby the overall confidence in the safety. Hazard Relation Diagrams provide an objective visualization of the dependencies between safety assessment artifacts and requirements engineering artifacts. While the positive impact of Hazard Relation Diagrams on validation has been established, future work might be concerned with investigating the impact of Hazard Relation Diagrams in improving the confidence in the safety argument of a system under development and their ability to improve the safety of the system.

Improving the Tool Prototype to Create Hazard Relation Diagrams

Obviously, the limitations of the tool prototype from Section 14.2 shall be the focus of future work.

References

A

- [Allenby and Kelly 2001] K. Allenby, T. Kelly: *Deriving Safety Requirements using Scenarios*. In: Proceedings of the 5th IEEE International Symposium on Requirements Engineering, 2001, pp 228–235.
- [ARP4761 1996] SAE International: *ARP4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment*. 1996.
- [Asnar and Giorgini 2006] Y. Asnar, P. Giorgini: *Modelling Risk and Identifying Countermeasure in Organizations*. In: Proceedings of the 1st International Workshop on Critical Information Infrastructures Security, 2006, pp. 55-66.
- [Aurum et al. 2002] A. Aurum, H. Petersson, C. Wohlin: *State-of-the-Art: Software Inspections after 25 Years*. *Software Testing, Verification, and Reliability*, 12(3), 2002, pp. 133–154.
- [Awodey 2011] S. Awodey: *From Sets to Types, to Categories, to Sets*. In: *Foundational Theories of Classical and Constructive Mathematics*, Springer, Heidelberg, 2011, pp. 113-125.

B

- [Bashir and Qadir 2006] M. Bashir, M. Qadir: *Traceability Techniques: A Critical Study*. In: Proceedings of the 2006 IEEE Multitopic Conference, 2006, pp. 265-268.
- [Basili et al. 1996] V. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Sørumgård, M. Zelkowitz: *The Empirical Investigation of Perspective-Based Reading*. *Empirical Software Engineering* 1(2), 1996, pp. 133–164.
- [Basir et al. 2010] N. Basir, E. Denney, B. Fischer: *Deriving Safety Cases for Hierarchical Structure in Model-Based Development*. In: Proceedings of the 29th International Conference on Computer Safety, Reliability, and Security, 2010, pp. 68-81.
- [Beckers et al. 2013] K. Beckers, M. Heisel, T. Frese, D. Hatebur: *A Structured and Model-Based Hazard Analysis and Risk Assessment Method for Automotive Systems*. In: Proceedings of the 24th International Symposium on Software Reliability, Engineering, 2013, pp. 238-247.
- [Belli et al. 2007] F. Belli, A. Hollmann, N. Nissanke: *Modeling, Analysis and Testing of Safety Issues – An Event-Based Approach and Case Study*. In: Proceedings of the 26th International Conference Computer Safety, Reliability and Security, 2007, pp. 276–282.
- [Berry 1998] D. Berry: *The Safety Requirements Engineering Dilemma*. In: Proceedings of the 9th International Workshop on Software Specification and Design, 1998, pp. 147-149.
- [Bharadwaj and Heitmeyer 1999] R. Bharadwaj, C. Heitmeyer: *Model Checking Complete Requirements Specifications Using Abstraction*. *Automated Software Engineering* 6(1), 1999, pp. 37–68.

- [Biehl et al. 2010] M. Biehl, Chen DeJiu, M. Törngren: *Integrating Safety Analysis into the Model-based Development Toolchain of Automotive Embedded Systems*. In: Proceedings of the ACM SIGLAN/SIGBED 2010 Conference on Languages, Compilers, and Tools for Embedded Systems, 2010, pp. 125-132.
- [Bishop et al. 2004] P. Bishop, R. Bloomfield, S. Guerra: *The Future of Goal-Based Assurance Cases*. In: Proceedings of the Workshop on Assurance Cases. Supplemental Volume of the 2004 International Conference on Dependable Systems and Networks, 2004, pp. 390–395.
- [Bitsch 2001] F. Bitsch: *Safety Patterns — The Key to Formal Specification of Safety Requirements*. In: Proceedings of the 20th International Conference on Computer Safety, Reliability and Security, 2001, pp. 176–189.
- [Bogenstahl and Junge 2016] C. Bogenstahl, R. Junge: *Datengrab*. Grafit Verlag, 2016.
- [Boehm 1981] B. Boehm: *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.
- [Boehm and Basili 2001] B. Boehm, V. Basili: *Software Defect Reduction Top-10 List*. IEEE Computer 34(1), 2001, pp. 135–137.
- [Bohm 1984] B. Boehm: *Verifying and Validating Software Requirements and Design Specifications*. IEEE Software, January 1984, pp. 75-88.
- [Bresciani et al. 2001] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, A. Perini: *Modeling Early Requirements in Tropos: A Transformation based Approach*. In: Proceedings of the 2nd International Workshop on Agent-Oriented Software Engineering, 2001, pp. 151-168.

C

- [Cancila et al. 2009] D. Cancila, F. Terrier, F. Belmonte, H. Dubois, H. Espinoza S. Gérard, A. Cuccuru: *SO-PHIA: A Modeling Language for Model-Based Safety Engineering*. In: Proceedings of the 2nd International Workshop on Model Based Architecting and Construction of Embedded Systems, 2009, pp. 11-26.
- [Caramihai and Dumitrache 2013] S. Caramihai, I. Dumitrache: *Urban Traffic Monitoring and Control as a Cyber-Physical System Approach*. In: Advances in Intelligent Control Systems and Computer Science. Springer Heidelberg, Germany, 2013, pp. 355–366.
- [Carver et al. 2003] J. Carver, L. Jaccheri, S. Morasca, F. Shull: *Issues in Using Students in Empirical Studies in Software Engineering Education*. In: Proceedings of the 9th International Software Metrics Symposium, 2003, pp. 239–249.
- [Carver et al. 2008] J. Carver, N. Nagappan, A. Page: *The Impact of Educational Background on the Effectiveness of Requirements Inspections: An Empirical Study*. IEEE Transactions on Software Engineering 34(6), 2008, pp. 800–812.
- [Chen et al. 2008] DeJiu Chen, R. Johansson, H. Lönn, Y. Papadopoulos, A. Sandberg, F. Törner, M. Törngren: *Modelling Support for Design of Safety-Critical Automotive Embedded Systems*. In: Proceedings of the 27th International Conference on Computer Safety, Reliability, and Security, 2008, pp. 72-85.
- [Chen and Motet 2009a] Z. Chen, G. Motet: *Modeling System Safety Requirements using Input/Output Constraint Meta-Automata*. In: Proceedings of the 4th International Conference on Systems, 2009, pp. 228-233.

-
- [Chen and Motet 2009b] Z. Chen, G. Motet: *System Safety Requirements as Control Structures*. In: Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference, 2009, pp. 324-331.
- [Chen and Motet 2009c] Z. Chen, G. Mote: *Formalizing Safety Requirements using Controlling Automata*. In: Proceedings of the 2nd International Conference on Dependability, 2009, pp. 81-86.
- [Cheung and Kramer 1999] S. Cheung, J. Kramer: *Checking Safety Properties using Compositional Reachability Analysis*. ACM Transactions on Software Engineering and Methodology 8, 1999, pp. 49-78.
- [Cleland-Huang et al. 2002a] J. Cleland-Huang, C. Chang, G. Sathi, K. Javvaji, H. Hu, J. Xia: *Automating Speculative Queries Through Event-based Requirements Traceability*. In: Proceedings of the 8th IEEE Joint International Conference on Requirements Engineering, 2002, pp. 289-296.
- [Cleland-Huang et al. 2002b] J. Cleland-Huang, C. Chang, Y. Ge: *Supporting Event Based Traceability Through High-Level Recognition of Change Events*. In: Proceedings of the 26th Annual International Computer Software and Applications Conference, 2002, pp. 595-600.
- [Cleland-Huang et al. 2003] J. Cleland-Huang, C. Chang: *Event-Based Traceability for Managing Evolutionary Change*. IEEE Transactions on Software Engineering 29(9), 2003, pp. 796-810.
- [Cleland-Huang 2005] J. Cleland-Huang: *Toward Improved Traceability of Non-Functional Requirements*. In Proceedings of the 3rd International Workshop on Traceability in Emerging Forms of Software Engineering, 2005, pp. 14-19.
- [Cleland-Huang et al. 2005] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezanskaya, S. Christina: *Goal-Centric Traceability for Managing Non-Functional Requirements*. In: Proceedings of the 27th International Conference on Software Engineering, 2005, p. 362-371.
- [Cleland-Huang et al. 2012] J. Cleland-Huang, M. Heimdahl, J. Huffman Hayes, R. Lutz, P. Maeder: *Trace Queries for Safety Requirements in High Assurance Systems*. In Proceedings of the 18th International Working Conference on Requirements Engineering: Foundation for Software Quality, 2012, pp. 179-193.
- [Correa et al. 2010] T. Correa, L. Becker, J.-M. Farines, J.-P. Bodeveix, M. Filali, F. Vernadat: *Supporting the Design of Safety Critical Systems using AADL*. In: Proceedings of the 15th IEEE International conference on Engineering of Complex Computer Systems, 2010, pp. 331-336.
- [Cohen 1992] J. Cohen: *A Power Primer*. Psychological Bulletin 112(1), 1992, pp. 155-159.
- [Conway 1967] M. Conway: *How Do Committees Invent?* Datamation 14(4), 1967, pp. 28-31.
- [Corbin and Strauss 2008] J. Corbin, A. Strauss: *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 3rd Edition. Sage Publications, Los Angeles, 2008.
- [Cronbach 1951] L. Cronbach: *Coefficient Alpha and the Internal Structure of Tests*. Psychometrika 16(3), 1951, pp. 297-334.
- [Cziharz et al. 2016] T. Cziharz, P. Hruschka, S. Queins, T. Weyer: *Handbook of Requirements Modeling According to the IREB Standard*. International Requirements Engineering Board CPRE Advanced Level Handbook, Version 1.3. Available at: <https://goo.gl/8ScgPB>, accessed October 1, 2016.
-

D

- [Darimont and van Lamsweerde 1996] R. Darimont, A. van Lamsweerde: *Formal Refinement Patterns for Goal-Driven Requirements Elaboration*. In: Proceedings of the 4th ACM International Symposium on the Foundations of Software Engineering, 1996, pp. 179-190.
- [Denger 2009] C. Denger: *SafeSpection – A Framework for Systematization and Customization of Software Hazard Identification by Applying Inspection Concepts*. Dissertation, University of Kaiserslautern, Germany, 2009.
- [Denger et al. 2008] C. Denger, M. Trapp, P. Liggesmeyer: *SafeSpection – A Systematic Customization Approach for Software Hazard Identification*. In: Proceedings of the 27th International Conference on Computer Safety, Reliability, and Security, 2008, pp. 44-57.
- [Dittel and Aryus 2010] T. Dittel, H. Aryus: *How to “Survive” a Safety Case According to ISO 26262*. In: Proceedings of the 29th International Conference on Computer Safety, Reliability and Security, 2010, pp 97–111.

E

- [Eclipse Modeling Tools] Eclipse Modeling Tools, Luna Package Distribution. Available at <https://goo.gl/qo9Sf5>, accessed July 13, 2016.
- [Eclipse Papyrus] Eclipse UML2 Tools, Luna Package Distribution. Available at <https://goo.gl/6EfDUi>, accessed July 13, 2016.
- [Eclipse QVT v3.5.0] QVT Operational Eclipse Plugin, v3.5.0. Available at <https://goo.gl/SglK1F>, accessed July 13, 2016.
- [Eclipse UML2 Tools] Eclipse UML2 Tools, Luna Package Distribution. Available at <https://goo.gl/fXLxfe>, accessed July 13, 2016.
- [Enterprise Architect] SparxSystems Enterprise Architect, Version 11. Available at <https://goo.gl/V7z4Ms>, accessed July 13, 2016.
- [Ericson 2005] C. Ericson III: *Hazard Analysis Techniques for System Safety*. John Wiley & Sons, Inc, Hoboken, NJ, USA, 2005.
- [Eshuis and Wieringa 2001] R. Eshuis, R. Wieringa: *A Formal Semantics for UML Activity Diagrams – Formalizing Workflow Models*. Technical Report, University of Twente, 2001.

F

- [Fagan 1976] M. Fagan: *Design and Code Inspections to Reduce Errors in Program Development*. IBM Systems Journal 15(3), 1976, pp. 182–211.
- [Fagan 1986] M. Fagan: *Advances in Software Inspections*. IEEE Transactions on Software Engineering SE-12(7), 1986, pp. 744–751.

-
- [Fenelon and McDermid 1993] P. Fenelon, J. McDermid: *An Integrated Tool Set for Software Safety Analysis*. Journal of Systems and Software 21(3), 1993, pp. 279–290.
- [Finkelstein et al. 1992] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, M. Goedicke: *Viewpoints: A Framework for Integrating Multiple Perspectives in System Development*. International Journal of Software Engineering and Knowledge Engineering 2(1), 1992, pp. 31–58.
- [Firesmith 2004] D. Firesmith: *Engineering Safety Requirements, Safety Constraints, and Safety-Critical Requirements*. Journal of Object Technology 3(3), 2004, pp. 27–42.
- [Fisher 1918] R. Fisher: *The Correlation Between Relatives on the Supposition of Mendelian Inheritance*. Philosophical Transactions of the Royal Society of Edinburgh 52, 1918, pp. 399–433.
- [Flynn and Warhurst 1994] D. Flynn, R. Warhurst: *An Empirical Study of the Validation Process Within Requirements Determination*. Information Systems Journal 4(3), 1994, pp. 185–212.
- [Fowler and Parsons 2010] M. Fowler, R. Parsons: *Domain Specific Languages*. Addison-Wesley, Reading, MA, USA, 2010.
- [Fuentes-Fernandés and Vallecillo-Moreno 2004] L. Fuentes-Fernandés, A. Vallecillo-Moreno: *An Introduction to UML Profiles*. Upgrade 5(2), 2004, pp. 6–16.
- ## G
- [Gacitua et al. 2009] R. Gacitua, L. Ma, B. Nuseibeh, P. Piwek, A. de Roeck, M. Rouncefield, P. Sawyer, A. Willis, H. Yang: *Making Tacit Requirements Explicit*. In: Proceedings of the 2nd International Workshop on Managing Requirements Knowledge, 2009, pp. 40–44.
- [Giorgini et al. 2005] P. Giorgini, J. Mylopoulos, R. Sebastiani: *Goal-oriented Requirements Analysis and Reasoning in the Tropos Methodology*. Engineering Applications of Artificial Intelligence 18(2), 2005, pp. 159–171.
- [Glinz 2000] M. Glinz: *Improving the Quality of Requirements with Scenarios*. In: Proceedings of the 2nd World Congress on Software Quality, 2000, pp. 55–60.
- [Glinz 2007] M. Glinz: *On Non-Functional Requirements*. In: Proceedings of the 15th IEEE International Conference on Requirements Engineering, 2007, pp. 21–26.
- [Glinz and Fricker 2015] M. Glinz, S. Fricker: *On Shared Understanding in Software Engineering: An Essay*. Computer Science Research and Development 30(3-4), 2015, pp. 363–376.
- [Glinz and Fricker 2015] M. Glinz, S. Fricker: *On Shared Understanding in Software Engineering: An Essay*. Computer Science Research & Development 30(3-4), 2015, pp. 363–376.
- [Goodhue 1998] D. Goodhue: *Development and Measurement Validity of a Task-Technology Fit Instrument for User Evaluations of Information System*. Decision Sciences 29(1), 1998, pp. 105–138.
- [Gotel and Finkelstein 1994] O. Gotel, A. Finkelstein: *An Analysis of the Requirements Traceability Problem*. In: Proceedings of the 1st International Requirements Engineering Conference, 1994, pp. 94–101.
-

H

- [Hansen et al. 1998] K. Hansen, A. Ravn, V. Stavridou: *From Safety Analysis to Software Requirements*. IEEE Transactions on Software Engineering 24(7), 1998, pp. 573–584.
- [Hart et al. 2000] C. Hart, P. Mulhall, A. Berry, J. Loughran, R. Gunstone: *What is the Purpose of this Experiment? Or Can Students Learn Something from Doing Experiments?* Journal of Research in Science Teaching 37(7), 2000, pp. 655–675.
- [Hatcliff et al. 2014] J. Hatcliff, A. Wassyng, T. Kelly, C. Comar, P. Jones: *Certifiably Safe Software-Dependent Systems: Challenges and Directions*. In: Proceedings on the Future Software Engineering, 2014, pp. 182–200.
- [Hawkins et al. 2011] R. Hawkins, T. Kelly, J. Knight, P. Graydon: *A New Approach to Creating Clear Safety Arguments*. In: Advances in Systems Safety. Springer London, UK, 2011, pp 3–23.
- [Heim and Kratzer 1998] I. Heim, A. Kratzer: *Semantics in Generative Grammar*. Wiley, Chichester, 1998.
- [Heimdahl 2007] M. Heimdahl: *Safety and Software Intensive Systems: Challenges Old and New*. In: Proceedings on the Future of Software Engineering, 2007, pp 137–152.
- [Heitmeyer et al. 1998] C. Heitmeyer, J. Kirby, B. Labaw, M. Archer, R. Bharadwaj: *Using Abstraction and Model Checking to Detect Safety Violations in Requirements Specifications*. IEEE Transactions on Software Engineering 24(11), 1998, pp. 927–948.
- [Hill and Tilley 2010] J. Hill, S. Tilley: *Creating Safety Requirements Traceability for Assuring and Recertifying Legacy Safety-Critical Systems*. In: Proceedings of the 18th IEEE International Requirements Engineering Conference, 2010, pp. 297-302.

I

- [IBM SPSS 22] IBM: SPSS Statistics, Version 22.0. Available at: <https://goo.gl/esFvP4>, accessed July 13, 2016.
- [IREB e.V. 2016] International Requirements Engineering Board: IREB Glossary, version 1.6. Available at: <https://goo.gl/NOh7NX>, accessed November 29, 2016.
- [ISO26262 2011] International Organization for Standardization: *ISO26262: Road Vehicles – Functional Safety*. 2011.

J

- [Jedlitschka et al. 2008] A. Jedlitschka, M. Ciolkowski, D. Pfahl: *Reporting Experiments in Software Engineering*. In: Guide to Advanced Empirical Software Engineering. Springer London, 2008, pp 201–228.
- [Johnson and Holloway 2006] C. Johnson, C. Holloway: *Questioning the Role of Requirements Engineering in the Causes of Safety-Critical Software Failures*. In: Proceedings of the IET International Conference on System Safety, 2006, pp. 352–361.

[Joshi and Heimdahl 2005a] A. Joshi, M. Heimdahl: *Model-Based Safety Analysis of Simulink Models Using SCADE Design Verifier*. In: Proceedings of the 24th International Conference on Computer Safety, Reliability, and Security, 2005, pp. 122–135.

[Joshi and Heimdahl 2005b] A. Joshi, M. Heimdahl: *A Proposal for Model-Based Safety Analysis*. In: Proceedings of the 24th Digital Avionics Systems Conference, 2005.

K

[Kaiser et al. 2010] B. Kaiser, V. Klaas, S. Schulz, C. Herbst, P. Lascych: *Integrating System Modelling with Safety Activities*. In: Proceedings of the 29th International Conference on Computer Safety, Reliability, and Security, 2010, pp. 452-465.

[Katta and Stålhane 2011] V. Katta, T. Stålhane: *A Conceptual Model of Traceability for Safety Systems*. In: Proceedings of the 2nd International Conference on Complex Systems Design and Management, 2011.

[Kelly 2007] T. Kelly: *Reviewing Assurance Arguments: A Step-by-Step Approach*. In: Proceedings of the Workshop on Assurance Cases for Security – The Metrics Challenge, 2007.

[Kelly and Tolvanen 2008] S. Kelly, J.-P. Tolvanen: *Domain-Specific Modeling – Enabling Full Code Generation*. John Wiley & Sons, Inc, New York, NY, USA, 2008

[Kelly and Weaver 2004] T. Kelly, R. Weaver: *The Goal Structuring Notation – A Safety Argument Notation*. In: Proceedings of the Workshop on Assurance Cases of Dependable Systems and Networks, 2004.

[Knight 2002] J. Knight: *Software Challenges in Aviation Systems*. In: Proceedings of the 21st International Conference on Computer Safety, Reliability, and Security, 2002, pp. 106-112.

[Kotonya and Sommerville 1997] G. Kotonya, I. Sommerville: *Integrating Safety Analysis and Requirements Engineering*. In: Proc. of the Joint 4th International Computer Science Conference and the 4th Asia-Pacific Software Engineering Conference, 1997, pp. 259-271.

[Kotonya and Sommerville 1998] G. Kotonya, I. Sommerville: *Requirements Engineering: Processes and Techniques*. John Wiley & Sons, Inc, New York, NY, USA, 1998.

L

[Lagarde et al. 2007] F. Lagarde, H. Espinoza, F. Terrier, S. Gérard: *Improving UML Profile Design Practices by Leveraging Conceptual Domain Models*. In: Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering, 2007, pp. 445-448.

[Lagarde et al. 2008] F. Lagarde, H. Espinoza, F. Terrier, C. André, S. Gérard: *Leveraging Patterns on Domain Models to Improve UML Profile Definition*. In: Proceedings of 11th International Conference on Fundamental Approaches to Software Engineering, 2008, pp. 116-130.

[Lee and Boehm 2005] K. Lee, B. Boehm: *Empirical Results from an Experiment on Value-based Review (VBR) Processes*. In: Proceedings of the International Symposium on Empirical Software Engineering, 2005, pp. 3-12.

References

- [Lehman et al. 2016] E. Lehman, F. Leighton, A. Meyer: *Mathematics for Computer Science*. Available at: <https://goo.gl/hWhICO>, accessed February 19, 2017.
- [Letier and van Lamsweerde 2002] E. Letier, A. van Lamsweerde: *Deriving Operational Software Specifications from System Goals*. In: Proceedings of the 10th ACM International Symposium on the Foundations of Software Engineering, 2002, pp. 119-128.
- [Letier and van Lamsweerde 2004] E. Letier, A. van Lamsweerde: *Reasoning About Partial Goal Satisfaction for Requirements and Design Engineering*. In: Proceedings of the 12th ACM International Symposium on the Foundations of Software Engineering, 2004, pp. 53-62.
- [Leveson 1986] N. Leveson: *Software Safety: Why, What, and How*. ACM Computing Surveys, 18(2), 1986, pp. 125-163.
- [Leveson 1991] N. Leveson: *Software Safety in Embedded Computer Systems*. Communications of the ACM, 34(2), 1991, pp. 34-46.
- [Leveson 1995] N. Leveson: *Safeware: System Safety and Computers*. Addison-Wesley, Reading, MA, USA, 1995.
- [Leveson 2004] N. Leveson: *A Systems-Theoretic Approach to Safety in Software-Intensive Systems*. IEEE Transactions on Dependable and Secure Computing 1(1), 2004, pp. 66–86.
- [Leveson 2011a] N. Leveson: *Engineering a Safer World: Systems Thinking Applied to Safety*. MIT Press, Cambridge, MA, USA, 2011.
- [Leveson 2011b] N. Leveson: *The Use of Safety Cases in Certification and Regulation*. Journal of System Safety 47(6), 2011. Available at: <http://goo.gl/j9NW5Y>, accessed July 13, 2016.
- [Li et al. 2011] Q. Li, B. Boehm, Y. Yang, Q. Wang: *A Value-based Review Process for Prioritizing Artifacts*. In: Proceedings of the International Conference on Software Systems Process, 2011, pp 13–23.
- [Liggesmeyer 2002] P. Liggesmeyer: *Software-Qualität: Testen, Analysieren und Verifizieren von Software*. Spektrum, Heidelberg, 2002.
- [Lisagor et al. 2010] O. Lisagor, L. Sun, T. Kelly: *The Illusion of Method: Challenges of Model-Based Safety Assessment*. In: Proceedings of the 28th International System Safety Conference, 2010.
- [Lisagor et al. 2011] O. Lisagor, T. Kelly, R. Niu: *Model-Based Safety Assessment – Review of the Discipline and its Challenges*. In: Proceedings of the 9th International Conference on Reliability, Maintainability, and Safety, 2011.
- [Lutz 2000] R. Lutz: *Software Engineering for Safety: A Roadmap*. In: Proceedings of the Future of Software Engineering, 2000, pp. 215-224.

M-N

- [Mao and Chen 2012] J. Mao, L. Chen: *Runtime Monitoring for Cyber-physical Systems: A Case Study of Cooperative Adaptive Cruise Control*. In: Proceedings of the 2nd International Conference on Intelligent Systems Design and Engineering Applications, 2012, pp. 509–515.
- [Menon and Kelly 2010] C. Menon, T. Kelly: *Eliciting Software Safety Requirements in Complex Systems*. In: Proceedings of the 4th Annual IEEE Systems Conference, 2010, pp. 616-621.
- [Moody 2009] D. Moody: *The “Physics” of Notation: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering*. IEEE Transactions on Software Engineering 35(6), 2009, pp 756-779.
- [Mouratidis et al. 2003] H. Mouratidis, P. Giorgini, G. Manson: *Integrating Security and Systems Engineering: Towards the Modelling of Secure Information Systems*. In: Proceedings of the 15th International Conference on Advanced Information Systems Engineering, 2003, pp. 63-78.

O

- [OMG 2015a] Object Management Group: *OMG Meta Object Facility (MOF) Core, Version 2.5*. OMG Document Number formal/2015-06-05. Available at <http://goo.gl/phs4kA>, accessed July 13, 2016.
- [OMG 2015b] Object Management Group: *OMG Unified Modeling Language (OMG UML), Version 2.5*. OMG Document Number formal/2015-03-01. Available at <http://goo.gl/7cQyPv>, accessed July 13, 2016.
- [OMG 2016] Object Management Group: *Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, Version 1.3*. OMG Document Number formal/2016-06-03. Available at <http://goo.gl/RGUr44>, accessed July 13, 2016.
- [Osgood et al. 1957] C. Osgood, G. Suci, P. Tannenbaum: *The Measurement of Meaning*. University of Illinois Press, Illinois, 1957.

P

- [Peraldi-Frati and Albinet 2010] M.-A. Peraldi-Farti, A. Albinet: *Requirements Traceability in Safety Critical Systems*. In: Proceedings of the 1st Workshop on Critical Automotive applications: Robustness & Safety, 2010, pp. 11-14.
- [Porter et al. 1995] A. Porter, L. Votta, V. Basili: *Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment*. IEEE Transactions on Software Engineering 21(6), 1995, pp. 563–575.
- [Pumfrey 1999] D. Pumfrey: *The Principled Design of Computer System Safety Analyses*. Dissertation, University of York, 1999.

Q-R

- [Panesar-Walawege et al. 2013] R. Panesar-Walawege, M. Sabetzadeh, L. Briand: *Supporting the Verification of Compliance to Safety Standards via Model-Driven Engineering: Approach, Tool-Support, and Empirical Validation*. *Information and Software Technology* 55, 2013, pp. 836-864.
- [Raspotnig and Opdahl 2013] C. Raspotnig, A. Opdahl: *Comparing Risk Identification Techniques for Safety and Security Requirements*. *Journal of Systems and Software* 86(4), 2013, pp. 1124–1151.
- [Ramesh and Jarke 2001] B. Ramesh, M. Jarke: *Toward Reference Models for Requirements Traceability*. *IEEE Transactions on Software Engineering* 27(1), 2001, pp. 58-93.
- [Reif 2010] K. Reif: *Fahrstabilisierungssysteme und Fahrerassistenzsysteme*. Vieweg+Teubner, Wiesbaden, Germany, 2010.
- [Rochimah et al. 2007] S. Rochimah, W. Wan Kadir, A. Abdullah: An Evaluation of Traceability Approaches to Support Software Evolution. In: *Proceedings of the 2nd International Conference on Software Engineering Advances*, 2007, pp. 19-26.
- [Rumbaugh et al. 2004] J. Rumbaugh, I. Jacobson, G. Booch: *The Unified Modeling Language Reference Manual*, 2nd Ed. Pearson, 2004.

S

- [Saeed et al. 1993] A. Saeed, R. de Lemos, T. Anderson: *Robust Requirements Specifications for Safety-Critical Systems*. In: *Proceedings of the 12th International Conference on Comp. Safety, Reliability and Security*, 1993, pp. 219-229.
- [Sánchez et al. 2011] P. Sánchez, D. Alonso, F. Rosique, B. Álvarez, J. Pastor: *Introducing Safety Requirements Traceability Support in Model-Driven Development of Robotic Applications*. *IEEE Transactions on Computers* 60(8), 2011, pp. 1059-1071.
- [Sandberg et al. 2010] A. Sandberg, DeJiu Chen, H. Lönn, R. Johansson, L. Feng, M. Törngren, S. Torchiario, R. Tavakoli-Kolagari, A. Abele: *Model-based Safety Engineering of Interdependent Functions in Automotive Vehicles using EAST-ADL2*. In: *Proceedings of the 29th International Conference on Computer Safety, Reliability, and Security*, 2010, pp. 332-346.
- [Selic 2007] B. Selic: *A Systematic Approach to Domain-Specific Language Design Using UML*. In: *Proceedings of the 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing*, 2007, pp. 2-9.
- [Shull et al. 2000] F. Shull, I. Rus, V. Basili: *How Perspective-based Reading can Improve Requirements Inspections*. *IEEE Computer* 33(7), 2000, pp. 73–79.
- [Shull et al. 2002] F. Shull, V. Basili, B. Boehm, A. Brown, P. Costa, M. Lindvall, D. Port, I. Russ, R. Tesoriero, M. Zelkowitz: *What We Have Learned About Fighting Defects*. In: *Proceedings of the 8th International Symposium on Software Metrics*, 2002, pp 249–258.

-
- [Sikora et al. 2011] E. Sikora, B. Tenbergen, K. Pohl: *Requirements Engineering for Embedded Systems: An Investigation of Industry Needs*. In: Proceedings of the 17th International Working Conference on Requirements Engineering: Foundations for Software Quality, 2011, pp. 151-165.
- [Sikora et al. 2012] E. Sikora, B. Tenbergen, K. Pohl: *Industry Needs and Research Directions in Requirements Engineering for Embedded Systems*. Requirements Engineering 17(1), 2012, pp. 57–78.
- [Sindre 2007] G. Sindre: *A Look at Misuse Cases for Safety Concerns*. In: Proceedings of the IFIP WG 8.1 Conference, 2007, pp 252–266.
- [Sindre and Opdahl 2001] G. Sindre, A. Opdahl: *Capturing Security Requirements through Misuse Cases*. In: Proceedings of the Proceedings of Norsk Informatikkonferanse, 2001, pp 219-230.
- [Sjøberg et al. 2005] D. Sjøberg, J. Hannay, O. Hansen, V. Kampenes, A. Karahasanovic, N. Liborg, A. Rekdal: *A Survey of Controlled Experiments in Software Engineering*. IEEE Transactions on Software Engineering 31(9), 2005, pp. 733-753.
- [Snelting et al. 2006] G. Snelting, T. Robschink, J. Krinke: *Efficient Path Conditions in Dependence Graphs for Software Safety Analysis*. ACM Transactions on Software Engineering and Methodology 15, 2006, pp. 410-457.
- [SoSci Survey] SoSci Survey Website. Available at: <https://goo.gl/x7FEhO>, accessed July 13, 2016.
- [Spanoudakis et al. 2004] G. Spanoudakis, A. Zisman, E. Pérez-Miñana, P. Krause: *Rule-Based Generation of Requirements Traceability Relations*. Journal of Systems and Software 72, 2004, pp. 105-127..
- [SparxSystems 2014] SparxSystems: *Enterprise Architect User Guide*, 2014. Available at: <https://goo.gl/w2Enek>, accessed July 13, 2016.
- [Sternbach and Okuda 1991] R. Sternbach, M. Okuda: *USS Enterprise: Technical Manual*, 1st Edition, Pocket Books, New York, 1991.
- [Stoneburner 2006] G. Stoneburner: *Toward a Unified Security-Safety Model*. IEEE Computer 39(8), 2006, pp. 96–97.
- [Störrle 2004] H. Störrle: *Semantics of Control-Flow in UML 2.0 Activities*. In: Proceedings of the IEEE Symposium in Visual Languages and Human Centric Computing, 2004, pp 235–242.
- [Student 1908] Student: *The Probable Error of a Mean*. Biometrika 6(1), 1908, pp. 1–25.
- [Sun 2012] L. Sun: *Establishing Confidence in Safety Assessment Evidence*. Dissertation, University of York, 2012.

T

- [Tabachnik and Fidell 2010] B. Tabachnick, L. Fidell: *Using Multivariate Statistics*, 5th Edition. Pearson, Boston, 2010.

- [Tenbergen et al. 2013] B. Tenbergen, P. Bohn, T. Weyer: *A Structured Approach to Derive Method- Specific UML/SysML-Profiles by means of the SPES 2020 Requirements Viewpoint*. In: Proceedings of the 3rd Workshop on the Future of the Development of Software-intensive Embedded Systems, 2013, pp. 235-244.
- [Tenbergen et al. 2014] B. Tenbergen, A. C. Sturm, T. Weyer: *A Hazard Taxonomy for Embedded and Cyber Physical Systems*. In: Proceedings of the 2nd Workshop on Emerging Ideas and Trends in Engineering of Cyber-Physical Systems, 2014.
- [Tenbergen et al. 2015] B. Tenbergen, T. Weyer, K. Pohl: *Supporting the Validation of Adequacy in Requirements-Based Hazard Mitigations*. In: Proceedings of the 21st International Working Conference on Requirements Engineering: Foundation for Software Quality, 2015, pp 17–32.
- [Tenbergen et al. 2017] B. Tenbergen, T. Weyer, K. Pohl: *Hazard Relation Diagrams: A Diagrammatic Representation to Increase Validation Objectivity of Requirements-based Hazard Mitigations*. Requirements Engineering Journal, 2017, DOI: 10.1007/s00766-017-0267-9.
- [Thabane et a. 2010] L. Thabane, J. Ma, J. Cheng, A. Ismaila, L. Rios, R. Robson, M. Thabane, L. Giagregorio, C. Goldsmith: *A Tutorial on Pilot Studies: The What, Why, and How*. BMC Medical Research Methodology 10(1), 2010, DOI: 10.1186/1471-2288-10-1.
- [Torkar et al. 2012] R. Torkar, T. Gorschek, R. Feldt, M. Svahnberg, U. Rajar, K. Kamran: *Requirements Traceability: A Systematic Review and Industry Case Study*. International Journal of Software Engineering and Knowledge Engineering 22(3), 2012, pp. 1-49.
- [Toulmin 1958] S. Toulmin: *The Uses of Argument*. Cambridge University Press, 1958.
- [Troubitsyna 2008] E. Troubitsyna: *Elicitation and Specification of Safety Requirements*. In: Proceedings of the 3rd International Conference on Systems, 2008, pp. 202–207.

U

- [US DOD 2010] K. Cooper, M. DePrenger, S. Mattern, A. McKinley, A. Pajouhesh, D. Shampine: *Joint Software Systems Safety Engineering Handbook*. United States Department of Defense, Version 1.0, 2010. Available at: <http://goo.gl/er2mWY>, accessed July 13, 2016.
- [US FAA 2009a] D. Lempia, S. Miller: *Requirements Engineering Management Findings Report*. Technical Report DOT/FAA/AR-08/34, Federal Aviation Administration, 2009. Available at: <https://goo.gl/dNqndm>, accessed July 13, 2016.
- [US FAA 2009b] D. Lempia, S. Miller: *Requirements Engineering Management Handbook*. Technical Report, DOT/FAA/AR-08/32, Federal Aviation Administration, 2009. Available at: <https://goo.gl/syNIId0>, accessed July 13, 2016.
- [US NASA 2006] A. Joshi, M. Heimdahl, S. Miller, M. Whalen: *Model-Based Safety Analysis*. US National Aeronautics and Space Administration, Document No. NASA/CR-2006-213953, 2006. Available at: <https://goo.gl/Sh6tnK>, accessed July 13, 2016.

[US NASA 2011] H. Dezfuli, A. Benjamin, M. Everett, C. Smith, M. Stamatelatos, R. Youngblood: *NASA System Safety Handbook. Volume 1, System Safety Framework and Concepts for Implementation*. US National Aeronautics and Space Administration, Document No. NASA/SP-2010-580, 2011. Available at: <http://goo.gl/udgv5y>, accessed July 13, 2016.

V

[van Lamsweerde 2009a] A. van Lamsweerde: *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley, Chichester, 2009.

[van Lamsweerde 2009b] A. van Lamsweerde: *Reasoning About Alternative Requirements Options*. In: *Conceptual Modeling: Foundations and Applications*. Springer, Heidelberg, 2009, pp 380–397.

[van Lamsweerde and Letier 1998] A. van Lamsweerde, E. Letier: *Integrating Obstacles in Goal-Driven Requirements Engineering*. In: *Proceedings of the 20th International Conference on Software Engineering*, 1998, pp. 53-62.

[van Solingen and Berhout 1999] R. van Solingen, E. Berghout: *The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development*. McGraw-Hill, London, 1999.

[Venkatesh and Bala 2008] V. Venkatesh, H. Bala: *Technology Acceptance Model 3 and a Research Agenda on Interventions*. *Decision Sciences* 39(2), 2008, pp. 273–315.

[Verhagen et al. 2015] T. Verhagen, B. Hooff, S. van den Meents: *Toward a Better Use of the Semantic Differential in IS Research: An Integrative Framework of Suggested Action*. *Journal of the Association for Information Systems* 16(2), 2015, Article 1.

W

[Wiegers 2002] K. Wiegers: *Peer Reviews in Software: A Practical Guide*. Addison-Wesley, 2002.

[Wilson et al. 1997] S. Wilson, T. Kelly, J. McDermid: *Safety Case Development: Current Practice, Future Prospects*. In: *Proceedings of the 12th Annual CSR Workshop on Safety and Reliability of Software Based Systems*, 1997, pp. 135–156.

[Wohlin et al. 2012] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, A. Weelén: *Experimentation in Software Engineering*. Springer, Heidelberg, 2012.

X-Y-Z

[Yu 1997] E. Yu: *Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering*. In: *Proceedings of the 13th IEEE International Symposium on Requirements Engineering*, 1997, pp. 226-235.

[Zafar and Dromey 2005] S. Zafar, F. Dromey: *Integrating Safety and Security Requirements into Design of an Embedded System*. In: *Proceedings of the 12th Asia-Pacific Software Engineering Conference*, 2005, pp. 629–636.

»... *iunctos non diriment aetas.*«
