# Using Assurance Cases in Requirements Engineering for Safety-Critical Systems

**Camilo Camilo Almendra**[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Recife – PE – Brazil

cca3@cin.ufpe.br

**Program**: Pós-graduação em Ciência da Computação
**Level**: PhD
**Adviser**: Profa. Dra. Flavia Barros (fab@cin.ufpe.br)
**Co-adviser**: Profa. Dra. Carla Silva (ctlls@cin.ufpe.br)
**Entry year in the program**: 2017.1
**Expected date for final presentation**: 2020.2
**Date of approval of the thesis**: Scheduled for August 2019

***Abstract.*** ***Context:*** *Certification of safety-critical systems (SCS) demands thorough documentation that demonstrates why a system shall be considered safe. Assurance Case Development (ACD) is an approach for discussing, analyzing and assessing the safety properties of systems. Software requirements of a SCS are an essential information included in assurance cases, alongside system design and safety analysis information. Lack of integrated analysis of requirements and safety concerns may lead to safety issues in the development of critical systems. One of the challenges for the agile development of SCS is to address both Requirements Engineering (RE) and ACD in an integrated way throughout the development life cycle.* ***Objective:*** *This research proposes a framework to integrate the Assurance Case Development and Agile RE in the development of SCS.* ***Method:*** *This research is organized in three main phases. First, a systematic mapping study is performed to understand how incremental ACD is addressed by current approaches, and a survey with experts is carried out to understand the development of ACs during Agile RE activities in practice. Second, the framework and its supporting tools and documentation will be designed and developed. Finally, a series of empirical studies will evaluate aspects of the framework in a multi-perspective manner and as long as it is developed.* ***Conclusions:*** *We expect that this approach contributes to leverage the development of ACs earlier and integrated with RE activities in agile development of SCS.*

***Keywords:*** *Safety-Critical System. Assurance Case. Agile Requirements Engineering.*
***CBSoft related events:*** *SBES, SBCARS.*

## 1. Problem Characterization

Software is an essential part of modern embedded, electrical/electronic and control systems [Hatcliff et al. 2014]. Software-intensive systems permeate several domains, such as automotive, robotic and medical devices. Safety-critical systems (SCS) are those whose operation brings concerns regarding harms or damages they can cause to people, organizations or environment in case of failure [Hatcliff et al. 2014]. Many standards acknowledge software components as essential parts of modern safety-critical systems and demand the fulfillment of a series of high-level safety requirements and process compliance obligations [Ebert 2015]. In the context of Agile Software Development (ASD) approaches, it is paramount that the production of certification documentation follows the system development [Heeager and Nielsen 2018].

Nowadays, some standards mandate or recommend the use of assurance cases (AC) as part of the certification documentation, such as the standards ISO 26262 (for road vehicles[1]) and Def Stan 00-56 (from UK Minister of Defence and applicable for all defence systems[2]). The purpose of an AC is to present "*a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context*" [Kelly 1999]. ACs are an amalgamation of information that comes from standard requirements, product requirements, risk and hazard analysis results, design decisions and rationale, validation and verification results, and process management records [Bloomfield and Bishop 2010].

Traditional approaches for AC development (ACD) elaborate the ACs after completing construction and verification of a system [Bloomfield and Bishop 2010]. Although ACs require information that will only be completely available at the end of the lifecycle, it is not recommended to start the elaboration of them in the late phases of development [Kelly 1999, Hatcliff et al. 2014, Doss and Kelly 2016]. In fact, according to [Hatcliff et al. 2014], it may raise safety and project management risks, which may lead to increased rework caused by late-discovered safety issues, since a hazard may not be adequately eliminated or mitigated by design. In addition, developers may not have accurately recorded many decisions and safety evidence, maybe because of process deviation or lack of safety planning. Even so, late development of ACs is still a current practice [Myklebust et al. 2017].

Loss of requirements and design rationale affect the quality and completeness of documentation sent to certification authorities [Hatcliff et al. 2014]. Good RE is key to the production of quality systems, even more in the context of software-intensive critical systems [Vilela et al. 2017]. There is extensive research on specification and analysis of requirements for safety-critical systems, but there still are gaps when tracing requirements to safety analysis information [Lutz 2000]. As modern development processes rely more and more on exploratory or iterative-incremental approaches – such as ASD, there is a lack of mechanisms to disseminate knowledge about new or derived safety requirements throughout the software development life cycle (SDLC) [Lutz 2000].

This research explores the development of ACs integrated with Agile RE activities. The objective of this thesis is to propose a framework to support the integrated devel-

[1]https://www.iso.org/standard/43464.html
[2]https://www.skybrary.aero/bookshelf/books/344.pdf

opment of ACs in the context of Agile RE activities. The rest of this paper is structured as follows. Section 2 describes the background concepts of the study. Section 3 presents the expected contributions. Section 4 describes the current status of this research. Section 5 describes preliminary results and ongoing work. Section 6 presents a comparison between our proposal and related works. Finally, we present our conclusions.

## 2. Background

### 2.1. Assurance Case

Initially, AC artifacts consisted of unstructured text comprising hundreds of pages [Bloomfield and Bishop 2010]. The need for a more formal structure has led to the development of structured argument notations to specify assurance cases. The structure of safety arguments comprises three main elements: Claims, Arguments and Evidence [Kelly 1999]. *Claims* are high-level goals or constraints addressed during system development, and the *Arguments* are elaborated to demonstrate that the claims were satisfied. *Evidence* is any piece of information generated during design, implementation and verification activities that substantiates some *Claim* or *Argument*. Arguments are reasoning steps used to show how pieces of evidence support the claims. Often, a top claim needs to be decomposed into sub-claims, thus creating a hierarchy of claims (see Figure 1).

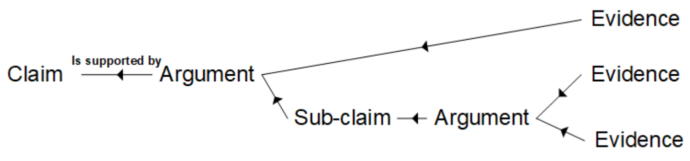

**Figure 1. Safety argument structure (adapted from [Kelly 1999]).**

The provision of evidence to back up claims is the fundamental objective of safety argument development. It requires an explanation, a rationale to explain why the pieces of evidence together confirm a claim. Much of the information needed to develop ACs is stored in project management tools (PMT[3]), specially the tools storing requirements and design decisions.

### 2.2. Requirements Engineering in Safety-Critical Development

As a knowledge-intensive work, requirements elicitation and analysis goes from scattered data in multiple sources (human or not) to structured information and identified decisions, understood and negotiated among stakeholders [Kotonya and Sommerville 1998]. Often we need to take the opposite course, going from decisions and specifications back to their source to confirm understanding. Such explanation interrelating high-level requirements and low-level artifacts is known as the rationale [Burge et al. 2008]. In the context of ACD, rationale are the essence of arguments. It provides the confidence to assure that requirements decomposition and design decisions reasonably address all identified risks and hazards.

[3]We refer to *project management tools* as any information system, tool, model, document or spreadsheet that holds project information regarding requirements, design, safety information, tests, code and traceability.

**Table 1. Expected benefits of ACD in RE-related activities**

| RE-related activity | Purposes for RE | Benefits of ACD |
|---|---|---|
| Release Planning | Decomposition of features into requirements and identification of hazards and high level safety requirements | Preliminary AC reviewing provides a way to early assessment of requirements decomposition |
| Backlog Grooming | Revising of whole product planning and refinement of backlog items for forthcoming iterations | Updated AC generated from PMTs provides a snapshot view of the safety assurance strategy, highlighting points that need further discussion |
| Change Management | Analyze the impact of a change request | This activity requires support to "what-if" analysis, and the AC provide a way to analyze the impact of changes in the safety assurance |
| Iteration Planning | Last discussion towards the scope and acceptance criteria of requirements, and decomposition of backlog items (e.g. stories) into tasks | Updated AC can be used to recall the safety evidence needs |
| Iteration Review | Validation of requirements implementation and quality of deliverables | AC generated at end of iteration will provided stakeholders with an updated status of safety assurance of the system |

Agile RE involves many roles, practices and artifacts [Leffingwell 2010]. We identified five agile development activities that are related to RE and could benefit from ACD. During these activities, the main Agile RE artifacts (e.g. user story and prototypes) are created, updated and discussed [Schön et al. 2017]. The Table 1 presents the purpose of each activity (with respect to RE) and the expected benefits of integrating ACD into it. In a general manner, the process of construction and reviewing of assurance cases provide stakeholders and teams with updated status of the safety assurance. Gaps, inconsistencies and enhancements found in the assurance argumentation trigger improvements to development plans.

## 3. Contributions

The scope of this research refers to the integration of ACD into Agile RE activities, in the context of agile development. We argue that the incorporation of assurance development as single practices alongside the RE activities improve the quality of safety requirements and safety evidence management in SCS development. The objective of this work is to propose a framework that encompasses a model for assurance rationale management and a mechanism for generation of AC from PMTs. The novelty of our approach is to manage assurance rationale directly into PMTs. These tools commonly store requirements, architecture design, tests, code and tracing information. Using the tools enriched with assurance rationale, we will provide a mechanism for AC generation. These cases would be readily available anytime and be used as up-to-date models to support requirements activities. Also, they serve as base to prepare final AC for certification procedures.

The main research questions that guide our work are: *(RQ1) How can assurance information be integrated into agile project management tools?*, and *(RQ2) How to incorporate incremental assurance case development practices into Agile RE processes?*. We seek to answer those research questions by achieving the following specific objectives:

**O1 State of the art on incremental ACD:** We aim to identify and analyze existing approaches for integrating ACD as part of the SDLC in an incremental manner. We aim to characterize the approaches by expected benefits, points of integration with existing processes, roles, activities, and artifacts involved.

**O2 State of the practice on integration between ACD and RE:** We aim to gather practitioners practices and perspectives regarding the development of AC during RE activities.

**O3 Framework design:** A framework will be designed and implemented to provide: i) a model to manage assurance rationale integrated with PMTs; ii) an ontology to guide the assurance rationale information management; iii) a mechanism to generate assurance case from PMTs enriched by assurance rationale information; and iv) a set of guidelines to support the integrated development of AC in the context of RE activities. The initial design of this framework is presented in Section 4.

**O4 Framework evaluation:** We aim to evaluate the overall proposal by a combination of multiple empirical studies. The evaluation plan is discussed in Section 5.

## 3.1. Framework proposal

This work proposes a framework to support the integrated development of ACs in the context of Agile RE activities. There are four main parts that compose the framework, depicted in Figure 2 and discussed as follows.

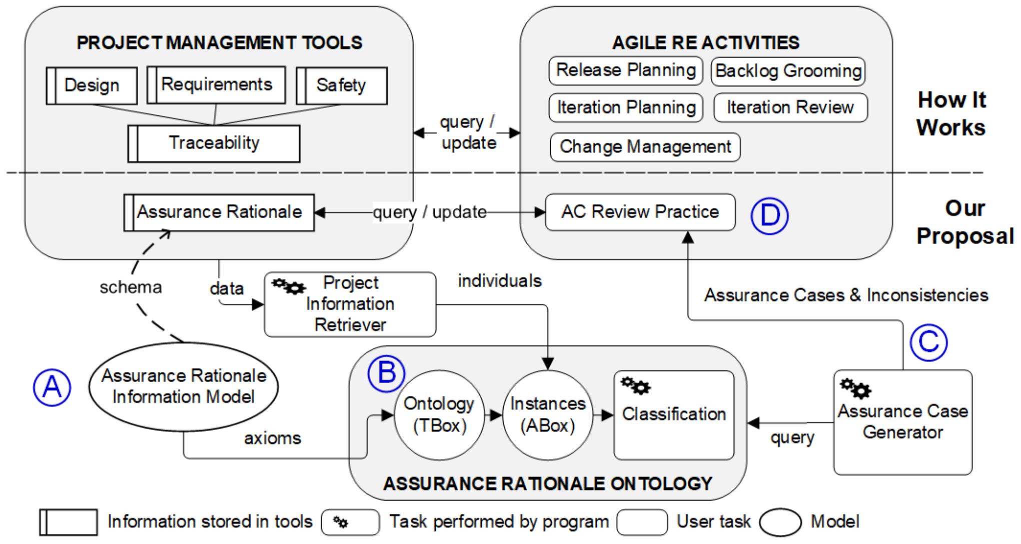

**Figure 2. Overview of thesis proposal.**

- *Assurance rationale information model (ARIM) (Figure 2-A):* We aim to design a model to manage assurance rationale inside PMTs. The ARIM guides the inclusion of safety assurance rationale directly associated with the information items that already exist in the tools and artifacts. The model serves as a schema for implementation of a Project Information Retriever (PIR), that generates input to feed the ontology.
- *Assurance rationale ontology (ARO) (Figure 2-B):* We aim to design an ontology to classify the information retrieved from PMTs. Our ontology will be designed to classify project information regarding requirements, design, tests, code, safety analysis, tracing information, and their relationships. Also, the ontology will classify additional assurance rationale information stored after the adaptation of PMT to incorporate the ARIM concepts. The ARO will be implemented using OWL-DL [4] and Protégé tool [5].

[4]https://www.w3.org/TR/owl-guide/
[5]https://protege.stanford.edu/

- *AC Generator (Figure 2-C):* We aim to implement a mechanism to generate ACs from PMTs enriched by assurance rationale information. We aim to generate ACs in multiple perspectives based on different argumentation patterns.
- *Process guidance to integrated development (Figure 2-D):* We aim to prepare guidelines to support the reviewing of the AC in the context of Agile RE activities, and to support the systematic update of assurance rationale information into PMTs.

## 4. Current Status

The methodology to develop this thesis is described in Figure 3.

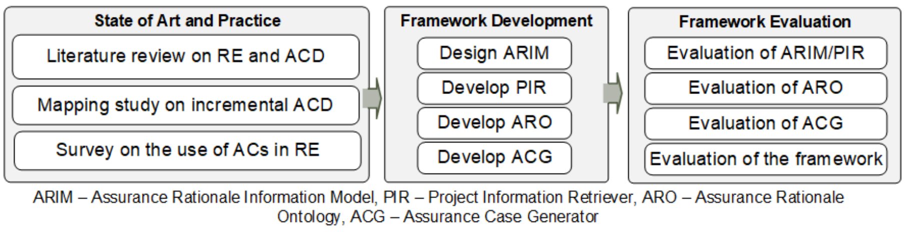
State of Art and Practice
Literature review on RE and ACD
Mapping study on incremental ACD
Survey on the use of ACs in RE
Framework Development
Design ARIM
Develop PIR
Develop ARO
Develop ACG
Framework Evaluation
Evaluation of ARIM/PIR
Evaluation of ARO
Evaluation of ACG
Evaluation of the framework
ARIM – Assurance Rationale Information Model, PIR – Project Information Retriever, ARO – Assurance Rationale Ontology, ACG – Assurance Case Generator

**Figure 3. Thesis methodology steps.**

**Literature review and Mapping study.** We performed literature review on RE and ACD, that leads to the formulation of the research questions and planning of the systematic mapping study. We then conducted a systematic mapping study (SMS) on incremental ACD. Our aim was to identify and analyse existing approaches for integrating ACD as part of the SDLC in an incremental manner and an analysis of them in terms of expected benefits, points of integration with SDLC, roles, activities, and artifacts involved [Almendra et al. 2019]. The main findings of the SMS are: i) an overview of the existing approaches to integrate ACD; ii) ACD is performed in various phases of SDLC; iii) most of the SDLC models identified were agile or incremental-based; iv) 60% of studies put developers in active role of building and/or reviewing AC; v) all studies adopted manual construction of AC and only 20% of the studies used quantitative metrics for assessment. We further discuss these findings in Section 6. From these results, we identify a gap in supporting teams in the incremental development of assurance case within the context of an agile, scope-driven development. This context largely differs from the model-driven approach for which there are solutions for model-based management of assurance cases [de la Vara et al. 2018].

**Survey.** We are planning a survey with assurance case researchers and practitioners. The main questions that we want to gain insights are: “What are the benefits and pitfalls of incremental ACD in the context of RE activities?”, “How the tracing information between requirements and safety evidence is created and managed?”, and “Which kind of assurance information is difficult to recall in later stages of development?”. It is a cross-sectional survey administered through a web-based questionnaire. The population is professionals involved in the development of critical systems and that have any experience with ACD. The sample will be constituted by convenience from participants invited, selected from database of previous surveys applied by research partners, authors of studies selected in previous literature reviews, and practitioners found in professional social networks (e.g. LinkedIn).

## 5. Next steps

**Framework development.** We have drafted the framework (see Section 3.1). We are currently developing the ARIM.

**Evaluation of ARIM.** One real-world PMT will be adapted to incorporate the ARIM (Figure 2-A). The PIR will be customized to extract information from the adapted PMT. This adaptation serves as an initial illustrative evaluation of the ARIM and PIR. Furthermore, we aim to empirically evaluate the adequacy of ARIM to extend PMTs with assurance rationale information. Thus, we will conduct a case study that focus on the observation of developers adapting the ARIM and the PIR to other real-world PMTs. We expected both ARIM, PIR and their companion documentation are sufficient for guiding developers into customizing PMTs for assurance case rational recording.

**Evaluation of ARO.** An ontology is a knowledge-based system, and shall be evaluated by the information it helps to leverage. The evaluation criteria for an ontology content are consistency, completeness, conciseness, expandability and sensitiveness. The experts will be presented to ARO implementation and a running illustrative scenario. Their feedback will be gathered by asking open-ended questions. The objective of this evaluation is to identify gaps or errors in the ontology design for further refinement.

**Evaluation of ACG.** We will investigate how the ACG could improve the construction of ACs in the context of requirements engineering activities. We compare the ACG with a traditional manual construction of ACs by means of a controlled experiment. The manual construction is often used for the incremental development of ACs [Almendra et al. 2019]. We plan to compare these two approaches (factors of the experiment design) about their effectiveness for supporting the construction of assurance cases and its arguments based on project management information, safety information and assurance rationale. We plan to measure the effectiveness by means of how much it influences in subject's ability to associate correct information to ACs. The use of ACG is expected to increase the numbers of correctly stated elements, and to decrease the number of incorrectly stated and missing elements.

**Evaluation of the framework.** The overall solution proposed by the framework will be evaluated by experts (researchers and practitioners). The framework will be presented by means of a working instance (demo) of the tools, illustrative scenarios and instructional material. The idea is to evaluate the perception of experts about the usability of the framework. We understand that a real case study would provide rich and insightful information for the evaluating this framework. In case of finding a suitable team/company to conduct such case studies, out methodology will be updated. Notwithstanding, the methodology planned so far seeks to evaluate the framework from different perspectives.

## 6. Comparison with Related Work

A comparison with related work is presented in Table 2. We selected five aspects for comparison, these aspects are those that we will propose contributions. The first two aspects are *AC Planning* and *AC Construction*, these aspects refer to if and how occurs ACD during planning or development phases of software life cycle. In case of agile methods, planning refers to any activities performed before the iterations, and construction refers to activities performed during or in the end of iterations. *Traceability Management* refers to how the links between hazards, safety requirements and design decisions are organized in PMTs. *Rationale Management* refers to how arguments, justifications, assumptions and any other rationale information is maintained. The last aspect is *Argumentation Perspective*, which refers to how the assurance argumentation is organized from top claims down to evidences. We identified that most works suggests that an initial assurance argumentation is produced before the beginning of development iterations. In the subsequent iterations, these preliminary cases are refined and extended to address the assurance arguments related to the scope selected for implementation. In all works, there is no semi or full automation for assurance case generation. The lack of expertise and guidance in the construction of the arguments is regarded as a recurring problem [Cheng et al. 2018, Doss and Kelly 2016].

**Table 2. Comparison of related works.**

| **Work** | **AC planning** | **AC construction** | **Traceability management** | **Assurance rationale management** | **Argumentation presentation** |
|---|---|---|---|---|---|
| Hall & Rapanotti (2008) | Safety requirements and hazards stated as initial *problems* | Unified process *transforms* problems into design decisions and assurance arguments | Requirements and design decisions stated directly in the *problem transformations* | Recorded in a set of *problem transformations* | Problem-oriented |
| Ge et al. (2010) | Initial AC produced based on module internal architecture | Manual construction/update during iterations | Not discussed | Recorded in the assurance case | Software module-oriented |
| Stålhane & Myklebust (2016) | Initial AC produced based on hazard identification | Manual construction/update during iterations | Directly in the assurance case | Recorded in the assurance case | Hazard-oriented |
| Cleland-Huang & Vierhauser (2018) | Not discussed | Manual construction/update after iterations | Embedded in PMTs | Recorded in the assurance case | Requirements-oriented |

We found three different approaches for traceability management, in the works that discussed this aspect. The problem-oriented approach [Hall and Rapanotti 2008] combines much information in the problem transformations, although requirements and architecture information probably have external supporting documents. In [Stålhane and Myklebust 2016], it is proposed that the assurance case needs to refer to relevant documents such as code, test results, review results and solution evidences. A different approach is proposed by [Cleland-Huang and Vierhauser 2018], trace links between hazard, safety requirements and design decisions are managed inside PMTs. This tracing information is then used to support the construction of the arguments, together with the assurance rationale.

We found that all approaches manage the assurance rationale inside the assurance case arguments. In fact, this is what assurance case notation language and problem transformations are designed for. In all approaches, AC are external artifacts that have to be maintained alongside the development. A consequence is that assurance case structure tend to follow a specific perspective, and to view the same information organized in another perspective, it requires the construction of a new case.

## 7. Conclusions

We presented a proposal for a framework to support the development of assurance cases in the context of Agile RE activities. Our proposal contributes to the state of the art in three aspects. First, we propose to enhance project management tools with assurance rationale information, that would be managed iteratively and incrementally alongside the development. Second, we propose a mechanism to generate AC from information stored in project management tools. It will use information commonly recorded in project repositories and the assurance rationale management model we will propose. Finally, we will support the generation of assurance arguments in multiple perspectives, thus reducing the effort to analyze the assurance rationale and project information in different ways.

## Acknowledgements

The author thanks CNPq for funding the execution of this work.

## References

Almendra, C., Vilela, J., , and Silva, C. (2019). A systematic mapping study on incremental development of safety cases. *Yet to be published.*

Bloomfield, R. and Bishop, P. (2010). Safety and assurance cases: Past, present and possible future – an adelard perspective. In Dale, C. and Anderson, T., editors, *Making Systems Safer*. Springer.

Burge, J. E., Carroll, J. M., McCall, R., and Mistrik, I. (2008). *Rationale and Requirements Engineering*, pages 139–153. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cheng, J., Metoyer, R., Cleland-huang, J., Dame, N., and Dame, N. (2018). How Do Practitioners Perceive Assurance Cases in Safety-Critical Software Systems ? In *Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 5–8. ACM.

Cleland-Huang, J. and Vierhauser, M. (2018). Discovering , Analyzing , and Managing Safety Stories in Agile Projects. *IEEE 26th International Requirements Engineering Conference*.

de la Vara, J. L., Ruiz, A., and Espinoza, H. (2018). Recent Advances towards the Industrial Application of Model-Driven Engineering for Assurance of Safety-Critical Systems. In *6th International Conference on Model-Driven Engineering and Software Development.*, pages 632–641.

Doss, O. and Kelly, T. P. (2016). Challenges and opportunities in agile development in safety critical systems: A survey. *SIGSOFT Softw. Eng. Notes*, 41(2):30–31.

Ebert, C. (2015). Implementing functional safety. *IEEE Software*, 32(5):84–89.

Ge, X., Paige, R. F., and McDermid, J. A. (2010). An Iterative Approach for Development of Safety-Critical Software and Safety Arguments. In *Agile Conference*, pages 35–43.

Hall, J. G. and Rapanotti, L. (2008). Assurance-Driven Design. In *2008 The Third International Conference on Software Engineering Advances*, pages 379–388.

Hatcliff, J., Wassyng, A., Kelly, T., Comar, C., and Jones, P. (2014). Certifiably safe software-dependent systems: Challenges and directions. In *Proceedings of the on Future of Software Engineering*, FOSE 2014, pages 182–200, New York, NY, USA. ACM.

Heeager, L. T. and Nielsen, P. A. (2018). A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology*, 103(July):22–39.

Kelly, T. P. (1999). *Arguing safety: a systematic approach to managing safety cases*. PhD thesis, University of York.

Kotonya, G. and Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques*. Wiley Publishing, 1st edition.

Leffingwell, D. (2010). *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional.

Lutz, R. R. (2000). Software engineering for safety: a roadmap. In *Proceedings of the Conference on The Future of Software Engineering*, pages 213–226. ACM.

Myklebust, T., Hanssen, G. K., and Lyngby, N. (2017). A survey of the software and safety case development practice in the railway signalling sector. In *European Safety and Reliability Conference (ESREL)*.

Schön, E.-M., Thomaschewski, J., and Escalona, M. J. (2017). Agile requirements engineering: A systematic literature review. *Computer Standards & Interfaces*, 49:79–91.

Stålhane, T. and Myklebust, T. (2016). The agile safety case. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*.

Vilela, J., Castro, J., Martins, L. E. G., and Gorschek, T. (2017). Integration between requirements engineering and safety analysis: A systematic literature review. *Journal of Systems and Software*, 125:68 – 92.