



UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CAMILO CAMILO ALMENDRA

On the Integration of Requirements Engineering and Assurance Case Development

Recife

2023

CAMILO CAMILO ALMENDRA

On the Integration of Requirements Engineering and Assurance Case Development

A Ph.D. Thesis presented to the Center of Informatics of Universidade Federal de Pernambuco in partial fulfillment of the requirements for the degree of Philosophy Doctor in Computer Science.

Main Area: Software Engineering and Programming Languages

Advisor: Profa. Dra. Carla Taciana Lima Lourenço Silva Schuenemann

Recife

2023

Catálogo na fonte
Bibliotecária Luiza M.P. de Oliveira, CRB4-1316

A448i Almendra, Camilo Camilo
 On the Integration of Requirements Engineering and Assurance Case
 Development. / Camilo Camilo Almendra. – 2023.
 237 f.: il., fig., tab.

 Orientadora: Carla Taciana Lima Lourenço Silva Schuenemann.
 Tese (Doutorado) – Universidade Federal de Pernambuco. CCEN, Ciência
 da Computação, Recife, 2023.
 Inclui referências.

 1. Engenharia de requisitos. 2. Casos de garantia. 3. Ontologia. 4. Sistemas
 críticos. I. Schuenemann, Carla Taciana Lima Lourenço Silva (orientadora). II.
 Título.

 005.1 CDD (23. ed.) UFPE- CCEN 2023 - 194

Camilo Camilo Almendra

“On the Integration of Requirements Engineering and Assurance Case Development”

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Engenharia de Software e Linguagens de Programação

Aprovada em: 15/09/2023.

Orientadora: Profa. Dra. Carla Taciana Lima Lourenço Silva Schuenemann

BANCA EXAMINADORA

Prof. Dr. Alexandre Marcos Lins de Vasconcelos
Centro de Informática/ UFPE

Profa. Dra. Jéssyka Flavianne Ferreira Vilela
Centro de Informática/ UFPE

Profa. Dra. Monalessa Perini Barcellos
Departamento de Informática / UFES

Profa. Dra. Márcia Jacyntha Nunes Rodrigues Lucena
Departamento de Informática e Matemática Aplicada / UFRN

Prof. Dr. André Luiz de Oliveira
Departamento de Ciência da Computação / UFJF

To my family.

ACKNOWLEDGEMENTS

I would like to express my acknowledgements in Portuguese:

“À minha esposa Germana, meus pais Aluizio e Liduina, meus irmãos Aluizio e Raquel, meus filhos Guilherme e Heitor, por todo o apoio direto e indireto para que eu pudesse me dedicar a este trabalho.

À minha orientadora Carla, pela paciência, apoio, críticas, elogios e tempo dedicados a me guiar por este trabalho. Sua orientação foi imprescindível para os resultados alcançados.

A todos os parceiros de pesquisa que colaboraram na produção dos artigos, em especial ao grupo de pesquisa DARE.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pelo suporte financeiro através do processo nº 141642/2017-3.

Às pessoas que fazem o Centro de Informática/UFPE, por todo o suporte nas disciplinas e no apoio na execução deste trabalho.

Aos colegas do Campus Quixadá/UFC, parentes e amigos, pelo apoio e incentivo.

A todas as pessoas que participaram dos estudos envolvidos neste trabalho, pelo tempo e disposição em compartilhar seus conhecimentos.

À banca examinadora, pela disponibilidade em participar da avaliação deste trabalho.

A todas as pessoas que colaboram com o desenvolvimento dos materiais e ferramentas abertas usadas neste trabalho, pelo esforço dedicado em prol da sociedade.

A todas minhas professoras e professores, desde o jardim de infância até o doutorado, por se dedicarem à profissão mais importante do mundo e por compartilharem seus conhecimentos comigo.”

ABSTRACT

Assurance Case Development (ACD) is an emerging approach for demonstrating that a system is safe. Assurance cases (ACs) comprise various project information, including requirements and their traceability to other artefacts. It is challenging for teams to build ACs as they have to identify and gather the project information relevant to the argumentation and assess its consistency and completeness. Moreover, regular project information (e.g., requirements, design, rationale) and assurance-related information (e.g., hazard, causes, mitigation strategies) are likely managed separately. In this context, software and assurance teams lack a shared information repository on which they can cross-collaborate, integrate artefacts, and review traceability. Motivated by this scenario, we propose a framework called AssuRance CAsE DEvelopment (ARCADE) to support teams in managing assurance information inside regular project management tools and to provide automated assessment and AC generation. This research was performed in four steps. First, a Systematic Mapping Study to investigate the existing approaches for incremental ACD. Second, a practitioners' survey to understand how requirements engineering (RE) and ACD activities currently interplay. Third, the development of the framework. Fourth, ARCADE evaluation through a) illustrative scenarios; and b) a qualitative study with practitioners. The first step resulted in the identification of 16 approaches for incremental development of ACs through the development lifecycle. The second step resulted in new evidence that integration between RE and ACD occurs across all RE activities, and that practitioners perceive benefits such as raising safety assurance awareness, early traceability development, and early identification of assurance evidence needs. Moreover, practitioners see requirements specification and change request analysis as the most suitable activities for integrating ACD and RE. In the third step, ARCADE framework was designed and implemented to address some of the research and practice gaps identified in the previous studies. The core of the framework is an operational ontology that interrelates concepts from RE and AC domains. The framework tool automates the retrieval of data from project management tools, to ingest into an ontology reasoner. The framework automation encompasses the extraction of data from the tools, the data mapping into semantic triples, the ingestion of the triples into an ontology reasoner, the quality analysis, and the collection of data items to generate the AC fragments. In the fourth step, ARCADE was evaluated through two illustrative scenarios based on publicly available datasets. The analysis of reports and ACs generated for the datasets resulted in identification of traceability gaps and AC fragments based on different

argumentation patterns. In the qualitative study with practitioners, we gather their perceptions on the ARCADE tool features. As evidenced in the evaluations, ARCADE has the potential to help teams to collaborate across requirements and assurance artefacts using regular project management tools, thus fostering early and continuous review of safety assurance aspects of the system under development.

Keywords: requirements engineering; assurance cases; ontology; safety-critical systems.

RESUMO

Assurance Case Development (ACD) é uma abordagem emergente para demonstrar segurança de sistemas. Casos de garantia (CG) compreendem várias informações de projeto, incluindo requisitos e sua rastreabilidade para outros artefatos. É desafiador para equipes construírem CGs, pois precisam identificar e reunir informações de projeto relevantes para a argumentação e avaliar sua consistência e integridade. Além disso, informações de projeto (por exemplo, requisitos, design, justificativa) e relacionadas à garantia (por exemplo, perigo, causas, estratégias de mitigação) provavelmente são gerenciadas separadamente. Assim, equipes de software e de garantia de qualidade carecem de repositórios compartilhados de informações no qual possam colaborar, integrar artefatos e revisar rastreabilidade. Motivados por esse cenário, propomos o framework AssuRance CAse DEvelopment (ARCADE) para apoiar equipes no gerenciamento de informações de garantia de segurança dentro de ferramentas de gerenciamento de projetos e fornecer recursos automatizados de avaliação e geração de CGs. Esta pesquisa foi realizada em quatro etapas. Primeiro, um estudo de mapeamento sistemático para investigar abordagens existentes para desenvolvimento incremental de CGs. Segundo, uma pesquisa com profissionais para entender como atividades de engenharia de requisitos (ER) e de ACD interagem atualmente. Terceiro, o desenvolvimento do framework. Quarto, a avaliação do ARCADE por meio de a) cenários ilustrativos; e b) um estudo qualitativo com profissionais. A primeira etapa resultou na identificação de 16 abordagens de desenvolvimento incremental de CGs ao longo do ciclo de desenvolvimento. A segunda etapa resultou em novas evidências de que a integração entre ER e ACD ocorre em todas as atividades de ER, e que os profissionais percebem benefícios como aumento da compreensão sobre garantia de segurança, desenvolvimento precoce de rastreabilidade e identificação precoce das necessidades de evidências de garantia. Além disso, os profissionais veem a especificação de requisitos e a análise de mudanças como atividades mais adequadas para integrar ACD e ER. Na terceira etapa, o framework ARCADE foi desenvolvido para abordar lacunas de pesquisa e prática identificadas nos estudos anteriores. O núcleo do framework é uma ontologia operacional que inter-relaciona conceitos de RE e ACD. O framework automatiza a recuperação de dados de ferramentas de gerenciamento de projetos, para inserir em um raciocinador de ontologia. A automação envolve a extração de dados, o mapeamento dos dados para triplas semânticas, a ingestão das triplas em um raciocinador de ontologia, a análise de qualidade, e a coleta de dados para geração de fragmentos de CGs. Na quarta etapa, o ARCADE foi avaliado através de dois cenários

ilustrativos baseados em dados disponíveis ao público. A análise dos relatórios e CGs gerados para os conjuntos de dados resultou na identificação de lacunas de rastreabilidade e fragmentos de CGs com base em diferentes padrões de argumentação. No estudo qualitativo com profissionais, coletamos percepções sobre recursos do ARCADE. Conforme evidenciado nas avaliações, o ARCADE tem potencial de ajudar equipes a colaborar em requisitos e artefatos de garantia usando ferramentas de gerenciamento de projetos, promovendo assim a revisão precoce e contínua dos aspectos de segurança do sistema em desenvolvimento.

Palavras-chave: engenharia de requisitos; casos de garantia; ontologia; sistemas críticos.

LIST OF FIGURES

Figure 1 – An illustration of the relationship among claims, arguments and evidence in an assurance argument	25
Figure 2 – Example of assurance case argument structure	25
Figure 3 – Thesis methodology steps.	31
Figure 4 – Assurance case construction methodologies	39
Figure 5 – Argument example with rationale information	43
Figure 6 – Another argument example with rationale information	44
Figure 7 – RDF triple	52
Figure 8 – RDF graph example	52
Figure 9 – Study selection methodology steps.	62
Figure 10 – Coding example from analysis of Survey Q3	89
Figure 11 – Category roles in the organisation	93
Figure 12 – Countries in which participants have worked	94
Figure 13 – Nature of need for AC development for all participants and grouped by role of organisation	95
Figure 14 – Frequency of participation in ACD activities for all participants	96
Figure 15 – ACD practices observed by all participants and grouped by the organisation's role	97
Figure 16 – AC report organisation	98
Figure 17 – Traceability management scenarios	99
Figure 18 – Frequency of which ACs are consulted during RE activities for all participants	99
Figure 19 – Safety assurance consideration during RE activities for all participants and grouped by the organisation's role	101
Figure 20 – Collaboration scenarios and the frequency of occurrence for all participants	101
Figure 21 – Opinion on ACD during RE activities for all participants and grouped by experience in ACD	104
Figure 22 – Common problems in SCS development that could benefit from RE and ACD integration	105
Figure 23 – Opinion on the integration of ACD with RE activities for all participants and grouped by experience in ACD	105

Figure 24 – Illustrative scenario for collaborative work between safety and software teams.	114
Figure 25 – Overview of assurance case development supported by ARCADE	116
Figure 26 – Information metamodel of Safety Assurance Ontology	127
Figure 27 – Information Retriever internal workflow	136
Figure 28 – Inconsistency Explanation	137
Figure 29 – Consistency Analysis internal workflow	138
Figure 30 – Project data provided and inferred	139
Figure 31 – Answers for competency questions	139
Figure 32 – Knowledge Discovery internal workflow	140
Figure 33 – Assurance Case Generator workflow	142
Figure 34 – Assurance case fragment for Functional Decomposition pattern in GSN notation	142
Figure 35 – Assurance case fragment for Hazard Avoidance pattern in GSN notation . .	142
Figure 36 – (a) Assurance case fragment. (b) Assurance and project data represented in AIM. (c) Same data mapped to Jira.	146
Figure 37 – (a) Safety requirement represented in Story issue. (b) Issue link tree visualisation.	147
Figure 38 – GSN notation and a safety argument example	189
Figure 39 – A patient-controlled analgesia infusion pump.	198
Figure 40 – Open PCA pump use	199
Figure 41 – NOR-STA visual notation for assurance arguments.	204
Figure 42 – GSN diagram for Claim 0.	205
Figure 43 – GSN diagram for Claim 1.	205
Figure 44 – GSN diagram for Claim 1.1.	206
Figure 45 – GSN diagram for Claim 2.2.	207
Figure 46 – GSN diagram for Claim 2.2.A.5.	208
Figure 47 – Custom issue types	221
Figure 48 – Custom fields	221
Figure 49 – Custom issue link types	222
Figure 50 – View of all issues	222
Figure 51 – A safety requirement recorded as a Story issue with label “safety”	223
Figure 52 – A hazard recorded as a Hazard issue	223

LIST OF SOURCE CODES

Source Code 1	–	OWL specification for <i>mitigation</i> relationship and associated rules	. 129
Source Code 2	–	Example of data represented in the ARCADE interchange format	. . 130
Source Code 3	–	SPARQL queries for two competency questions 140
Source Code 4	–	SPARQL query for Functional Decomposition Pattern 143
Source Code 5	–	Safety Assurance Ontology OWL Specification 211

LIST OF TABLES

Table 1 – Methodological framework	31
Table 2 – Mapping of evidence types and source processes	42
Table 3 – Overview of related works on integrated management	56
Table 4 – Research questions and motivations.	60
Table 5 – SMS selected studies.	63
Table 6 – Quality scores and number of citations for each selected study.	64
Table 7 – Venues of publication.	65
Table 8 – Overview of the selected studies (RQ1).	67
Table 9 – Classification of studies by SDLC Model, Task Integration and Argument Development	68
Table 10 – Roles involved in incremental ACD approaches	73
Table 11 – Expected outcomes of incremental ACD approaches	75
Table 12 – Challenges addressed by incremental ACD approaches	77
Table 13 – Structuring techniques of incremental ACD approaches	79
Table 14 – Assessment techniques and categories (RQ5).	80
Table 15 – Terms used to denote development of ACs alongside the SDLC.	84
Table 16 – Survey questions	90
Table 17 – Frequency of participation in ACD activities grouped by experience in ACD .	96
Table 18 – Frequency in which ACs are consulted during RE activities for all participants and grouped by type of organisation	100
Table 19 – Collaboration scenarios and the frequency of occurrence grouped by the organisation's role	102
Table 20 – Comparison of state of practice and opinion on the integration of RE and ACD	106
Table 21 – Knowledge sources for Safety Assurance Ontology	124
Table 22 – Competency questions for Safety Assurance Ontology	126
Table 23 – Information model concepts and relations of Safety Assurance Ontology . .	127
Table 24 – Formation and classification rules of Safety Assurance Ontology	128
Table 25 – ARCADE requirements	135
Table 26 – Mapping of AIM concepts into Jira.	145

Table 27 – Mapping of AIM relationships into Jira.	146
Table 28 – Pilot participants' profile	154
Table 29 – Participants' profile	156
Table 30 – Task and opinion questions for Part I – ES-RQ1	157
Table 31 – Participant answers for steps 1.4.1 to 1.4.8 (ES-RQ1)	158
Table 32 – Opinion questions for Part I – ES-RQ2	161
Table 33 – Participant answers for question 1.5.2 (ES-RQ2)	161
Table 34 – Task and opinion questions for Part II – ES-RQ3	164
Table 35 – Participant answers for steps 2.2.2 to 2.2.7 (ES-RQ3)	165
Table 36 – Opinion questions for Part II – ES-RQ4	167
Table 37 – Participants' answers for steps 2.3.1 to 2.3.4 (ES-RQ4)	167
Table 38 – Opinion questions for Part II – ES-RQ5	168
Table 39 – Participants' answers for steps 2.3.5 to 2.3.6 (ES-RQ5)	169
Table 40 – Tracing information among project items.	204
Table 41 – Examples of relationships between AC elements and project items.	208
Table 42 – Examples of argument elements that bring new information.	209

LIST OF ABBREVIATIONS AND ACRONYMS

ABox	Assertional Axioms
AC	Assurance Case
ACD	Assurance Case Development
AIM	ARCADE Information Model
ALM	Application Lifecycle Management
CA	Consistency Analysis
CQ	Competency Question
DL	Description Logic
GSN	Goal Structuring Notation
IR	Information Retriever
ITS	Issue Tracking System
KD	Knowledge Discovery
OWL	Ontology Web Language
RE	Requirements Engineering
SAO	Safety Assurance Ontology
SCS	Safety-Critical System
SDLC	Software Development Life Cycle
SMS	Systematic Mapping Study
TBox	Terminological Axioms

CONTENTS

1	INTRODUCTION	22
1.1	CONTEXT	22
1.2	PROBLEM	27
1.3	RESEARCH QUESTIONS AND OBJECTIVES	29
1.4	RESEARCH METHODOLOGY	31
1.4.1	Research Classification	31
1.4.2	Research Steps	31
1.5	SUMMARY OF PUBLICATIONS	32
1.6	THESIS ORGANISATION	33
2	BACKGROUND AND RELATED WORKS	34
2.1	ASSURANCE CASE DEVELOPMENT	34
2.1.1	Problems Faced by Practitioners	35
2.1.2	Assurance Case Structure	36
2.1.3	Argument Development	38
2.1.4	Assurance Evidence Management	41
2.2	REQUIREMENTS ENGINEERING FOR SAFETY-CRITICAL SYSTEMS . .	44
2.2.1	Requirements Traceability	45
2.2.2	Issue Tracking Systems	46
2.3	SEMANTIC WEB	47
2.3.1	Description Logic	48
2.3.2	Semantic Triples	51
2.4	PREVIOUS RESEARCH ON THE TOPIC	52
2.4.1	Incremental Assurance Case Development	52
2.4.2	Requirements Engineering and Assurance Case Development	53
2.4.3	Integrated Management of Project and Assurance Information . . .	55
2.5	CHAPTER SUMMARY	58
3	INCREMENTAL DEVELOPMENT OF ASSURANCE CASES . . .	59
3.1	SMS RESEARCH METHODOLOGY	59
3.1.1	Search Strategy	60
3.1.2	Inclusion and Exclusion Criteria	61

3.1.3	Study Selection	62
3.1.4	Quality Assessment	63
3.1.5	Venues of Publication	65
3.2	SMS RESULTS AND ANALYSIS	66
3.2.1	Overview of the Studies (RQ1)	66
3.2.2	Integration of ACD and SDLC (RQ2)	68
3.2.2.1	<i>SDLC Model</i>	69
3.2.2.2	<i>Task Integration</i>	71
3.2.2.3	<i>Argument Development</i>	71
3.2.3	Roles Involved (RQ3)	73
3.2.4	Expected Outcomes (RQ4)	74
3.2.5	Challenges and Needs (RQ5)	76
3.2.6	Techniques and Tools (RQ6)	78
3.2.7	Discussion	80
3.3	SMS THREATS TO VALIDITY	83
3.4	CHAPTER SUMMARY	84
4	REQUIREMENTS ENGINEERING AND ASSURANCE CASE DEVELOPMENT INTERPLAY	86
4.1	SURVEY METHODOLOGY	86
4.1.1	Research Questions	86
4.1.2	Survey Design	87
4.1.3	Pilot Application	88
4.1.4	Survey Instrument	88
4.1.5	Survey Sampling and Application	89
4.2	SURVEY RESULTS AND ANALYSIS	92
4.2.1	How did you find this survey? (Q1)	92
4.2.2	How long have you been working on the development of safety-critical systems? (Q2)	92
4.2.3	What is your current main role in the organisation? (Q3)	93
4.2.4	How many projects that involved Assurance Cases for safety-critical systems have you participated in? (Q4)	93
4.2.5	In which country(ies) have you principally worked upon Assurance Case or RE for safety-critical systems? (Q5)	94

4.2.6	What is the main role of the organisation for which you have worked regarding the development of safety-critical systems? (Q6)	94
4.2.7	Is the development of Assurance Cases a mandatory, recommended, or optional activity? (Q7)	94
4.2.8	How often have you been involved in these following Assurance Case Development activities? (Q8)	95
4.2.9	Which of the following statements apply to how Assurance Cases (ACs) are developed in your projects? (Q9)	96
4.2.10	Which is the main notation used to construct the Assurance Arguments? (Q10)	97
4.2.11	Which of the following scenarios best describes how Assurance Case Reports are organised, considering only one system to be certified? (Q11)	97
4.2.12	Is there any kind of automation in the creation or maintenance of Assurance Arguments and Reports? (Q12)	98
4.2.13	Considering the management of traceability among requirements and safety analysis information, which of the following scenarios best describes the most used practice in your projects? (Q13)	98
4.2.14	How often are Assurance Cases consulted during Requirements Engineering activities? (Q14)	99
4.2.15	How often does your team consider safety assurance when performing Requirements Engineering activities? (Q15)	100
4.2.16	Which of the following collaborations between requirements and safety specialists occur, and in which frequency? (Q16)	100
4.2.17	What are the benefits of Requirements Engineers' participation in Assurance Case Development or safety analysis development activities? (Q17)	102
4.2.18	What are the benefits of Safety Engineers' participation in Requirements Engineering activities? (Q18)	103
4.2.19	How much do you agree with the development of Assurance Cases during Requirements Engineering activities? (Q19)	103

4.2.20	To what extent do you believe the development of Assurance Cases during Requirements Engineering activities could mitigate the following problems in SCS development? (Q20)	103
4.2.21	In your opinion, with which Requirements Engineering activities do you think the development or sketching of Assurance Cases could be combined? (Q21)	104
4.3	SURVEY DISCUSSION	107
4.3.1	Participant Profiles	107
4.3.2	State of ACD Practice	107
4.3.3	State of Integration	109
4.3.4	Views of Practitioners on Integration	110
4.4	SURVEY THREATS TO VALIDITY	111
4.4.1	Construct Validity	111
4.4.2	External Validity	111
4.4.3	Internal Validity	112
4.5	CHAPTER SUMMARY	112
5	FRAMEWORK DESIGN	114
5.1	GOALS	115
5.1.1	Framework Usage Overview	116
5.1.2	Design Decisions	117
5.1.3	Expected Benefits	120
5.2	SAFETY ASSURANCE ONTOLOGY	121
5.2.1	Methodology	121
5.2.2	Specification	122
5.2.2.1	<i>Purpose and Scope</i>	122
5.2.2.2	<i>Intended Users and Scenarios of Use</i>	122
5.2.2.3	<i>Sources of Knowledge</i>	123
5.2.2.4	<i>Competency Questions</i>	125
5.2.2.5	<i>Information Model</i>	126
5.2.3	Implementation	129
5.2.4	Evaluation	131
5.3	CHAPTER SUMMARY	132
6	FRAMEWORK IMPLEMENTATION	133

6.1	DEVELOPMENT METHODOLOGY	133
6.2	REQUIREMENTS	134
6.3	INFORMATION RETRIEVER	135
6.4	CONSISTENCY ANALYSIS	137
6.5	KNOWLEDGE DISCOVERY	138
6.6	ASSURANCE CASE GENERATOR	140
6.7	ADOPTING ARCADE	143
6.8	CHAPTER SUMMARY	144
7	EVALUATION	145
7.1	ILLUSTRATIVE INTEGRATION WITH AN ISSUE TRACKER	145
7.2	ILLUSTRATIVE SCENARIO WITH PUBLIC DATASETS	148
7.2.1	Open Patient-Controlled Analgesia Pump Project	148
7.2.2	Dronology Project	149
7.3	EXPERT SURVEY EVALUATION	150
7.3.1	Expert Survey Methodology	150
7.3.1.1	<i>Research Questions</i>	150
7.3.1.2	<i>Short Video Lectures</i>	151
7.3.1.3	<i>Participant Procedure</i>	152
7.3.1.4	<i>Pilot Application</i>	154
7.3.1.5	<i>Sampling</i>	155
7.3.2	Expert Survey Results and Analysis	156
7.3.2.1	<i>ES-RQ1 – How useful and easy to use is the ARCADE information model to facilitate the organisation of assurance-relevant information in the regular project repositories?</i>	157
7.3.2.2	<i>ES-RQ2 – How useful is the ARCADE information model to represent core concepts/relationships related to assurance and project information?</i>	160
7.3.2.3	<i>ES-RQ3 – How useful and easy to use is it to explore assurance-relevant information (retrieved from the project repository) using the ARCADE web tool?</i>	164
7.3.2.4	<i>ES-RQ4 – How useful are the ARCADE web tool features (Consistency Analysis, Knowledge Explorer, Assurance Case Generator)?</i>	167
7.3.2.5	<i>ES-RQ5 – Does ARCADE tool bring the expected benefits of our design goals?</i>	168

7.3.3	Expert Survey Threats to Validity	170
7.4	CHAPTER SUMMARY	171
8	CONCLUSIONS	172
8.1	ANSWERING THE RESEARCH QUESTIONS	173
8.2	CONTRIBUTIONS	173
8.3	LIMITATIONS	175
8.4	FUTURE RESEARCH	176
	REFERENCES	178
	APPENDIX A – GOAL STRUCTURING NOTATION EXAMPLE .	189
	APPENDIX B – SURVEY SUPPLEMENTARY MATERIAL	191
	APPENDIX C – OPEN PATIENT-CONTROLLED ANALGESIC	
	PUMP	197
	APPENDIX D – SAFETY ASSURANCE ONTOLOGY OWL SPEC-	
	IFICATION	211
	APPENDIX E – CUSTOMIZED JIRA INSTANCE	221
	APPENDIX F – INFORMED CONSENT AND PERSONAL PRO-	
	FILE QUESTIONNAIRES	224
	APPENDIX G – EVALUATION PROCEDURES	230

1 INTRODUCTION

This chapter presents the context and problem of this thesis and discusses our research questions, objectives and proposed methodology. We conclude the chapter with an overview of the thesis-related publications and organisation.

1.1 CONTEXT

The concept of safety is the freedom from accidents (LEVESON, 2016). Accidents are any unplanned event that may result in a loss of human life, harm, injury, property damage or environmental disaster. The concept of hazard is associated with any system state that together with worst-case environment conditions may lead to an accident. The avoidance of accidents by seeking to eliminate or mitigate hazards is the main objective of the safety engineering.

The presence of computational systems in several human activities brings concerns regarding harms or damages that software systems could cause to people, organisations or environment. Safety-Critical System (SCS) are system whose failure can lead to hazard situations. Development of SCS requires identification and analysis of potential hazards so that they could be eliminated or mitigated by design (LUTZ, 2000). For these kind of system, quality assurance is paramount and shall be addressed in the beginning of the development (HEIMDAHL, 2007).

Software-intensive system refers to electronic systems in which software is a relevant part of the design. Many application domains such as aerospace, avionics, automobiles and medical devices have seen an explosive growth in use of software (HEIMDAHL, 2007).

SCS development poses many challenges because eventual failures can produce great, irreversible harm to people, the environment and infrastructure. Software is an integral part of modern embedded, electrical/electronic and control systems, permeating several domains, such as aviation, military, automotive, and medical devices (HATCLIFF et al., 2014). Therefore, more scrutiny must be applied to the development processes of software-intensive critical systems, as regulations demand the fulfilment of high-level safety requirements and process compliance obligations (EBERT, 2015).

Safety requirement refers to characteristics and behaviours of the system that should be satisfied to avoid the occurrence of hazards or to mitigate them (VILELA et al., 2018). It may be

a normal function, data, interface, and quality requirement that is relevant to the achievement of safety. It may also be a description of actions and/or constraints that should or should not be performed to maintain the system in a safe state. Finally, it may be an architecture or design constraint mandating the use of specific safety mechanism or safeguards.

Requirements Engineering (RE) is a key process for producing quality systems, even more in the context of software-intensive critical systems (MARTINS; GORSCHKE, 2016; VILELA et al., 2017a). As a knowledge-intensive work, requirements processes start from scattered data in multiple sources (stakeholders, regulations, existing products, team expertise, and others) and end up in structured specifications and identified decisions, understood and negotiated among stakeholders (WIEGERS; BEATTY, 2013). Requirements traceability management is a strategic process to ensure that software is safe for use. It encompasses the tracking of links among requirements, design, code, tests, hazards, risks, cause of hazards, and other artefacts. Many regulators of various industry domains have incorporated it into standards and guidelines (MÄDER et al., 2013). Those regulations in many cases define levels of integrity/confidence which are related risk assessment. The higher the risks associated with some functions, more stringent are the requirements the system design has to cope with. However, maintaining the traceability among safety requirements and other artefacts alongside the system development lifecycle is challenging (VILELA et al., 2017a).

Traceability is a fundamental discipline needed to achieve safety assurance goals. Many problems related to traceability could delay the certification of SCS products, just as lack of a traceability information model, uneven granularity of tracing links, and missing or redundant links (MÄDER et al., 2013). The information needed to build a compelling assurance case can be compared to a high-end usage of requirements traceability (RAMESH; JARKE, 2001). Among other characteristics, a high-end usage is indicated by the application of traceability for full coverage of life-cycle, capture of decisions and rationale, capture traces across product and process dimensions, and requirements decomposition (RAMESH; JARKE, 2001).

Requirements decomposition refers to the breakdown of high level requirements into more low level, technical requirements which are then allocated into components or sub-systems. In each level of decomposition, the hazards and risks being addressed need to be clearly stated, and aspects still undeveloped shall be addressed in the next level.

Many information objects need to be traced during development processes, such as requirements, design decisions, components, code, and others (RAMESH; JARKE, 2001). Besides the links among information objects, the rationale may be recorded to explain the interrela-

tion between information objects (RAMESH; JARKE, 2001; BURGE et al., 2008). In the context of SCS development, rationale enhances the confidence to assure that requirements decomposition and design decisions reasonably address all identified risks and hazards. Together with specifications and common traceability, rationale needs to be propagated throughout the development process (HATCLIFF et al., 2014).

SCS are often subject to certification, which refers to “*the process of assuring that a product or process has certain stated properties, which are then recorded in a certificate*” (COUNCIL, 2007). A certification procedure is necessary for the development lifecycle of regulated domains before releasing a critical system for operation (HATCLIFF et al., 2014). Nowadays, some regulations mandate or recommend elaborating an Assurance Case (AC) as part of the certification documentation (DEFENCE..., 2017; CENELEC..., 2018; ISO..., 2018; FDA..., 2014). The purpose of an AC is to present an argumentation in favour of a system. This argumentation shall demonstrate how the system requirements and design handle the known hazards and risks so that development teams and certification authorities can build confidence in the system.

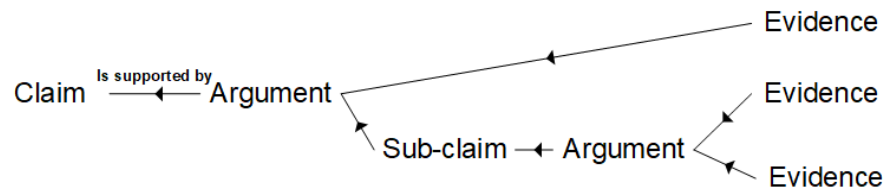
Nowadays, some regulations mandate or recommend elaborating an Assurance Case (AC) as part of the certification documentation:

- Defence Standard 00-056 – Safety Management Requirements for Defence Systems – Part 2 (DEFENCE..., 2017);
- Cenelec EN 50129 – Railway applications – Communication, signalling and processing systems – Safety related electronic systems for signalling (CENELEC..., 2018);
- ISO 26262 – Road vehicles – Functional safety – Part 10 (ISO..., 2018); and
- Infusion Pumps Total Product Life Cycle – Guidance for Industry and FDA Staff (FDA..., 2014).

Assurance cases are built using information produced during various development activities. Assurance arguments are hierarchical, beginning with a top claim that is decomposed into sub-claims or satisfied by a piece of evidence (Figure 1). They amalgamate information from standards, product requirements, risk/hazard analysis, architecture decisions, test specifications, and process execution records (BLOOMFIELD; BISHOP, 2010). All this evidence is bonded together to argue that a specific system version complies with high-level safety requirements and that its design and implementation acceptably address all hazards. Thus,

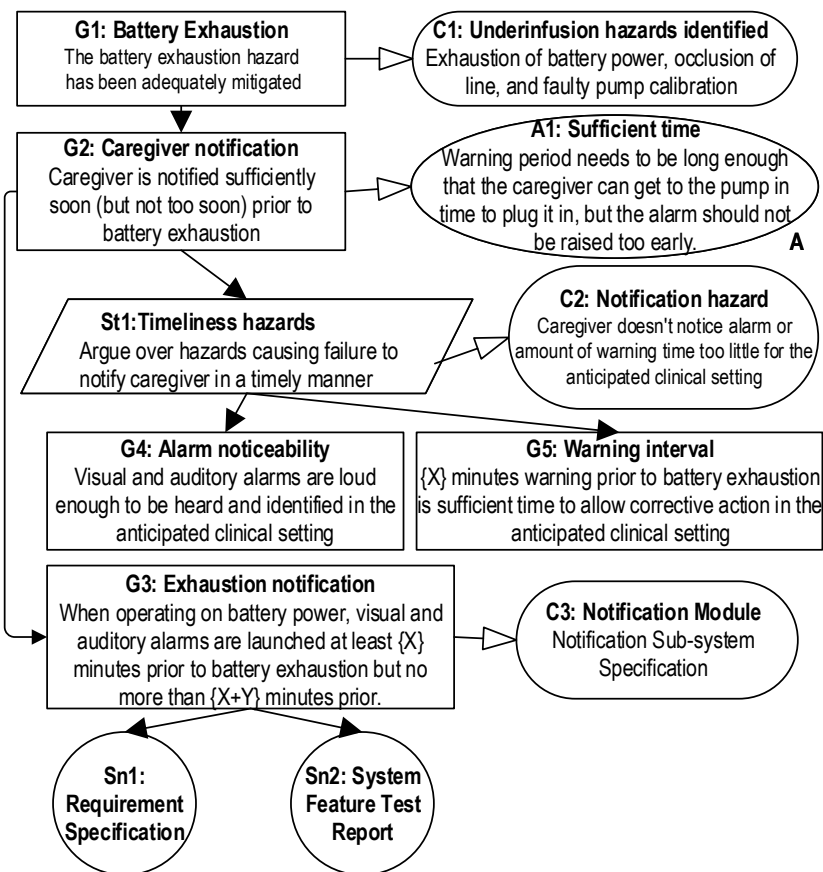
the main objective of an assurance case is to serve as an orchestration of the dependability justification (KELLY, 1999; VARA et al., 2022).

Figure 1 – An illustration of the relationship among claims, arguments and evidence in an assurance argument



Source: adapted from RUSHBY (2015)

Figure 2 – Example of assurance case argument structure



Source: adapted from WEINSTOCK; GOODENOUGH (2009)

Figure 2 presents a partial example of a SAC using the GSN notation (GOAL..., 2021). The example depicts a safety argument structure showing how the design of an Infusion Pump handles the hazard related to battery exhaustion. The argument structure represents the requirements decomposition and design decisions towards the mitigation of the Battery Exhaustion hazard. It was identified in the hazard analysis as a kind of Underinfusion hazard.

The main design choice to mitigate the hazard is to include a caregiver notification safety requirement in the system. The safety requirement appears as a goal that support the mitigation of the hazard. In order to give additional information and link the high-level goal with evidence items, the argument structure is filled with contextual information and sub-goals refinements.

In Figure 2, the goal **G1** represents the battery exhaustion hazard that must be addressed, and the context **C1** represents the group of hazards to which it belongs. A safety requirement is derived and represented in goal **G2**. The rationale for this requirement is depicted in the assumption **A1**, giving an explanation for this design decision. A low-level requirement is derived and represented in goal **G3**. This requirement is realised by the system module represented in context **C3**. The evidence needed to fulfil the requirement is the project artefacts represented in solutions **Sn1** and **Sn2**. However, the derivation of **G2** to address **G1** results in a new hazard, depicted in context **C2**. So, the fulfilment of **G2** depends on the fulfilment of **G3** and the mitigation of the new hazards derived from it. The requirements **G4** and **G5** are then derived to mitigate the notification hazard. The strategy **St1** is an argument step that provides an explanation of how hazard mitigation will be demonstrated.

How to create and structure ACs is pointed out by researchers and practitioners as an open challenge (LANGARI; MAIBAUM, 2013; NAIR et al., 2014; VARA et al., 2019). When producing assurance arguments, practitioners lack expertise and guidance in constructing the arguments (CHENG et al., 2018; DOSS; KELLY, 2016). For example, they find it difficult to identify what project information should become a claim, an argument or an evidence item and, in turn, difficult to determine the proper assembly of the argument structures (NAIR et al., 2015).

The development of ACs during the Software Development Life Cycle (SDLC) brings benefits beyond the certification needs. Assurance Case Development (ACD) could also enhance stakeholders' and team's awareness of safety and risk management, as the collaborative work toward ACs increase communication of developers with safety experts (STÅLHANE; MYKLEBUST, 2016). Also, developers engaged in the definition and development of ACs foster a more risk-oriented analysis and safety thinking (CICOTTI, 2017; CLELAND-HUANG; VIERHAUSER, 2018). Early ACD also provides support for emergent design that are performed in domains such as autonomous systems (ALEXANDER; KELLY; GORRY, 2010). Finally, the co-development of SCS and its companion AC can help manage incomplete or uncertain requirements (GRAYDON; KNIGHT; STRUNK, 2007).

Traditional SCS approaches perform AC production after completing the implementation and verification of a system (KELLY, 2018). Although ACs require much information that

will only be completely available at the end of the system development process, it is not recommended to start elaborating on them in the late phases of development (DOSS; KELLY, 2016). Practitioners could better approach ACD as an evolutionary activity integrated with the rest of the safety and software lifecycle (KELLY, 2018).

However, the integration of ACD into the SDLC is challenging (MYKLEBUST; HANSSEN; LYNGBY, 2017; NAIR et al., 2015), and some recurring problems are reported in the literature when this integration does not occur:

1. Late rework due to impossibility to build a compelling assurance case (KELLY, 2018);
2. Poor quality (incompleteness or inconsistency) of assurance cases (HAWKINS et al., 2013);
3. Loss of requirements and design rationale (HAWKINS et al., 2013; KELLY, 2018);
4. Late discovery of safety requirements or constraints (CHENG et al., 2018);
5. Lack of integration between safety engineering and software development (LANGARI; MAIBAUM, 2013; MYKLEBUST; HANSSEN; LYNGBY, 2017).

1.2 PROBLEM

Practitioners report that the lack of integration between ACD and SDLC makes it harder to handle changes as well as to discover and organise a large amount of evidence that should be presented in the ACs (CHENG et al., 2018; VARA et al., 2022). The postponement of ACD may lead to increased rework as eliminating or mitigating a late-discovered hazard may be difficult and costly. Besides, requirements, design decisions, and safety evidence that are not accurately recorded, maybe due to process deviation or lack of safety planning, may affect the quality and completeness of documentation sent to certification authorities (HAWKINS et al., 2013).

Practitioners pointed out that continuous development of ACs could foster early discussion of safety concerns, enable the system design to be reviewed from a safety perspective, and bridge various stakeholders' expertise in SCS development (CHENG et al., 2018; DOSS; KELLY, 2016; LANGARI; MAIBAUM, 2013).

ACD and RE are disciplines altogether bound, as RE activities produce the information for argument development, and the building and assessment of the arguments act as a review of

the requirements decisions. Besides, these two disciplines share similarities regarding their life-cycles. First, they should be addressed early and continuously throughout the SDLC. Second, they both need to cope with undeveloped specifications and late refinement/detailing as modern development (even in the safety domain) moves towards agile, incremental approaches. However, we could not find works directly addressing the intersection between RE and ACD. In this thesis, we investigate the intersection of RE and ACD and propose a solution to support their integrated development.

In software engineering, “continuous” approaches refer to the adaptation of processes and activities to be more iterative, flexible and interconnected with other activities of the SDLC (BARCELLOS, 2020). To support the automation of continuous approaches, the project infrastructure would benefit from data repositories and semantic integration to interconnect information produced by the management tools involved (BARCELLOS, 2020).

Some research initiatives are aimed towards model-based approaches to support safety assurance and certification, notably the projects OPENCROSS (VARA; RUIZ; ESPINOZA, 2018) and AMASS (AMASS Consortium, 2023). These two projects resulted in an assurance and certification open tool platform called “OpenCert” (Eclipse Foundation, 2023). Recent studies reported that model-based approaches provide perceived benefits for the developers and organisations (LIEBEL et al., 2018; HULDT; STENIUS, 2019). However, some aspects hinder the adoption of models in RE: lack of customisability of modelling tools, lack of interoperability among multiple tools, and the capabilities to access information stored in models (LIEBEL; TICHY; KNAUSS, 2019). Therefore, organisations willing to incorporate assurance-related capabilities into their processes may find it costly to adopt a model-based solution that does not connect to their (continuous) development environment.

Other works focus on automating the effort to manage the evolution of assurance cases upon changes in the associated project information. Kokaly et al. (2016) provides a semi-automated approach to identify parts of ACs that can be reused or need review upon the release of a new specification. The specification models are elements of class and sequence diagrams and do not include assurance-related information such as justification, assumptions, and mitigation strategy.

In Agrawal et al. (2019), the aim is also to support identifying parts of ACs affected by changes in the system specification. Their specification is based on a traceability information model that can represent development artefacts and their associations with elements (claims, strategies, justifications, assumptions) of an assurance case. Also, their traceability information

model does not encompass assurance-related information, although it includes hazard and safety analysis artefacts.

In these two discussed works, assurance-related information is kept only on the AC specification. We could not find works that promote the management of assurance-related information inside regular project management tools, such as issue tracking systems and product/application lifecycle management (P/ALM) systems.

Modern development methodologies promote iterative and incremental development with comprehensive tool support for processes (ONE, 2019; DIGITAL.AI, 2022). A popular tool in modern development settings is the Issue Tracking System (ITS), which has evolved from simple bug tracking tools to central knowledge repositories for software projects (MERTEN et al., 2015).

In this thesis, we investigate how a single shared repository could be used to register, update and review regular project information, assurance information and their associated traceability. We define regular project information as the kind of the data that permeates any traceability information model for software projects, such as requirements, design, and tests. We define assurance-related information as the kind of data related to the safety design and safety specifications, such as hazards, causes, mitigations, assumptions, and any kind of rationale needed to compose assurance case arguments.

1.3 RESEARCH QUESTIONS AND OBJECTIVES

From the previously described context, the research problem is presented in the following Research Questions (RQ):

RQ1 How does assurance case development interplay with software development?

RQ2 How does assurance case development interplay with requirements engineering?

RQ3 How can assurance information be managed together with project information?

RQ4 How can the integrated work on project and assurance information be supported?

Our main objective is to develop a framework to support safety and software teams in managing assurance information inside regular project management tools, performing automated assessment and assurance case generation. The proposed framework has the following design goals:

- Support the integrated management of assurance and regular project information;
- Support the automated assessment of integrated assurance and regular project information;
- Support the automated generation of assurance case fragments; and
- Support the customisation of the framework for its adoption in different SCS project settings.

In this thesis, our specific objectives consist of the following:

Goal I Understand how assurance case development is performed alongside the development processes (to answer RQ1).

- Characterise the methods of integration and roles involved;
- Identify expected outcomes of the integration; and
- Identify challenges and needs.

Goal II Understand how assurance case development activities currently interplay with requirements activities (to answer RQ2).

- Characterise the current state of practice regarding AC development;
- Identify practices towards the integrated development of AC and requirements;
- Gather the views of practitioners on the integration of RE and ACD processes.

Goal III Define a framework to integrate assurance information into project management tools (to answer RQ3).

- Develop a reference implementation for the framework (to answer RQ3 and RQ4).
- Design and build materials and procedures for a qualitative survey with experts (to answer RQ4).
- Present evaluation of illustrative scenarios and a qualitative survey with experts (to answer RQ4).

The purpose of Goal I is to identify the state of the art in the interplay of ACD and SDLC. The aim of Goal II is to identify the state of the practice in the interplay of RE and ACD, based on the findings of Goal I tasks. Goals III and IV are to propose a solution based on

the findings of Goal I and Goal II tasks. Finally, goals V and VI aim to evaluate the proposed solution.

1.4 RESEARCH METHODOLOGY

1.4.1 Research Classification

The methodological classification of the various research activities involved in this thesis is summarised in Table 1, according to the classification guidelines proposed by Easterbrook et al. (2008).

Table 1 – Methodological framework

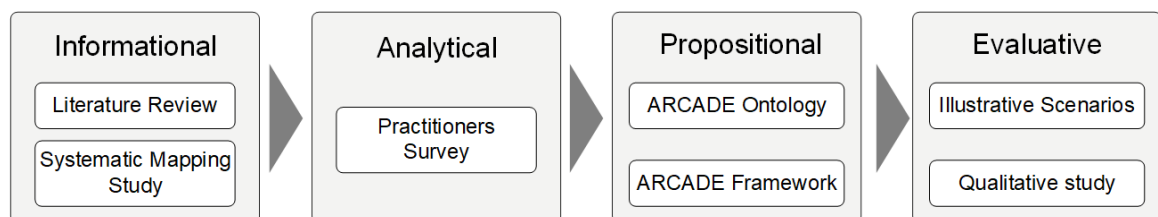
Aspect	Classification
Types of Research Question	Description and Classification, Design
Philosophical stance	Pragmatism
Approach method	Inductive
Nature	Qualitative and Quantitative
Research methods	Systematic mapping study, qualitative survey with practitioners, illustrative scenario, and qualitative survey with experts

Source: The author (2023)

1.4.2 Research Steps

Our methodology follows the phases proposed by Glass (1995): Informational, Analytical, Propositional, and Evaluative. Figure 3 presents the methodology overview.

Figure 3 – Thesis methodology steps.



Source: The author (2023)

The Informational Phase encompasses the literature review and mapping study. Initially, an preliminary literature review was performed and we observed a lack of studies in the intersec-

tion between ACD and software development. Then, we planned and conducted a Systematic Mapping Study to identify and analyse the existing approaches for incremental assurance case development. In the Analytical Phase, we investigated the state of the practice through a practitioners' survey to understand how requirements engineering and ACD activities currently interplay. In the Propositional Phase, we designed and implemented the ARCADE framework. Finally, in the Evaluative Phase we performed evaluation studies regarding the ARCADE framework.

1.5 SUMMARY OF PUBLICATIONS

In this section, we list papers and book chapter published that are related to this thesis.

1. ALMENDRA, C. C.; BARROS, F.; SILVA, C. Using assurance cases in requirements engineering for safety-critical systems. In: *Anais Estendidos da X Conferência Brasileira de Software: Teoria e Prática*.
2. ALMENDRA, C.; SILVA, C. Managing assurance information: A solution based on issue tracking systems. In: *Proc. of the 34th Brazilian Symposium on Software Engineering*. 2020. p. 580–585.
3. ALMENDRA, C.; SILVA, C.; VILELA, J. Incremental Development of Safety Cases: a Mapping Study. In: *Proc. of the 34th Brazilian Symposium on Software Engineering*. 2020. p. 538–547.
4. ALMENDRA, C.; SILVA, C.. The Role of Requirements Engineering in Safety Cases. In: *Requirements Engineering for Safety-Critical Systems (book)*. 2022. p. 145–154
5. ALMENDRA, C.; SILVA, C.; MARTINS, L. E. G.; MARQUES, J. How assurance case development and requirements engineering interplay: A study with practitioners. *Requirements Engineering*, Springer-Verlag, v. 27, n. 2, p. 273–292, Jun 2022.
6. ALMENDRA, C.; SILVA, C. ARCADE: a framework for integrated management of safety assurance information. In: *IEEE 31st International Requirements Engineering Conference (RE)*. 2023.

We also have started planning and writing a paper on the ARCADE framework with regard to its design (Chapter 5), implementation (Chapter 6) and evaluation (Chapter 7). It will be an extended version of the paper published in the RE Conference 2023.

1.6 THESIS ORGANISATION

This thesis is organised into eight chapters, which include this introductory chapter and the chapters described below:

- Chapter 2: We discuss the background concepts about Assurance Case Development, Requirements Engineering for safety-critical systems, and Semantic Web technologies.
- Chapter 3: We present the mapping study and discuss its results.
- Chapter 4: We present the practitioners survey and discuss its results.
- Chapter 5: We present the design goals of the ARCADE framework and the development of the Safety Assurance Ontology.
- Chapter 6: We present the design and implementation of the ARCADE framework tool.
- Chapter 7: We discuss the ARCADE framework evaluation approaches.
- Chapter 8: We present final considerations, a summary of contributions, limitations and a discussion on future works.

2 BACKGROUND AND RELATED WORKS

In this chapter, we discuss the theoretical background that supports this research. Section 2.1 introduces Assurance Case Development (ACD). Section 2.2 introduces Requirements Engineering (RE) for Safety-Critical System (SCS). Section 2.3 introduces Semantic Web concepts. Section 2.4 presents the previous research related to our work. Finally, in Section 2.5, a chapter synthesis is shown.

2.1 ASSURANCE CASE DEVELOPMENT

Regulations for critical systems are built and evolved towards a body of knowledge gathered from domain experts and the systems' field operation. Traditionally, regulations for Safety-Critical System (SCS) followed only prescriptive approaches. They comprised obligatory development practices and explicitly required the submission of documentation that communicates how process and product evidence demonstrate the safety of the software functions (HAWKINS et al., 2013).

Evidence for safety certification refers to information that contributes to build confidence in the safety properties of a system and to satisfy regulatory requirements (NAIR et al., 2014). Thus, evidence management comprises the collection, structuring, and assessment of safety information to address certification needs (NAIR et al., 2014).

However, software-intensive systems are increasingly complex and evolving rapidly, so the standards cannot cope with the incorporation of novelties as fast as the regulated domains evolve (HATCLIFF et al., 2014). Even within the same domain, the requirements, design, and implementation of systems may vary greatly, so their safety requires individual consideration. Standards are moving towards a goal-based approach, in which developers need to address high-level objectives and explicitly present an argument which communicates how the safety of the software functions was achieved (HAWKINS et al., 2013).

Assurance Case (AC) is a trending approach to communicate and argue over the safety of a system, offering more flexibility to justify requirements and design decisions (BLOOMFIELD; BISHOP, 2010). The purpose of an AC is to present a case in favour of a system. ISO/IEC 15026-2 states that *"an assurance case includes a top-level claim for a property of a system or product (or set of claims), systematic argumentation regarding this claim, and the evidence*

and explicit assumptions that underlie this argumentation" (ISO/IEC/IEEE..., 2022). In other words, an assurance case should communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context (KELLY, 1999). Assurance cases were born as "*safety cases*" or "*safety reports*". In this thesis, we use the term Assurance Cases, even when referring to Safety Assurance Cases.

Nowadays, some regulations mandate or recommend elaborating an AC as part of the certification documentation (RINEHART; KNIGHT; ROWANHILL, 2017). For example, Defence Standard 00-56:2017 regulates safety management requirements for all contracts of the UK government related to defence systems (DEFENCE..., 2017). CENELEC EN 50129:2018 covers safety requirements for railway communication, signalling and processing systems (CENELEC..., 2018). ISO 26262:2018 covers requirements for the management of functional safety of road vehicles (ISO..., 2018). All these standards require ACs as part of the certification documentation. The FDA provided a guideline for vendors of infusion pumps recommending that new submissions of new or updated products include an AC to demonstrate their safety (FDA..., 2014). More recently, a new technical specification is under development to guide the production of ACs for health software (ISO/AWI..., 2021). Indeed, the idea of developing assurance cases to demonstrate a system's fitness to use has spread to other concerns such as security and dependability (RINEHART; KNIGHT; ROWANHILL, 2015).

2.1.1 Problems Faced by Practitioners

ACs comprise an amalgamation of information from legal requirements, product requirements, risk and hazard analysis results, design decisions and rationale, testing results, and process management records (BLOOMFIELD; BISHOP, 2010). A complete AC report requires information that will only be available at the development's end. Still, it is not recommended to start elaborating on them in the late phases (DOSS; KELLY, 2016; KELLY, 2018). Even so, it is still a current practice (MYKLEBUST; HANSSEN; LYNGBY, 2017; ALMENDRA et al., 2022).

P1. Late rework due to the impossibility of building a compelling assurance case. Developers may not have accurately recorded many decisions and safety evidence, maybe because of process deviation or lack of safety planning (HAWKINS et al., 2013; KELLY, 2018).

P2. Poor quality (incompleteness or inconsistency) of assurance cases. Late development induces teams to adopt a confirmatory argumentation over a given design, with limited space for exploring alternatives (HATCLIFF et al., 2014). Also, late development may

impair the engagement of external reviewers during the development process (HAWKINS et al., 2013).

P3. Loss of requirements and design rationale. It could be hard to recall all the evidence in a long-term project. So, if not properly recorded, developers tend to lose decision-making rationale as the project schedule takes months and with changing of staff team members (LANGARI; MAIBAUM, 2013). Some challenges pointed out as of high importance by practitioners were the following (NAIR et al., 2015): (i) the support for argumentation when linking evidence to requirements/objectives from standards; (ii) how to determine which information should become evidence; and (iii) how to effectively build ACs.

P4. Late discovery of safety requirements or constraints. The postponement of AC development to later stages may result in late identification of safety requirements or constraints (CHENG et al., 2018). This could lead to increased safety and project management risks caused by late-discovered safety issues (e.g., a hazard may not be adequately eliminated or mitigated by design) (HATCLIFF et al., 2014).

P5. Lack of integration between safety engineering and software development. The gap between safety and software disciplines has been reported for decades (LUTZ, 2000; HEIMDAHL, 2007; VILELA et al., 2017a); such separation is also observed in the ACD domain (LANGARI; MAIBAUM, 2013; MYKLEBUST; HANSSEN; LYNGBY, 2017). The integration of AC development into systems development processes is indicated as a way to leverage the integration of safety and software engineering (WILSON; KELLY; MCDERMID, 1997; KELLY, 1999). However, such integration is still challenging (HATCLIFF et al., 2014; DOSS; KELLY, 2016).

2.1.2 Assurance Case Structure

Seminal approaches to present AC reports have comprised hundreds of pages of unstructured text. The need for a more formal structure led to the development of textual and graphical notations towards the concept of argument structures (BLOOMFIELD; BISHOP, 2010). These structures present and interrelate key information about a system's design (KELLY, 2018):

- Requirements and goals of the system;
- Evidence items that support the satisfaction of requirements and goals;
- Arguments that explain and justify how evidence items relate to the requirements and goals.

The structure of an AC follow a hierarchy comprising three kind of elements (RUSHBY, 2015):

- *Claims*: Statements representing high-level goals, requirements, constraints or expected properties of the system under concern;
- *Arguments*: Statements elaborated to demonstrate that the *Claims* were satisfied
- *Evidence*: Any piece of information generated during design, implementation and verification activities that substantiates a *Claim* or an *Argument*.

The provision of evidence to back up claims is the fundamental objective of assurance argument development. However, just tracing a top claim to pieces of evidence is not enough to argue that a claim is assured. It requires an explanation, a rationale to explain why the pieces of evidence together confirm a claim. 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 that go from top claims down to evidence items (Figure 1).

Two seminal graphical argumentation notations are the Claims-Arguments-Evidence (CAE) (ADELARD, 2023), and the Goal Structuring Notation (GSN) (GOAL..., 2021), respectively developed at City University (UK) and York University (UK) (BLOOMFIELD; BISHOP, 2010).

GSN is the most popular visual notation to specify ACs (NAIR et al., 2014; ALMENDRA et al., 2022). Figure 2 presents a partial example of an AC using the GSN notation, adapted from (WEINSTOCK; GOODENOUGH, 2009). The example depicts a safety argument structure showing how the design of an infusion pump handles the hazard related to battery exhaustion. An infusion pump is a critical system that delivers fluids into a patient's circulatory system intending to provide controlled delivery of medicine (WEINSTOCK; GOODENOUGH, 2009). Appendix A presents the notation and extended example of an AC using GSN.

A comparison among GSN and other AC visual notations and textual templates can be found in (RUSHBY, 2015). Some studies reported that industry practice is inclined towards adopting textual templates and structured text (NAIR et al., 2015; HOLLOWAY, 2008). An analysis of textual structure alternatives for AC specification can be found in (HOLLOWAY, 2008). Also, in Rao (2019) is presented a comparative analysis of various AC graphics-based notations like Goal Structuring Notation (GSN), Claims Arguments and Evidence (CAE),

Structured Assurance Case Metamodel (SACM), NOR-STA Services, Resolute and Dynamic Safety Cases.

SACM (Structure Assurance Case Metamodel) (OMG, 2021) is a standard developed by OMG (Object Management Group) that defines a metamodel for representing structured assurance cases. The standard aims to support model-based approaches for SAC management, to allow proper interchange of information of structured arguments among tools (VARA, 2014). GSN and CAE can be mapped to SACM. This metamodel is relatively recent and not yet often reported in state-of-practice studies (NAIR et al., 2015; ALMENDRA et al., 2022).

It is worth noting that the standard ISO/IEC/IEEE 15026-2 (one of the first general purpose regulations towards AC) does not demand the use of any particular notation, and acknowledges that different parts of an assurance case may require different notations on a case-by-case basis (ISO/IEC/IEEE..., 2022).

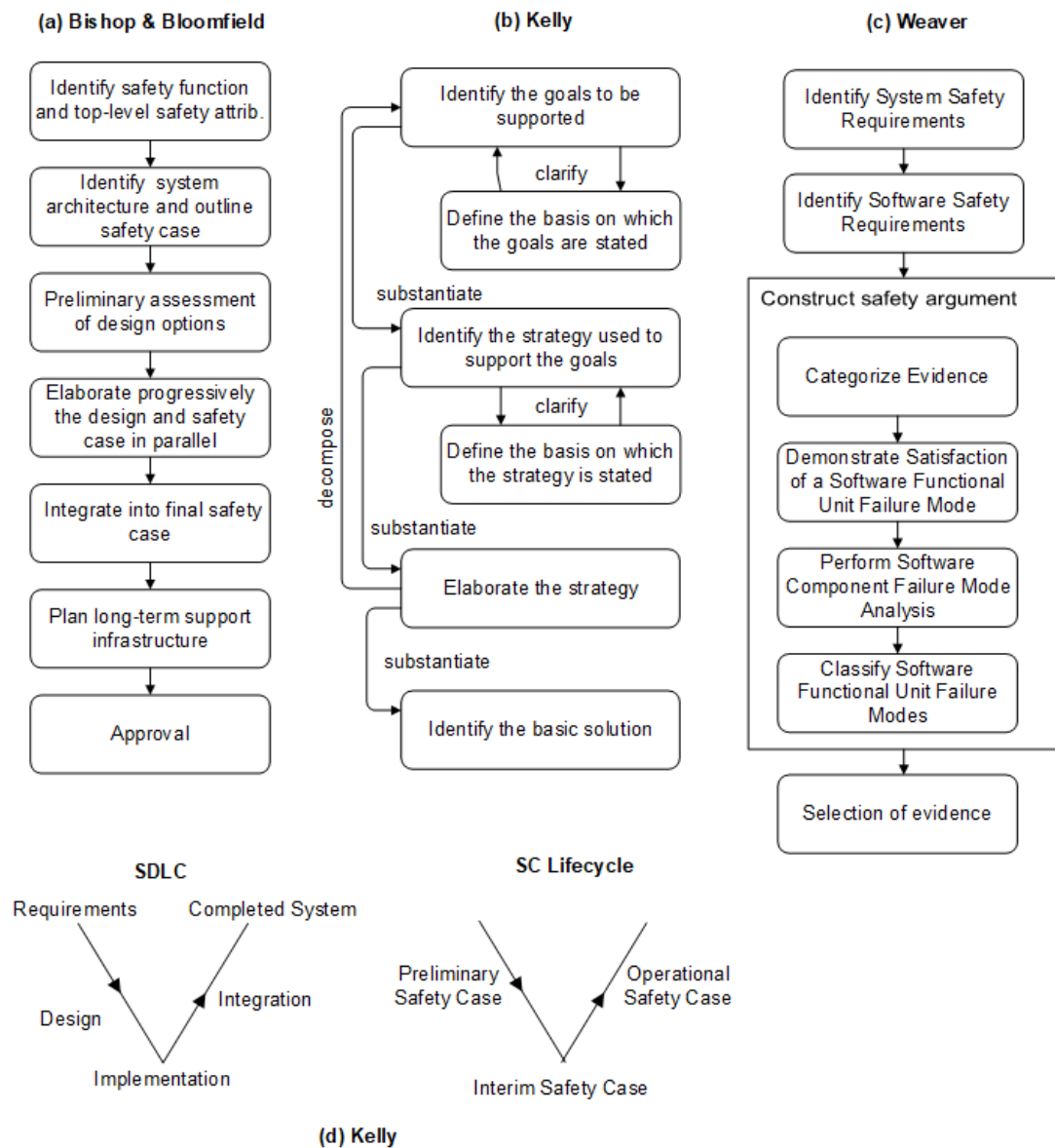
In summary, the central spine of safety arguments are requirements, hazards, design, and the rationale to justify the decisions. Certifiers, regulators and other stakeholders need to build trust that the provided evidence mitigates the hazards and satisfies systems goals. The intermediary arguments and rationale facilitate the review of project decisions. Thus, AC construction demands the explicit statement of assumptions, justifications and argumentation strategies (KELLY, 2018).

2.1.3 Argument Development

In this section, we present three seminal approaches for assurance case development: the Safety Case Methodology by Bishop and Bloomfield (2000), the Six-Step goal structure method by Kelly (1999) and the Conceptual Framework and Patterns Catalogue by Weaver (2003). Although these approaches recommend the integration of AC development into system development approaches, they proposed processes for AC development without discussion of how the integration with Software Development Life Cycle (SDLC) could occur. We briefly discuss these approaches below, highlighting their seminal contribution (Figure 4).

The work of Bishop and Bloomfield (2000) was in the context of developing the AS-CAD (Adelard Safety Case Development) manual, which is known as CAE (Claim-Argument-Evidence) notation. Their methodology includes guidance for classifying types of claims, types of argument, possible sources of evidence and strategy for layering ACs (Figure 4–a). Since the beginning of research and practice towards assurance cases, there was a recommendation for

Figure 4 – Assurance case construction methodologies



Sources: (a) BISHOP; BLOOMFIELD (2000), (b and d) KELLY (1999), (c) WEAVER (2003)

integration with system/software design and development processes. As safety assessment is a significant development factor, developers should consider the cost and feasibility of AC development when planning the development of a safety-critical system (BISHOP; BLOOMFIELD, 2000).

The work of Kelly (1999) comprises a methodology and a graphical notation for the development of AC arguments (KELLY, 1999). His work focuses on four main problems towards AC development: how to present clear safety arguments, how to support incremental development, how to manage them throughout their lifecycle, and how to support reuse. Kelly proposed a methodology named “Six-Step Method for Developing Goal Structures” (Figure 4–b) and ex-

tensions for GSN to enhance the notation support for abstraction, modularity and pattern documentation. His work also presented a three-stage evolutionary AC development lifecycle, which he mapped into a V-model (Figure 4–d). Preliminary safety cases are created based on high-level product requirements, standards requirements and constraints, and preliminary risk and hazard analysis. Thus, this AC stage aligns with the Requirements and Design stages of the SDLC. Interim safety cases are detailed refinements of preliminary cases incorporated with evidence items from the design and implementation activities (the assurance argument presented in Figure 2 represents an interim AC). They align with the Implementation stage of SDLC, when the team makes many low-level design and implementation decisions, thus generating much evidence information. In the last stage of the cycle, following the Integration and Testing stage of the SDLC, testing evidence items fill in the Operational safety case.

The work of Weaver (2003) proposed a conceptual framework that systematizes the selection of evidence required for building software safety arguments. The work proposed a method (Figure 4–c) and a set of AC patterns described in GSN, which serves as the basis to build arguments using the framework. He also defines an approach for determining the level of confidence in arguments and pieces of evidence, suggesting that how much confidence we can put in them is as important as the strategies used to decompose top goals and claims.

All these seminal approaches provided concepts, notations, and guidelines for AC development practices. However, how to create and structure ACs is still pointed out by researchers and practitioners as an open challenge (LANGARI; MAIBAUM, 2013; NAIR et al., 2014; NAIR et al., 2015; CHENG et al., 2018; VARA et al., 2019).

Some initiatives had delivered processes and tools for automating safety evidence management using model-based approaches (VARA; RUIZ; ESPINOZA, 2018). The OpenCert platform is an open-source integrated, holistic solution for assurance and certification management (Eclipse Foundation, 2023). It targets cyber-physical systems that belong to various safety-critical domains. The aim of the platform is to lower certification costs. Another initiative was the AMASS project, which continued the development and consolidation of OpenCert platform (AMASS Consortium, 2023). Its purpose is to provide an architecture-driven, multi-concern assurance, reuse-oriented and seamlessly interoperable tool platform (VARA et al., 2022).

All components and tools from these two projects were integrated into the Eclipse OpenCert bundle (Eclipse Foundation, 2023). The set of tools provides many features for model-based development with a focus on multi-concern assurance and certification. It is also possible to import project information data from other artefacts. However, OpenCert is designed to be the pri-

mary tool where people from different teams (requirements, architecture, safety) should work. The integrated data can then be reused and cross-referenced through the various modelling tools bundled together, including an assurance case editor (AMASS Consortium, 2018). Although many research projects aim to develop solutions for model-based assurance, OpenCert's use in industry is not yet consolidated (VARA; RUIZ; ESPINOZA, 2018; ALMENDRA et al., 2022).

There are some approaches in the literature that propose the use of formal methods in the development of ACs (GRAYDON, 2015; CÂRLAN; BEYENE; RUESS, 2016; MENG et al., 2020). These approaches are focused on the specification of the arguments and the use of formal models to assess consistency, confidence and other quality attributes (GRAYDON, 2015). In our work, we focus firstly on the early step of organising, collecting and structuring the project information before the application of any assessment technique. Then, our work also proposes an ontological approach to assess the quality of the information. It is important to perceive the ACD process as a pipeline of data management, in which diverse techniques could be assembled, according to specific needs of the organisation or regulated domain.

2.1.4 Assurance Evidence Management

What should compose an AC and how it should be structured varies depending on the target standards and AC notations used. This variation poses difficulties in reusing, within the same company, the know-how and assets across products targeting different domains or multiple domains at once (HATCLIFF et al., 2014; RODRÍGUEZ-DAPENA, 1999).

A recent survey catalogued a taxonomy composed of 49 different basic evidence types found in the literature (NAIR et al., 2014). As an illustration, Table 2 brings some examples of the variety of output evidence types sourced throughout a typical software development process. We selected the most discussed types of evidence from Nair et al. (2014) and related them to the corresponding source activities in the SDLC. The aim of this table is not to be exhaustive, but rather to illustrate how the evidence generation spreads along the development lifecycle.

As we discussed above, there is a great variety of information that need to be collected and structured to produce ACs. Also, these different kinds of data are often managed separately. For example, the Hazard and Risk Analysis (HARA) aims to determine the hazards, hazardous events, and hazardous situations related to the system under assessment (TIUSANEN, 2018). It shall cover all modes of operation, in all reasonably foreseeable circumstances, to determine

Table 2 – Mapping of evidence types and source processes

Most cited evidence types (NAIR et al., 2014)	Source process
Hazard causes specification	Safety analysis
Risk analysis results	Risk management
Hazard specification	Safety analysis
Accidents specification	Safety analysis
Requirements specification	Requirements engineering
Hazards mitigation specification	Design
Design specification	Design
Review results	Verification and Validation
Structural coverage testing results	Verification and Validation
System historical service data specification	Operation monitoring

Source: The author (2023)

the sequences of events leading to hazardous events. The output of HARA serves as input to system engineering activities, and as the system design evolves the HARA should be revised to check if any design decisions may itself be a cause of new hazards.

In this thesis, we investigated how the assurance information related to safety requirements and design decisions could be integrated into the tools that already store regular project information. As the term is used here, assurance rationale is the reasoning behind requirements, design, testing, implementation, and mitigation decisions that are related to safety.

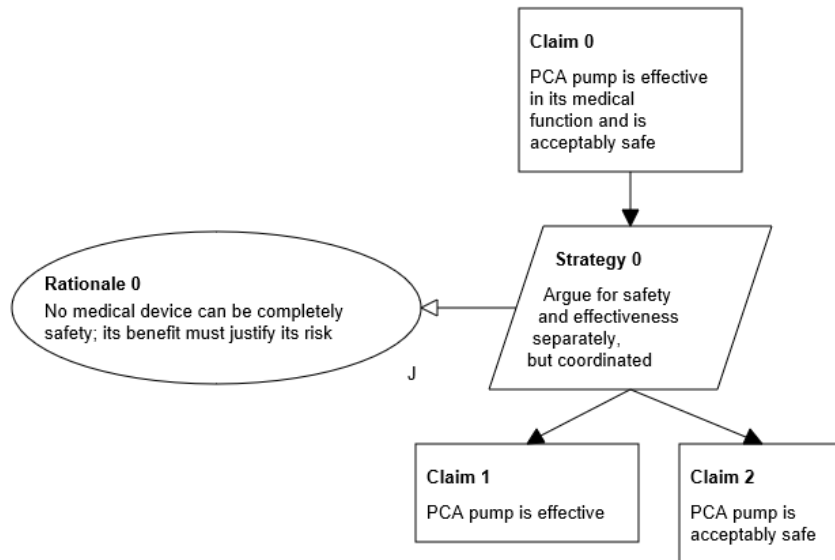
Rationale information represents the reasoning underlying the production of artefacts in software engineering activities (BURGE et al., 2008). The research on rationale-based software engineering investigates the methods and technologies that support decision-makers in explicitly registering this reasoning (BURGE et al., 2008). Design rationale is one of the best-known subjects of rationale research, and it deals with the reasoning within the design and architecture process. Requirements rationale is also an important field of research; one of the best-known approaches for requirements rationale management is goal-oriented requirement engineering (VILELA et al., 2017b).

We now illustrate two examples of assurance information items in the context of a safety-critical project using the Open PCA Pump dataset (HATLICFF, 2019), which is explained in Appendix C. The dataset contains a requirements specification (LARSON; HATLICFF, 2018) and an assurance case report (LARSON, 2014). The following excerpt illustrates key assurance information that is commonly only recorded in the AC artefacts.

In Figure 5, the root element states the approach to satisfy the main *Claim 0*: “*Strategy*

0 – *Argue for safety and effectiveness separately, but coordinated*". This strategy is a piece of assurance information, not stated in any single requirement or design item. However, it is an important project decision only stated in the AC artefact.

Figure 5 – Argument example with rationale information

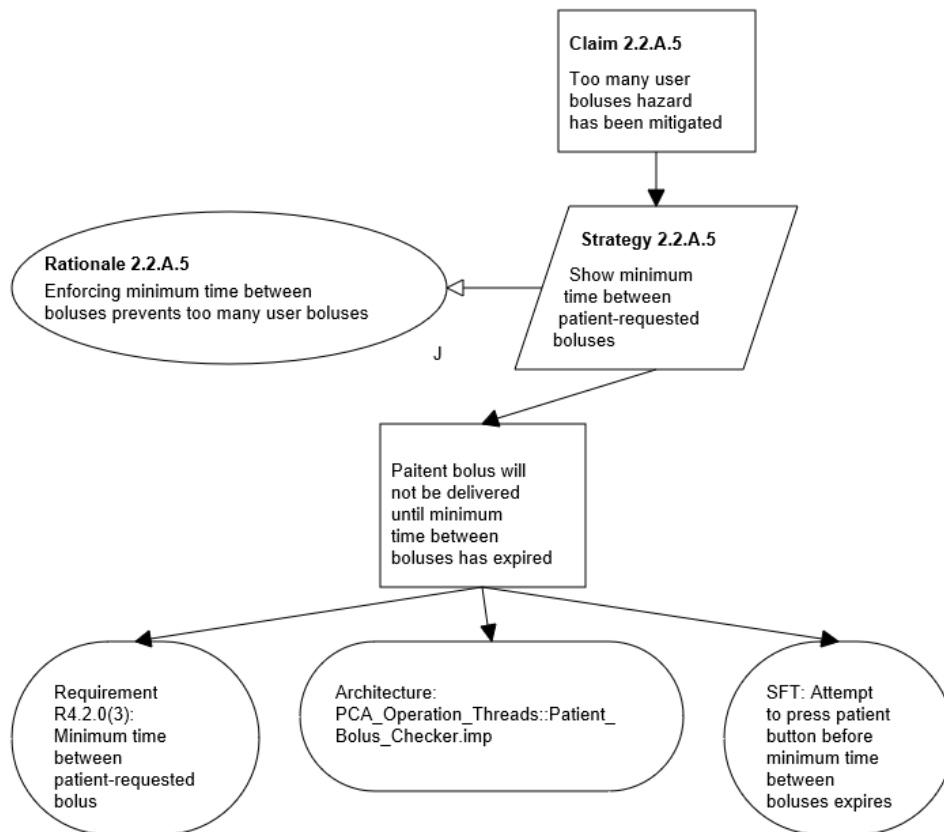


Source: Larson (2014)

In Figure 6, we see an example of rationale items supporting the explanation of the reasoning behind hazard mitigations or eliminations. The items "*Rationale 2.2.A.5*" and "*Strategy 2.2.A.5*" explain the argument behind why the single requirement "*Patient bolus will not be delivered until minimum time between boluses has expired*" is sufficient to mitigate the hazard "Too many user boluses". The requirements specification provided in the dataset does not state any hazard explicitly, and only in the AC report is the relationship between requirements and hazard described (LARSON; HATLICFF, 2018).

In the appendix, we explore more of the relationship between assurance case elements and project information (Appendix C). The examples of assurance information provided above show the very purpose of ACs: to explicitly state how the system developers have dealt with safety issues. These rationales bind requirements, design, test, and process decisions, and project management tools usually do not record them explicitly.

Figure 6 – Another argument example with rationale information



Source: Larson (2014)

2.2 REQUIREMENTS ENGINEERING FOR SAFETY-CRITICAL SYSTEMS

Requirements Engineering (RE) is a crucial process for the production of quality systems, even more in the context of software-intensive SCSs (SIKORA; TENBERGEN; POHL, 2012; MARTINS; GORSCHKE, 2016; VILELA et al., 2017a). As a knowledge-intensive work, RE processes start from scattered data in multiple sources (stakeholders, regulations, existing products, team expertise, and others) and end up in structured specifications and identified decisions, understood and negotiated among stakeholders. RE processes for critical systems need to account for safety engineering concerns, as requirements changes and safety analyses should be continuously harmonised through the development (SIKORA; TENBERGEN; POHL, 2012).

There is extensive research on the specification and analysis of requirements for safety-critical systems. However, there still are gaps when tracing safety requirements to testing procedures and safety analysis information (LUTZ, 2000; CLELAND-HUANG et al., 2014). As modern development processes rely more and more on exploratory or iterative-incremental approaches, there is a lack of mechanisms to disseminate and integrate knowledge about new

or derived safety requirements throughout the software development lifecycle (LUTZ, 2000; VILELA et al., 2017a).

As a knowledge-intensive work, requirements elicitation and analysis go from scattered data in multiple sources (human or not) to structured information and identified decisions, understood and negotiated among stakeholders (WIEGERS; BEATTY, 2013). 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 artefacts is known as the rationale (BURGE et al., 2008). In the context of assurance case development, rationale is the essence of arguments. It provides the confidence to assure that requirements decomposition and design decisions reasonably address all identified risks and hazards.

2.2.1 Requirements Traceability

Vilela et al. (2018) provide the Safe-RE metamodel that aims to support the specification of safety-related concepts for various industry standards. It provides concepts and relationships to model safety requirements with much detail. It can be used to design a complete model-driven approach for safety requirements specification and model transformation for the generation of ACs. For this thesis, we used the metamodel partially to achieve a lightweight approach. The aim is to provide extended semantics for the project management items and their relationships.

To meet safety assurance goals, traceability must be supported. Otherwise, several problems related to traceability could delay the certification of ACs products, such as lack of a traceability information model, uneven granularity of tracing links, and missing or redundant links (MÄDER et al., 2013). The information needed to build a compelling assurance case can be compared to a high-end usage of requirements traceability (RAMESH; JARKE, 2001). SCS demands the application of traceability for full lifecycle coverage, capture of decisions, rationale, and tracing across product and process dimensions, and requirements decomposition (RAMESH; JARKE, 2001). However, practitioners do not explicitly indicate traceability as a significant concern in requirement engineering for SCS (MARTINS; GORSCHKE, 2016; VILELA et al., 2017a).

Certification of a system requires evidence and information from all development disciplines. Assurance case development requires that diverse pieces of information be logically linked together to elaborate compelling arguments. As Cleland-Huang et al. (2014) found, practitioners who do not work in regulated domains tend to be unfamiliar or non-believers

of software traceability traits. This is another discipline to be promoted, as it is essential to support a cost-effective assurance case development.

Safety is a non-functional requirement and a cross-cutting concern directly related to system components and behaviour. Exploring and registering rationale for non-functional requirements often involves modelling alternatives to assist comparative analysis (BURGE et al., 2008). Current assurance case templates and specification languages provide support only to express the chosen requirements and design solutions, as well as the arguments for their positive contribution to safety. However, a compelling argumentation favouring a selected design is often backed by demonstrating its advantages compared with other alternatives. The management of rationale and the mechanisms to translate it into arguments that are part of an assurance case are key activities.

We also find approaches in the literature aiming to promote interoperability among various tools. For example, the Open Dependability Exchange (ODE) metamodel supports the representation of safety characteristics of a system in a comprehensible and transparent manner on different levels of abstraction (ZELLER et al., 2023). Its main purpose is to support the exchange of safety information between tools throughout the phases of the SDLC. It is a recent technology, but that has been applied in automotive, healthcare, and railway domains through real use cases

Assurance cases naturally replicate information recorded elsewhere in the project management tools or artefacts. Such duplication of information may increase cost and inconsistencies as the project scales. It is essential to provide means for a partially or fully automated composition of cases from application lifecycle management systems. Such management systems may need to be enhanced with safety-related information models (VARA; RUIZ; ESPINOZA, 2018). These information models may support a full model-based approach in which a single tool concentrates all data (e.g., OpenCert (Eclipse Foundation, 2023)), or more lightweight approaches that retrieve project data and perform specific computations. In the next section, we discuss the issue tracker systems, a category of project management tool largely used to support modern development processes.

2.2.2 Issue Tracking Systems

In recent years, agile and distributed development has stood out as cost-effective approaches, even in regulated environments (DIEBOLD; THEOBALD, 2018). Lightweight develop-

ment processes usually rely on collaborative tools to manage software project lifecycles. Many Application Lifecycle Management (ALM) tools are built on top of an Issue Tracking System (ITS) infrastructure or include it as part of their toolchains, such as Atlassian Jira and Azure DevOps. The use of ITSs has been reported in SCS development studies such as (HANSSEN et al., 2016).

Nowadays, ITS provides a flexible and lightweight platform in which project information can be managed (PAECH; DELATER; HESSE, 2014; MERTEN et al., 2015). User stories, a format for agile requirements, are often managed using tools based on ITS or collaborative workspaces (DIMITRIJEVIĆ; JOVANOVIĆ; DEVEDŽIĆ, 2015). Worth noting is that these tools handle RE concepts in very different ways, and practitioners invest significant effort in choosing and adapting the tools to their needs (DIMITRIJEVIĆ; JOVANOVIĆ; DEVEDŽIĆ, 2015). An ITS typically provides issues, fields, links and other configurations to organise project information. How these elements are used to manage project information may vary greatly and evolve according to the culture of teams and companies (MERTEN et al., 2015; DIEBOLD; THEOBALD, 2018). These tools commonly store requirements, architecture design, tests, code, and traceability information. By enriching them with assurance information, we avoid the need to keep separate artefacts and provide better support for automatic classification, analysis, and generation of assurance cases.

2.3 SEMANTIC WEB

Requirements engineering (RE) activities can be improved using ontologies to reduce ambiguity, inconsistency and incompleteness (DERMEVAL et al., 2016). Ontology is an explicit specification of a shared conceptualisation that holds in a particular context (SCHREIBER, 2008). It aims to organise domain knowledge over individuals, objects, types, attributes, relationships and functions. Once organised in a knowledge base, this information can be easily shared among people involved in the context of the ontology, such as users, customers, requirements analysts, architects and any other personnel involved in Software Engineering.

Two recent literature reviews bring interesting information on using ontologies in RE. Dermeval et al. (2016) identified that most studies target the reduction of ambiguity, inconsistency and incompleteness. In second place, there were studies purposing requirements maintenance and evolution, and, in third, studies focusing in domain knowledge representation. Valaski, Reinehr and Malucelli (2016) explored the functions of ontologies in RE. The review found

that the intended functions are well distributed in categories: i) Structuring and recovery of knowledge, ii) Verification and validation, iii) Support to understand/identify concepts, iv) Control/share of vocabulary, and v) Integration and transformation models. In the context of requirements modelling, ontologies are an effective approach to validate the consistency of models against a modelling language metamodel and formation rules.

Project management tools should be perceived as knowledge-based systems, especially with regard to requirements traceability information. Ontology-based approaches have been explored in the context of RE for the following (VALASKI; REINEHR; MALUCELLI, 2016):

- Support the structuring and recovery of knowledge;
- Verification and validation;
- Support to understand/identify concepts;
- Control/share of vocabulary; and
- Integration and transformation models.

In the context of our work, an ontology-based approach could bring the following benefits (NOY; MCGUINNESS et al., 2001):

- Make domain premises explicit: safety and software teams are implicitly considering domain premises while carrying out their activities, so an ontology can foster the exchange of knowledge among them.
- Analyse domain knowledge to compute implicit knowledge by invoking reasoning: the interrelation between safety assurance and regular project items may not be trivial or easy to handle adequately on a large scale, so the use of an ontology can support the identification and gathering of data to develop the ACs.
- Enable reuse of domain knowledge: SCS systems tend to be developed on top of other systems and components, so the ontology can be used to interrelate information extracted from different management tools.

2.3.1 Description Logic

Ontologies may be implemented and operationalised in many ways. We used Description Logic (DL) as our logic system in our work. It is sufficiently expressive to represent project

information, and the core reasoning for DLs is mostly decidable (i.e. reasoner will not run forever) (W3C, 2023a). Also, efficient reasoner tools are available (W3C, 2023a), so we can rely on this technology to build predictable tools for general-purpose uses. The Ontology Web Language (OWL) is one of the most common languages used for ontologies related to software engineering (DERMEVAL et al., 2016) and is compatible with DL. The use of an ontology implemented on top of OWL technology benefits from three main capabilities: consistency checking, automatic classification, and ease of extension/tailoring.

A description logic models the relationships between entities in a domain of interest, with three kinds of entities: classes, properties, and individuals (KRÖTZSCH; SIMANCIK; HORROCKS, 2012). Classes represent the categorisation of individuals, properties represent relations between individuals, and individuals represent single entities in the domain. A DL ontology comprises a set of axioms that will be checked together for validity. If one axiom contradicts another, then there is a conflict in the axioms. If there is no conflict, the set of axioms is valid.

We call Terminological Axioms (TBox) those that describe relations between two classes, between a class and a property, and between two properties (KRÖTZSCH; SIMANCIK; HORROCKS, 2012). These axioms represent the main design of a DL ontology, as they describe fundamental relations that individuals must satisfy. Some examples of terminological definitions that TBox axioms can represent are the following:

- T1. Requirement** is a class of individuals;
- T2. Hazard** is a class of individuals;
- T3. Cause** is a class of individuals;
- T4. Safety Requirement** is a sub-class of **Requirement**;
- T5.** *mitigates* is an object property (relation);
- T6.** The source of *mitigates* is **Safety Requirement**;
- T7.** The target of *mitigates* is **Hazard** or **Cause**;
- T8. Requirement** and **Hazard** are mutually distinct (individuals cannot be of both classes).

The Assertional Axioms (ABox) represent knowledge about the individuals by supporting the definition of classes to which they belong (class assertions) and the definition of relations to each other (property assertions) (KRÖTZSCH; SIMANCIK; HORROCKS, 2012). Examples of facts that can be represented using ABox axioms are the following:

A1. Req1 is a **Safety Requirement**

A2. Haz1 is a **Hazard**

A3. Haz2 is a **Hazard**

A4. Req1 *mitigates* Haz1

A5. Req2 is an individual

A6. Req2 *mitigates* Haz2

The axioms stated above support some consistency checking. Together, these axioms are valid, and there is no conflict. Now let us include a new axiom:

A7 *Haz2 mitigates Haz1*

This new axiom **A7** causes the classification of Haz2 to be a **Safety Requirement** (because of axiom **T6**). Haz2 already has the axiom **A2** stating that it is a **Hazard**. However, the axiom **T8** states that these two types of class are different (disjoint). A conflict arises, and the set of axioms is not valid anymore. Which is correct? Is Haz2 really a **Hazard**? Or, was it erroneously linked to a Haz1? The ontology helps the analysis by performing this kind of consistency checking.

Another feature of DL ontology is classification. In the same example above, let us consider the ABox axioms from **A1** to **A6**. As there is a mitigation link between Req2 and Haz2, Req2 will be classified as **Safety Requirement**. So, even without the type information of Req2, its classification can be inferred by the relationships to other concepts.

Indeed, many attributes of the individuals can be inferred by the association between them and cross-checked against pre-defined classifications and categorisations. As an additional example, consider a common relationship between two requirements: refinement. If a requirement *refines* a safety requirement, we can define that this individual is of type **Safety Requirement**. This example of classification serves to standardize the class hierarchy.

Furthermore, classification can be used to connect the domains of Requirements and Assurance Cases. One of the challenges in Assurance Case Development (ACD) is to collect, select, and organise the information to be used in the assurance arguments. Some examples follow:

- Safety Requirement is a Goal

- Hazard is a Goal
- A mitigation link between Requirement and Hazard represents a *Is Supported By* relation
- A refinement link between requirements represents a *Is Supported By* relation
- A component is a Context
- Test case is a Solution/Evidence
- Test result is a Solution/Evidence

In other cases, the classification of project items may lead to more than one concept. For example, a Preliminary Hazard Analysis (PHA) report can be a Solution/Evidence to demonstrate that appropriate safety analysis has been carried out. Otherwise, the same PHA report can also be a Context, indicating the source of hazards in the argument.

The ease of extension and tailoring is another benefit of OWL ontologies. TBox axioms are easy to manipulate, and there are very robust tools to support the design process of an ontology. If there is a need for a new concept, role or property, the ontology designer needs only to include new TBox axioms. If there are existent TBox axioms that no longer fit the specific purposes of the project, the designer may redesign and update the TBox axioms set.

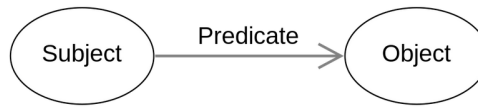
In both cases, DL reasoners automatically check the consistency of new/updated TBox axioms among the entire set of axioms of the ontology. After the new TBox axioms are ready, the ABox reasoning validates the new/updated consistency checking and classification.

2.3.2 Semantic Triples

A semantic triple (or RDF triple) is the atomic data entity that enables any knowledge to be represented in a machine-readable way (Figure 7) (W3C, 2023a). An important feature of RDF is that almost any kind of knowledge repository may be represented as a collection of RDF triples (RDF graph or just RDF) (W3C, 2023a). Also, there are many tools and libraries to support the manipulation of RDF triples and graphs (W3C, 2023a).

For example, a collection of issues from an issue tracking system may be represented as an RDF graph (Figure 8). The predicates used shall comply with a defined schema (semantic); in the example, the predicates “type”, “label”, and “description” are based on the RDF

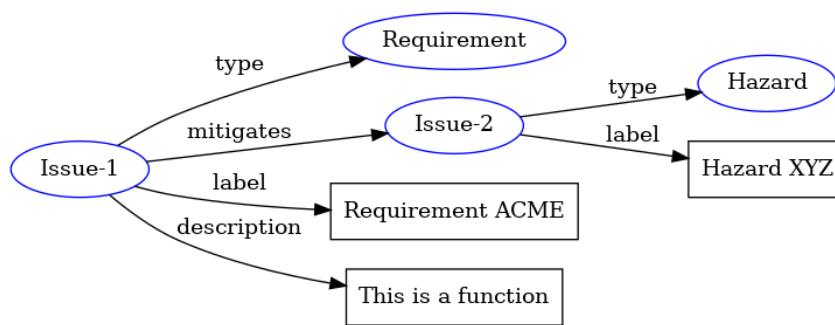
Figure 7 – RDF triple



Source: The author (2023)

Schema (BRICKLEY; GUHA, 2014), while the “mitigates” refers to the ARCADE ontology that will be discussed in the Chapter 5.

Figure 8 – RDF graph example



Source: The author (2023)

2.4 PREVIOUS RESEARCH ON THE TOPIC

In this section, we cover research that aims to understand how ACD interplay with the system development lifecycle, research that aims to understand how Requirements Engineering (RE) interplays with ACD, and finally research that is related to the ARCADE framework.

2.4.1 Incremental Assurance Case Development

The adoption of incremental safety assurance practices is challenging, and there are different approaches reported in the literature. Such approaches constitute the primary studies for our SMS. This section presents a comparison of our work with other relevant existing literature reviews and surveys.

Doss and Kelly (2016) presented the results of a practitioner survey that addressed opinions towards challenges and opportunities posed by the use of agile methods in safety-critical development. They found that practitioners supported the notion that ACs need to be constructed and evaluated incrementally. Also, practitioners indicated that external stakeholders

such as regulators and certifiers should take part in safety assurance alongside the system development. The study did not collect from participants information on the integration of ACD with SDLC.

Myklebust, Hanssen and Lyngby (2017) surveyed organisations and analysed SACs within the railways domain. They applied questionnaires to the organisation's representatives to gather information on typical SDLC models used in the safety-critical development, information about the profile of the external stakeholders involved in certification procedures and the perceived challenges related to certification. They also analysed 35 SACs from railway projects of four Nordic countries (Norway, Sweden, Denmark, and Finland) and found that they all used non-structured natural language. The authors did not discuss in the article the integration of ACD and the software development practices.

Gade and Deshpande (2015) presented a literature review on assurance-driven software design approaches and similar techniques. They did not discuss the methodological steps (e.g., search string used, year range considered, databases accessed) to select the studies. They neither performed a systematic study nor stated clear research questions. Also, they did not cover the integration between the incremental ACD and SDLC.

Heeager and Nielsen (2018) conducted an SLR to understand how agile processes can be used in safety-critical software development. They elaborated a conceptual model to organise concepts, challenges, and relations among concepts and challenges. The challenge of incremental verification and validation of safety surfaced in the study. Achieving balance in the extent of documentation produced during development is an important problem area. From a safety perspective, certification documentation is essential to demonstrate compliance, and this strict focus hinders agile practice adoption. The study acknowledges that the early and continuous production of software and its supporting evidence is a concern in the research community. However, no further analysis is provided for this aspect.

Although the above works cover many aspects related to ACD or incremental development of SCS, none of these works performs a comprehensive analysis of the integration of ACD with the SDLC.

2.4.2 Requirements Engineering and Assurance Case Development

Since its conception, Assurance Cases have been meant to be an essential artefact in the SCS development lifecycle. Special attention has been given to how ACs could be constructed

and assessed and how their development interplays with the SDLC. We are particularly interested in the interplay of RE and ACD.

De La Vara et al. (VARA et al., 2016) performed a survey of practices for Safety Evidence Change Impact Analysis (SECIA). The study aimed to characterise the state of practice and identify improvement opportunities. The study found that requirements specifications are essential in SECIA, design, test case, and traceability specifications. So, change analysis often involves these four types of artefacts, the backbone of ACs. The study found that change analysis usually does not comprise the inspection of ACs. They concluded that the industry might not adequately address how ACs are affected by changes in other artefact types and how to manage AC evolution.

Cheng et al. (CHENG et al., 2018) surveyed practitioners' perceptions of AC in SCS development to identify challenges and best practices. The results reaffirmed the benefits of ACs for communicating safety arguments and managing safety issues. ACs are a bridge between software developers, safety analysts, and certification experts, helping exchange expertise and connecting their works. They identified seven significant challenges; one of them was the lack of mechanisms to manage changes of ACs upon requirements changes. Another difficulty is gathering sufficient and accurate information for safety arguments. Both challenges were related to the insufficient integration of ACD into the development process. They identified six best practices, one indicating that ACD should start early and continuously through the process. Another recognised practice is to rely on ACs to strengthen the safety analysis process. Multiple stakeholders' involvement in reviewing ACs improves safety requirements and raises teams' confidence in the safety argumentation. Our survey provides insightful evidence that integrating ACD and RE is impactful and needs more investigation.

Recently, an important initiative mapped out the state of RE practice in the industry. The NaPiRE (Naming the Pain in Requirements Engineering) project investigated the contemporary problems practitioners experience in their industrial project settings through a global family of replicated surveys (FERNÁNDEZ et al., 2017). The data collected comprised 228 companies spread over ten countries. The global survey collected data from various perspectives, including a look into the status quo on practices, techniques, and processes towards RE (WAGNER et al., 2019). NaPiRE aimed to investigate general problems of RE without discussing specificities for SCS development or certification procedures.

The cross-collaboration of safety and requirements engineers contributes to bridging the gap between safety and software engineering (HATCLIFF et al., 2014). In the context of RE

and SCS development, Martins and Gorschek (2020) realised a series of in-depth interviews with practitioners of 11 companies regarding the interplay of RE and safety analysis. The study investigated the views of practitioners on the challenges of RE for SCS, how the RE process occurs concerning safety requirements, how the system requirements and safety are integrated, and how the certification process impact RE for SCS. They found three out of 11 companies with cross-function teams comprising safety and requirements professionals and three other companies in which safety engineers participate in requirements teams. None of the interviewed companies reported the participation of requirements engineers in the safety team. Regarding ACD, ACs were not mentioned by any company as the interview script didn't contain any questions.

Finally, Rinehart, Knight and Rowanhill (2017) conducted a literature review and a survey with 11 organisations to characterise what the research community consider as benefits and deficits of AC and to identify mechanisms to achieve these benefits and empirical evidence to support them. Their results also corroborated the idea that comprehensiveness is an essential benefit of ACs, both to refine and avoid missing requirements. Practitioners also see them as an approach to dealing with modern systems, in which novelty and high-level requirements drive the characteristics of the product as much as regulations do. The study focused on organisational aspects, methods for construction, and notations/tools. They did not discuss the integration of AC and development processes.

Although these studies provide valuable information on the state of practice regarding ACD and RE for SCS, they did not address the interrelation between RE and ACD. Indeed, to the best of our knowledge, this survey is the first empirical study investigating the integration of RE and ACD during the development of SCS.

2.4.3 Integrated Management of Project and Assurance Information

In recent years, researchers recognised the challenge development teams face regarding the collection and organisation of project information into assurance cases. Therefore, a number of approaches have been proposed to deal with this necessary flow of data that permeates any SCS project. In this section, we discuss related works that address the integrated management of project and assurance information. The works are summarised in Table 3.

Although our work is out of the scope of the model-based domain, there are initiatives in this domain that aim to leverage the management of assurance cases (VARA; RUIZ; ESPINOZA,

Table 3 – Overview of related works on integrated management

Work	Information	Integra- tion	Data Analysis	AC Generation
Denney and Pai (2012)	Input from multiple tables, that can be considered an information model. No assurance information in the information model.		No consistency checking. No implicit knowledge classification.	Predefined assumptions/strategies are placed in the generated argument fragments. Mapping rules between the information tables and assurance case fragments. No tool, just the algorithm.
Armengaud (2014)	Input from various artefacts/tools via wrappers. Manual step for semantic integration of data into the information model. No assurance information in the model.		Consistency and completeness checking. No implicit knowledge classification.	Generation of various reports, including ACs in fixed pattern.
Cleland- Huang and Vierhauser (2018), Agrawal et al. (2019)	Input from issue tracker. Some assurance concepts in the model.		Consistency checking via heuristics. No discussion on completeness. No implicit knowledge classification.	Generation of AC in fixed pattern. Manual editing to include assurance-related information. Semi-automated support to update AC for future versions of the project data.
ARCADE (our work)	Assurance and regular project information supported by the model. A reference implementation of project information retriever. Interchange format for automated input.		Consistency checking. Completeness assessment. Implicit knowledge classification. Use of semantic web technology.	Generation of AC in two argumentation patterns. Support for multiple argumentation patterns. No editing/management of the AC model (throwaway).

Source: The author (2023)

2018). Some examples are the works of (HAWKINS et al., 2015), (KOKALY et al., 2016) and (VARA et al., 2022). The main characteristic of such approaches is to define metamodels for input and output data, as they are supposed to be a part of a model-based toolchain. In fact, if a team is properly using a model-based approach for SCS development, it is likely that they will not face some of the problems discussed in the literature. However, model-based approaches have low usage reported in recent surveys related to RE and safety assurance (NAIR et al., 2015; MARTINS; GORSCHKE, 2020; ALMENDRA et al., 2022). On the other hand, the modern development processes for SCS development, such as Agile Software Development, are gaining

traction (HEEAGER; NIELSEN, 2018). These processes tend to make great use of issue trackers and application lifecycle management tools (DIGITAL.AI, 2022).

Denney and Pai (2012) proposed a lightweight method for automatically generating fragments of an AC based on information extracted from an existing set of artefacts. The method aims to give systems engineers a way to focus on the evolution of development artefacts without the need to create and maintain a separate artefact for the AC report. The approach retrieves data from multiple tables, which can be considered an information model. There is no incorporation of concepts related to requirements/design rationale or assurance in these tables. Also, there is no consistency checking or classification of implicit knowledge. In the generated argument fragments, predefined assumptions and strategies are placed. The work provides an algorithm that incorporates mapping rules between the information tables and AC fragments. There is no tool available.

Armengaud (2014) proposes a solution for the compilation of an assurance case based on automated extraction of information coming from existing work products. The approach comprises a wrapping step when data from various sources in the development process are extracted to be ingested in a database. Wrappers can be implemented for the extraction. The second step is the harmonisation of data, that is, when the data from various sources are semantically interconnected and mapped to the metamodel defined by the approach. Finally, the data harmonised is used to generate some reports, including an AC report. The purpose is to monitor development status and perform some checks such as completeness of description (are the elements complete – e.g., all requirement fields filled) or completeness of traces (are all the traces available). Although some steps are automated, the whole process is performed by someone using a tool. The harmonisation step is not automated. There is no support to register links from hazards and requirements to assurance information such as assumptions, justifications, and strategies. There is no discovery of implicit knowledge. Finally, the assurance case report generated is structured only in a fixed argumentation pattern, which is not credited but resembles the Functional Decomposition Pattern (KELLY; MCDERMID, 1998).

The works of Cleland-Huang and Vierhauser (2018) and Agrawal et al. (2019) propose the SAFA approach, which generates assurance cases from project traceability data, allows assurance-related information to be included in the generated ACs, and finally, supports the management of these ACs upon the evolution of the project data. The approach is based on a traceability information model that is implemented inside an issue tracker. The information model covers safety-related concepts such as hazards and links; but it does not cover justifica-

tion and strategies (only assumptions), which are key to assemble final assurance arguments. The SAFA approach requires the inclusion of such information later after the generation of the arguments. Consistency checking is performed by means of rules implemented as heuristics in Java language. There is no automatic classification of implicit knowledge. The generation of safety arguments is based on a fixed pattern, including the inclusion of fixed strategies. After the generation, users can edit the argument and include assurance-related information to complete the argumentation. The focus of the SAFA method is to support the management of deltas between versions of assurance cases, so users can identify areas of the argument that need to be manually edited after a new version of the project data is loaded into the solution.

We found that all approaches manage the assurance rationale inside the assurance case arguments, that is, there is no place to record assurance rationale. In all approaches, ACs are external artefacts maintained alongside the development. A consequence is that an AC structure tends to follow a single specific perspective; thus, the construction of other AC reports is needed if the team wish to view the same information organised from another perspective.

In our work, we aim to design an information model that maps the project repositories as they are. After the extraction, the trace links will be fed into the ontology, and some additional classifications and relationships could be inferred and fed back into the model. Also, we support generating an assurance case based on multiple argumentation patterns. By allowing the integration of assurance rationale information into the project management tool, we provide a lightweight mechanism for continuous creation and review of the project repository data traceability.

2.5 CHAPTER SUMMARY

In this chapter, we addressed three themes that theoretically support this research. In the first one, we introduced the concepts, methods and problems currently faced towards Assurance Case Development (Section 2.1). In the second theme, Section 2.2, we addressed the concepts and principles of RE for SCS, with focus on traceability and management tools. In the third theme, Section 2.3, we presented definitions related to semantic web. Finally, in Section 2.4, we addressed how ACD is being worked out together with software and requirements processes, showing previous research on the topic.

3 INCREMENTAL DEVELOPMENT OF ASSURANCE CASES

Assurance Case Development (ACD) early and continuously alongside the Software Development Life Cycle (SDLC) is a recommended practice (KELLY, 2018). In the context of iterative and incremental development approaches, it is paramount that the production of certification documentation accompanies the SDLC (HEEAGER; NIELSEN, 2018). Continuous development anticipates the assessment of safety-related project decisions, thus reducing the confirmation bias towards producing documentation upon a finished product. Practitioners report that lack of integration between ACD and SDLC makes it harder to handle changes and to discover and organise information that should be presented in the ACs (CHENG et al., 2018). However, such integration is challenging (NAIR et al., 2015; VARA et al., 2016).

Given the need and challenges of developing ACs, we carried out a Systematic Mapping Study (SMS) to investigate how the ACD can occur in an integrated way with SDLC. We seek to identify the approaches in the literature, classify how the integration can occur, identify the roles involved, and identify the expected outcomes of such integration. We use the methodology proposed by (KITCHENHAM; CHARTERS, 2007) to perform our research.

This chapter is organised as follows. In Section 3.1, we describe the research methodology used to conduct the study. In Section 3.2, we present the results and the synthesis related to our research questions. In Section 3.2.7, we discuss the findings of the study. Finally, our conclusions are presented in Section 3.4.

3.1 SMS RESEARCH METHODOLOGY

We are interested in mapping the state of the art regarding the incremental development of AC alongside SCS development. We used the PICOC criteria (KITCHENHAM; CHARTERS, 2007) to structure and scope our research objectives:

- **Population:** Our focus is on teams or organisations developing SCS that need to deliver ACs for external revision and approval;
- **Intervention:** We want to investigate approaches that promote incremental ACD throughout the software development process;
- **Comparison:** it does not apply; ¹

¹ It is out of scope of this study to discuss the direct comparison of incremental approaches with non-

- **Outcomes:** We want to characterise the approaches by their purpose, the SDLC models adopted, the way integration occurs, the roles involved, and expected outcomes;
- **Context:** We only consider approaches that address the development of ACs integrated with SCS development.

There are model-driven engineering (MDE) approaches for ACD, and a bibliographic review on this topic can be found in (VARA; RUIZ; ESPINOZA, 2018). However, there is a gap in research targeting practitioners that develop AC without MDE or automated toolchains (CHENG et al., 2018; VARA et al., 2016; NAIR et al., 2015), which motivated the scope of this study. As this mapping study focuses on integrating ACD with software development processes, we did not include studies focusing only on the construction or assessment of ACs. We seek to answer the research questions in Table 4.

Table 4 – Research questions and motivations.

Research Question	Motivation and Objective
RQ1. What are the characteristics of the approaches that promote incremental ACD integrated with SDLC?	We aim to identify and classify approaches proposed to integrate the development of ACs into the safety-critical system development processes.
RQ2. How does the integration of ACD and SDLC occur?	We aim to identify the activities proposed, which SDLC model they target, and how the integration with software and safety engineering processes occurs.
RQ3. What are the roles involved in the approaches?	We aim to identify what are the roles involved in the approaches, both from inside the development team and from external partners/certifiers.
RQ4. What are the expected outcomes of the proposed approaches?	We aim to identify the expected benefits and how they are measured.
RQ5. What are the challenges and needs addressed by the approaches?	We aim to identify the practical problems addressed by the approaches promoting continuous safety assurance and system development processes.
RQ6. What are the techniques and tools used for assurance case development?	We aim to identify the structuring and assessment techniques used by existing approaches and the tool support that those approaches provide.

Source: The author (2023)

3.1.1 Search Strategy

The following scientific databases were used to search for studies: ACM Digital Library, IEEE Xplore Digital Library, Science Direct, Springer Link, Scopus, and Engineering Village incremental ones.

(Compendex). After applying inclusion and exclusion criteria, we applied a snowballing search in the references of the selected studies.

We constructed the search string based on the PICOC criteria. We conducted pilot searches to refine the search string so that all keywords contributed to the study goals. The defined search string below was used to query within keywords, title, abstract and full text² of the publications:

```
("safety case" OR "safety argument" OR "assurance case" OR
"assurance argument" OR "dependable case" OR "dependable argument" OR
"trust case" OR "trust argument") AND
("safety-critical development" OR "software process" OR
"development process" OR "application development" OR
"software development" OR "system development")
```

The decision to apply the search string also to the full text was motivated by the observation that key studies used as the control sample do not bring the terms neither in the title nor in the abstract. The first part of the search string includes AC-related terms to constrain the domain, and the second part includes development-related terms. In the string refinement process, we found that the term “safety-critical development” is not the only way to refer to SCS development; for example, the term “system development” is also used.

3.1.2 Inclusion and Exclusion Criteria

We adopted inclusion (I) and exclusion (E) criteria for study selection as defined by (KITCHENHAM; CHARTERS, 2007):

- (I)** Primary study;
- (I)** Study published in any year until January 2020;
- (I)** Study that addresses the incremental development of ACs in an integrated way with safety-critical development.
- (E)** Short paper (equal to or fewer than four pages);
- (E)** Duplicated study (only one copy of each study was included);

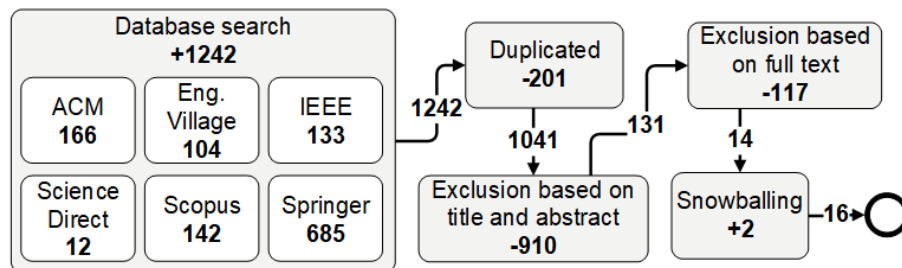
² Except for Scopus database that does not store full documents.

- (E) Non-English written;
- (E) Grey literature;
- (E) Redundant paper of the same authorship;
- (E) Publication whose text was not available;
- (E) Study irrelevant to the research, taking into account the research questions;
- (E) Study focused only on the construction or assessment of ACs with no discussion towards integration into SDLC.

3.1.3 Study Selection

We performed the database search and retrieved a total of 1,242 articles. The selection process is depicted in Figure 9. We used Parsifal tool ³ to check for duplicates. After applying the inclusion/exclusion criteria, we selected 14 articles. We then performed a forward and backward snowballing procedure starting from these 14 articles, based on the method described in Wohlin (2014). The backward procedure started from the references listed in each paper, and we did not find new articles. The forward procedure started from the citations found in Google Scholar,⁴ and two studies were found. In the citations and references of these two studies, we did not find new articles so the search had one degree of depth. Thus, our final set of papers is composed of 16 studies (Table 5 and Table 8).

Figure 9 – Study selection methodology steps.



Source: The author (2023)

³ Parsifal. Available at: <<https://parsif.al/>>. Accessed in: May 1, 2023.

⁴ Google Scholar. Available at: <<https://scholar.google.com>> Accessed in: May 1, 2023

Table 5 – SMS selected studies.

Id	Title	Authors
SMS-1	The potential for a generic approach to certification of safety critical systems in the transportation sector	Papadopoulos and McDermid (1999)
SMS-2	Assurance based development of critical systems	Graydon, Knight and Strunk (2007)
SMS-3	Evidence-based development - applying safety engineering techniques to the progressive assurance and certification of complex systems	Dick and Wills (2008)
SMS-4	Assurance-Driven Design	Hall and Rapanotti (2008)
SMS-5	Safety Lifecycle Activities for Autonomous Systems Development	Alexander, Kelly and Gorry (2010)
SMS-6	An Iterative Approach for Development of Safety-Critical Software and Safety Arguments	Ge, Paige and McDermid (2010)
SMS-7	Towards a Language for Communication among Stakeholders	Matsuno et al. (2010)
SMS-8	A Comprehensive safety lifecycle	Knight et al. (2015)
SMS-9	Generation of assurance cases for medical devices	Lin and Shen (2015)
SMS-10	Safety Case Driven Development for Medical Devices	Ruiz et al. (2015)
SMS-11	Continuously revised assurance cases with stakeholders' cross-validation: a DEOS experience	Kuramitsu (2016)
SMS-12	The agile safety case	Stålhane and Myklebust (2016)
SMS-13	An evidence-based risk-oriented V-model methodology to develop ambient intelligent medical software	Cicotti (2017)
SMS-14	Safety engineering with COTS components	O'Halloran, Hall and Rapanotti (2017)
SMS-15	Discovering, Analyzing, and Managing Safety Stories in Agile Projects	Cleland-Huang and Vierhauser (2018)
SMS-16	Introducing agile practices into development processes of safety critical software	Łukasiewicz and Górski (2018)

Source: The author (2023)

3.1.4 Quality Assessment

The quality assessment (QA) aims to measure how much we could rely on the selected studies as valuable contributions to our mapping. The QA of selected studies in our SMS was achieved by a scoring technique to evaluate the selected studies' credibility, completeness and relevance. All papers were evaluated against a set of seven quality criteria:

Q1. Is there a clear statement of the goals of the research (DERMEVAL et al., 2016)?

- Q2.** Is the proposed technique clearly described (DERMEVAL et al., 2016)?
- Q3.** Is there a discussion about the results of the study (DERMEVAL et al., 2016)?
- Q4.** Are the limitations of this study explicitly discussed (DERMEVAL et al., 2016)?
- Q5.** Is the article relevant for practitioners (TIWARI; GUPTA, 2015)?
- Q6.** Is there sufficient discussion of related works (TIWARI; GUPTA, 2015)?
- Q7.** Does the study significantly increase the knowledge about integrating ACD and SDLC research (TIWARI; GUPTA, 2015)?

Each quality assessment question is judged against three possible answers: “Yes” (score=1), “Partially” (score=0.5) or “No” (score=0). Then, the quality score is computed by taking the sum of the scores of the answers to the questions. The quality scores of the selected studies are presented in Table 6.

Table 6 – Quality scores and number of citations for each selected study.

Study	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Total	Qual.	Citations
SMS-1	1	1	0.5	0	1	0	.5	4	57.1%	59
SMS-2	1	1	0	0	1	0.5	.5	4	57.1%	74
SMS-3	1	0.5	0	0	1	0	0	2.5	35.7%	3
SMS-4	1	1	1	0	0	1	.5	4.5	64.2%	36
SMS-5	1	0.5	0	0	1	0	1	3.5	50%	13
SMS-6	1	1	1	1	1	0	1	6	85.7%	66
SMS-7	.5	.5	0	.5	.5	.5	.5	3	42.9%	21
SMS-8	1	1	0	0	1	0	1	4	57.1%	1
SMS-9	1	1	1	0	1	1	1	6	85.7%	6
SMS-10	1	1	.5	0	1	.5	.5	4.5	64.2%	2
SMS-11	1	1	1	0	1	1	1	6	85.7%	0
SMS-12	1	1	1	.5	1	1	1	6.5	92.8%	11
SMS-13	1	1	1	.5	1	1	.5	6	85.7%	2
SMS-14	1	1	1	1	1	0	1	6	85.7%	4
SMS-15	1	1	1	1	1	1	1	7	100%	7
SMS-16	1	1	0	0	1	.5	.5	4	57.1%	0
Average	.97	.91	.56	.28	.91	.5	.72	4.84	69.2%	19.0

Source: The author (2023)

The mean of quality was 69.2%, which indicates that the found studies constitute a good source for practitioners. Q4 (discussion of limitations) was the only criterion below 50%, and Q6

(discussion of related works) achieved 50%. The quality of the reports is key to disseminating research results in industry practice. However, the low quality identified in the discussion of limitations impairs practitioners from a better understanding of the approaches' pitfalls.

3.1.5 Venues of Publication

The 16 selected studies are from 14 different venues (conferences, journals, and book chapters) (see Table 7). Eight venues are related to reliability, safety and dependability research (V1–8), and six venues are related to software engineering and general computer science (V9–14). As the subject of this mapping covers two fields (safety engineering and software engineering), it was expected to find studies in venues of both scopes. Only two venues had two studies selected: the journal of Reliability Engineering and System Safety (V1) and the International Workshop on Assurance Cases for Software-Intensive Systems – ASSURE (V2). V1 is devoted to methods for the enhancement of the safety and reliability of complex technological systems, and V2 is mainly devoted to discussing the application of assurance case technology for supporting the assurance of software-intensive systems.

Table 7 – Venues of publication.

Venue	Studies
V1: Reliability Engineering and System Safety (<i>journal</i>)	SMS-1, SMS-14
V2: Intl. <i>Workshop</i> on Assurance Cases for Software-Intensive Systems	SMS-8, SMS-12
V3: Intl. <i>Conference</i> on Dependable Systems and Networks	SMS-2
V4: IET Intl. <i>Conference</i> on System Safety	SMS-3
V5: Intl. <i>Conference</i> on Computer Safety, Reliability, and Security	SMS-10
V6: <i>Journal</i> of Reliable Intelligent Environments	SMS-13
V7: Pacific Rim Intl. <i>Symposium</i> on Dependable Computing	SMS-7
V8: SEAS DTC Technical <i>Conference</i>	SMS-5
V9: Intl. Requirements Engineering <i>Conference</i>	SMS-15
V10: PeerJ Computer Science (<i>journal</i>)	SMS-11
V11: Intl. <i>Conference</i> on Software Engineering Advances	SMS-4
V12: Intl. <i>Conference</i> on Agile Software Development	SMS-16
V13: Agile <i>Conference</i>	SMS-6
V14: Computer and Information Science (<i>book series</i>)	SMS-9

Source: The author (2023)

3.2 SMS RESULTS AND ANALYSIS

In this section, we present the data retrieved for each RQ.

3.2.1 Overview of the Studies (RQ1)

We discuss in this section the characteristics of the selected studies regarding their publication year, application context, countries, domains, standards, type of contribution and type of evaluation method. Table 8 presents an overview. The interest in ACs has spread out of its cradle (UK), and there were studies from eight countries. Although there is a range of 19 years in the selected studies, the research interest has been increasing, with nine studies (56.2%) in the last six years. The majority of selected studies are in the application context of academic research. The low number of studies from industry (6, 37.5%) contrasts with recent surveys that pointed out the integration of ACD into SDLC as an open problem (MYKLEBUST; HANSSEN; LYNGBY, 2017; CHENG et al., 2018).

The domains targeted by the selected studies are common in safety research, and no particular domain draws more attention than others. It is interesting to see works related to network systems; these studies bring the AC to the broader scope of dependability and treat ACs as *dependable cases* (SMS-7, SMS-11). The dependable case notation adopted by SMS-7 and SMS-11 is designed to argue over dependability properties, including safety. Two studies (SMS-13, SMS-16) targeted the standard ISO/IEC 15026, which aims to guide AC management. This standard was published in 2011, yet few studies have adopted it. No study referred to the Structured Assurance Case Metamodel (SACM).

We identified studies targetting two of the first standards to oblige the presentation of ACs: Def Stan 00-56 (Military) and CENELEC EN 50129 (Railway). In the medical domain, FDA⁵ regulations recommend using ACs as part of certification documentation (SMS-9, SMS-13). ISO 14971 is related to the risk management of medical devices and provides general objectives to be guaranteed by vendors, and it does not impose the presentation of an AC. However, in the studies SMS-10 and SMS-16, the AC is applied to organise the risk-related information and decisions. Some studies do not target a specific standard due to their general-purpose proposals (SMS-2, SMS-4, SMS-7, SMS-11) or because the domain lacks specific standards (SMS-15). The IEC 61508 (functional safety) standard, cited in studies SMS-1

⁵ Food and Drug Administration/USA

Table 8 – Overview of the selected studies (RQ1).

Id	Study	Con- text	Country	Domain	Standard	Contrib.	Eval.
SMS-1	(PAPADOPOULOS; MCDERMID, 1999)	A	UK	Aviation, Railways	DO-178B, EN 50129, IEC 61508	Process	–
SMS-2	(GRAYDON; KNIGHT; STRUNK, 2007)	A/I	USA	Aviation	-	Process	Illustr. scenario
SMS-3	(DICK; WILLS, 2008)	I	UK	Aviation	DO-178B	Process	–
SMS-4	(HALL; RAPAN-OTTI, 2008)	A	UK	Aviation	-	Process	Illustr. scenario
SMS-5	(ALEXANDER; KELLY; GORRY, 2010)	A/I	UK	Aviation	Def Stan-00-56	Process model	–
SMS-6	(GE; PAIGE; MCDERMID, 2010)	A	UK	Aviation	DO-178C	Process	Illustr. scenario
SMS-7	(MATSUNO et al., 2010)	A/I	Japan	Network sys.	-	Process model	Case study
SMS-8	(KNIGHT et al., 2015)	I	USA	Generic	-	Process model	–
SMS-9	(LIN; SHEN, 2015)	A	USA	Medical	FDA ^a	Process	Illustr. scenario
SMS-10	(RUIZ et al., 2015)	A	Spain, Brazil	Medical	ISO 14971, IEC 62304	Process	Illustr. scenario
SMS-11	(KURAMITSU, 2016)	A	Japan	Network sys.	-	Process	Case study
SMS-12	(STÅLHANE; MYKLEBUST, 2016)	A	Norway	Generic	IEC-61508	Process	Illustr. scenario
SMS-13	(CICOTTI, 2017)	A	Italy	Medical	IEC-62304, ISO/IEC-15026, FDA ^b	Process	Case study
SMS-14	(O'HALLORAN; HALL; RAPAN-OTTI, 2017)	A/I	UK	Maritime	Def Stan-00-56	Process model	Case study
SMS-15	(CLELAND-HUANG; VIER-HAUSER, 2018)	A	USA	Aviation	-	Process	Case study
SMS-16	(ŁUKASIEWICZ; GÓRSKI, 2018)	A	Poland	Medical	ISO-14971	Tailoring method	Case study

Context: A - academic, I - industry. ^a Infusion pumps total product lifecycle: Guidance for industry and FDA staff. December, 2014. ^b Title 21, Chapter I, Sub-chapter H–Medical Devices, Part 820 Quality System Regulation. April, 2018.

Source: The author (2023)

and SMS-12, does not comprise ACs, but it is expected to include them in its next version (HANSEN; STÅLHANE; MYKLEBUST, 2018).

We classified the studies in terms of the type of research and evaluation method. Research

type is based on (WIERINGA et al., 2006), and the evaluation method is classified according to (EASTERBROOK et al., 2008). All selected studies were better classified as *Solution Proposals* as research type. The evaluation method of the studies was *Illustrative Scenario* (6, 37.5%) and *Case Studies* (6, 37.5%). Four articles (25%) do not discuss any evaluation. All case studies involved only a single case and a single unit of analysis (the approaches themselves).

Regarding the type of contribution, most of the studies took a prescriptive line, suggesting methods, methodologies or processes (11, 68.7%). Four studies proposed process models (25%), and only one proposed a tailoring method.

3.2.2 Integration of ACD and SDLC (RQ2)

In this section, we characterise the approaches regarding the reference *SDLC Model* to which the approach is targeted, the *Task Integration* approach adopted to combine ACD into SDLC, and the *Argument Development* approach to developing the AC. Table 9 summarises the classification of the studies.

Table 9 – Classification of studies by SDLC Model, Task Integration and Argument Development

Study	SDLC Model	Task Integration	Argument Development
SMS-6	Agile	Blended	Scope
SMS-12	Agile	Blended	Scope
SMS-15	Agile	Blended	Scope
SMS-16	Agile	Blended	Scope
SMS-5	Spiral	Blended	Scope
SMS-9	Unified	Blended	Scope
SMS-2	Incremental	Blended	Argument
SMS-4	Incremental	Blended	Argument
SMS-8	Incremental	Blended	Argument
SMS-11	Incremental	Parallel	Phased
SMS-3	V-model	Parallel	Phased
SMS-10	V-model	Parallel	Phased
SMS-13	V-model	Blended	Phased
SMS-14	V-model	Parallel	Argument
SMS-1	Waterfall	Parallel	Argument
SMS-7	Waterfall	Blended	Phased

Source: The author (2023)

3.2.2.1 SDLC Model

An SDLC model describes a framework of processes and activities that concern the lifecycle of a development process (ISO, 2010). Ruparelia (2010) provides a historical view and categorisation for *SDLC models*, which we adopted to classify the selected studies in our SMS. The discussion towards known SDLC models eases practitioners' task of comparing the research settings and their own.

Some studies explicitly refer to their reference SDLC model; others we classified are based on the approach's presentation. We extracted from the selected studies the model of SDLC, using these categories: Waterfall, B-model, V-model, Incremental, Spiral, Wheel-and-spoke, Unified Process (RUP – Rational Unified Process), and Rapid Application Development (RAD). According to Ruparelia (2010), the Incremental model may be used to classify a modification of the waterfall that approaches the spiral model, but not necessarily resembles the modern agile methods. The Agile model is characterised by the cycled development of increments, although the literature has already used these terminology to refer to other models. The RAD category includes Agile, Extreme programming, Joint Application Development, Lean, and Scrum. For this mapping study, we considered all the RAD category as "Agile".

Spiral, Unified, and Waterfall were the least adopted models (only 4/16 works). SMS-1 and SMS-7 presented their lifecycle in a linear Waterfall view. Although SCS development requires some up-front analysis and design, the Waterfall approach has become rare. The unified approach is also in disuse, and a single study by SMS-9 used it as a base process – the work focuses on demonstrating how to generate ACs from typical artefacts of RUP. SMS-5 targeted the Spiral model and discussed how the successive refinement approach of the Spiral model can be used to develop autonomous, non-deterministic systems.

Incremental. SMS-2 presented an assurance-based development approach. This approach focuses on the development process in the continuous refinement of goals and claims in the AC, which triggers developers to make development choices and modify the development artefacts and AC accordingly. SMS-8 extends SMS-2 by providing support for ACD in more development phases, such as operation. SMS-4 proposes the Problem Oriented Engineering (POE) framework, which guides the system design process into a series of refinements based on problem finding and solution selection. This process requires the recording and justification of each design decision, so it is straightforward to use such rationale directly or adapt it to ACs. These three studies were mapped as Incremental because the choice of increments to

be developed depends on the safety argument breakdown/refinement. SMS-11 proposal is based on the DEOS (Dependability Engineering for Open Systems) Process (TOKORO, 2015), which comprises cycles of change accommodation and failure response triggered by operational issues. The increments are change requests or failure reports that need to be addressed. It is an approach focused on the maintenance phase.

V-Model. SMS-3 defined a set of stages of safety assurance, correlating the evidence needed at each stage with activities from the V-model. Thus, for a given single requirement, the approach defined points in the SDLC in which evidence should be collected to compose the AC. The approach is proposed to address the DO-178B standard, which requires AC development to be performed alongside development. SMS-10 and SMS-13 defined a set of tasks for ACD and proposed them alongside a V-model process, in which ACD milestones accompany project progress. SMS-10 proposed the approach targetting medical device regulations but used ISO 26262 (Automotive) as motivation for the integration. SMS-14 also used the POE framework as its base process, integrating POE into a V-model process. Thus, it proposes the development of the AC in stages alongside the development process.

Agile. SMS-6 proposed an iterative process based on a generic agile process model in which teams develop the AC incrementally. The process emphasises the importance of balancing up-front analysis and design to lighten the process. Scrum was the target process used in studies of SMS-12 and SMS-15. Both studies propose the construction of ACs during the iterations (*sprints*) and the AC assessment at the end of the iterations. Preliminary discussions on safety arguments and evidence should be carried on in advance of iteration during planning or backlog grooming activities. SMS-16 proposed a tailoring method comprised a set of project practices, a set of safety assurance arguments, and a user guide to support the customisation. The approach guides users to select the software development practices to be later implemented into the SDLC. The set of safety arguments serves as a template for developing the ACs.

All the agile-based approaches mentioned in this section prescribed just a preliminary AC construction, so that teams have an initial argument structure to be further refined and expanded throughout the iterations. Thus, the “just enough” principle of agile methods reflects directly in the development of ACs. This approach does not deter safety experts from evaluating the quality of the ACs regularly. Indeed, the early availability of updated ACs and software releases provide a point for safety and software experts iteratively integrate their works.

3.2.2.2 Task Integration

We extracted from the selected studies how the tasks related to AC development were integrated into the development process. We devised two types of *Task Integration* approaches: Parallel and Blended. The approaches may be combined in different phases of development.

Parallel integration refers to approaches that separate the ACD flow from the software development flow. The main flow is the source of the content (goals, arguments, evidence), which feeds the development of the AC artefact in a different flow.

Blended integration refers to approaches that merge the ACD activities into the software development flow, keeping only one main flow. So, AC is handled as an artefact that shall be delivered/reviewed together with other development ones.

Table 9 presents the classification of the studies. The **Parallel** integration was proposed by five studies (31.3%), and mostly associated with V-model approaches (3/5 works). **Blended** integration was proposed by 11 studies (68.7%). We did not find studies that combined both types of integration for different phases of development.

We found that all approaches based on the Agile model proposed blended integration. This finding is in line with fundamental practices of agile methods, in which the team organised itself to deliver releases and their companion artefacts altogether. We also observed that for each SDLC model identified, there is at least one approach proposing blended integration. This may indicate blended integration as a versatile approach, as it was used in many different SDLC models.

3.2.2.3 Argument Development

We also looked into the approach to argument development adopted by selected studies. Although many studies refer to the AC development approaches discussed in Section 2.1, these seminal approaches were cited mostly because of their contribution towards visual or text notations. Only one of the selected studies (SMS-6) adopted the modular argument approach discussed in (KELLY, 1999). We devised three categories to classify the incremental argument development reported in the selected studies: Argument-driven, Scope-driven and Phased.

Argument-driven refers to approaches that start from high-level goals/claims, hazards and requirements, and go through a series of refinements and decomposition iterations. These

approaches put the safety argument as a first-class citizen that drives the design together with other key development artefacts. The evolution of argument refinement guides the development of ACs. Argument-driven approaches resemble Kelly's Six-Steps (see Figure 4–b), with the addition of system development activities into the argument decomposition process.

Scope-driven refers to approaches in which the development of working packages guides the AC development. Working packages may be components, requirements, or user stories, depending on how the scope of the project is broken into manageable items. When a project item is selected for development, the team then also develops the associated safety argumentation. In each iteration, the current assurance case is extended with the claims, arguments, and evidence so that it stays aligned with the project's progress. These approaches resemble Bishop and Bloomfield's process (Figure 4–a), where the elaboration of the assurance case is performed progressively yet is based on a preliminary assurance case. This kind of approach fits well with the agile culture, in which up-front analysis and design should be "just enough", and successive refinements happen throughout the development. The close distance between the implementation effort and the safety argument construction eases the collection and production of evidence, as the rationale is fresh in the memory of the team.

Phased approaches are those that establish specific points in the SDLC to carry out assurance case development. These approaches resemble Kelly's assurance case lifecycle (Figure 4–d), yet none of the studies considered the original terminology of Kelly (KELLY, 1999). The studies in these categories provide insights for practitioners willing to insert assurance case development into V-model-based processes – which are very common in safety-critical development (HEEAGER; NIELSEN, 2018).

Table 9 shows in the last column the approaches used by each study. **Argument-driven** approach was identified in five studies (31.2%). In three studies (SMS-2, SMS-4, SMS-8), this choice of argument development directly influenced the lifecycle, mapped as Incremental. In these studies, the "increments" are successive refinements of the ACs. **Phased** approach was identified in five studies (31.2%). This approach fits well with V-model lifecycles as in the works of SMS-3, SMS-10, and SMS-13. Phased development of AC is a mid-term between up-front and iterative development, and it may avoid the problems of full postponement. Finally, **Scope-driven** approach was identified in six studies (37.5%) (SMS-5, SMS-6, SMS-9, SMS-16, SMS-12, SMS-15). All studies classified as Agile adopted this approach.

3.2.3 Roles Involved (RQ3)

We overview the roles involved in the approaches of the selected studies. We found 25 roles cited in 12 studies, while four studies did not explicitly discuss any role (SMS-1, SMS-3, SMS-10, SMS-16). Table 10 presents all the roles identified in the selected studies.

Table 10 – Roles involved in incremental ACD approaches

Study	Role	Construction	Assessment
SMS-2	Developer	Build	Review
SMS-4	Customer	Build	Approve
	Developer		Approve
	Regulator		
SMS-5	Developer	Build	Review
SMS-6	Safety engineer	Build	Review
SMS-7	Customer	Build	Approve
	Developer		Review
	Development manager		
	Test manager		Review
SMS-8	Developer	Build	Approve
	System stakeholder		
	Certifier/regulator		Approve
SMS-9	Developer	Build	
SMS-11	Owner	Build	Review
	Developer	Build	Review
	Operator	Build	Review
	User		Review
SMS-12	Safety engineer	Build	Review
	Customer		Review
	Developer		Review
SMS-13	Assurance engineer	Build	
SMS-14	Developer	Build	Review
	Customer		Review
	Safety engineer		Review
SMS-15	Develop team	Build	Review
	Safety master	Overview	Review

Source: The author (2023)

We mapped each role to the ACD activities it is expected to perform: construction (build) and assessment (review and approve). We found roles that are known to ACD: customer (SMS-4, SMS-7, SMS-12, SMS-14), end-user (SMS-4, SMS-11), certifier/regulator (SMS-4, SMS-8),

and safety/ assurance engineer or experts (SMS-6, SMS-13, SMS-14, SMS-12). These roles were associated with assessment tasks, and the approaches proposed an early involvement of them in the SDLC.

We also identified roles related to the development team taking part in ACD tasks. *Developer* was associated with AC building (SMS-2, SMS-4, SMS-5, SMS-7, SMS-8, SMS-9, SMS-11, SMS-12, SMS-14, SMS-15) and AC reviewing (SMS-2, SMS-5, SMS-11, SMS-12, SMS-15). AC production works as a bridge between software engineering and safety engineering. Developers may not be responsible for full ACD, but they have to devise strategies to satisfy top-level claims, collect and organise evidence, and structure the final argument. From these studies, eight (out of nine) proposed the *Blended* integration.

Three management-related roles were found. SMS-7 included two roles as part of reviewers of ACs. The *Development manager* reviews cases during development phases, focusing on assuring that design and implementation meet the requirements. During the test phases, a *Test manager* reviews the testing results together with the AC produced in the development phase. SMS-15 proposed the role of *Safety master*, an additional role to the Scrum approach. This role overviews all safety-related tasks the team must perform, thus being responsible for guaranteeing that safety concerns are continually evaluated and addressed throughout the development. Also, this role overviews the construction and reviews the ACs produced by the team.

It is likely that developers also have a significant part in the studies that did not explicitly discuss any role (SMS-1, SMS-3, SMS-10, SMS-16). It is essential for research reports that they discuss the roles involved so that practitioners trying to adopt new approaches could better assess the required capabilities to perform the new activities.

Considering all roles identified in the studies, the majority (19 out of 25, 76%) were expected to perform assessment activities. This indicates that the review and discussion of AC arguments and evidence should be performed incrementally, along with the argument construction.

3.2.4 Expected Outcomes (RQ4)

We seek to understand what researchers indicated as outcomes of adopting their approaches. A clear indication of expected outcomes helps practitioners to decide on adopting/-experimenting with an approach and to self-assess its adequacy. We identified six categories

of expected outcomes in the selected studies:

EO1 – Better project management (8 studies, 50%): We classify in this category outcomes that relate to time, cost, effort, requirements, traceability and tracking management. SMS-4 expected that errors that weaken safety arguments could be found early, thus reducing late rework. SMS-6 pointed out that their approach may turn the process more “*cooperative, straightforward and adaptive*”. SMS-9 and SMS-16 indicated cost reduction as an expected result of adopting an incremental approach for ACD. SMS-3, SMS-10 and SMS-14 expect that the co-development of a system and its AC results in richer traceability between software and safety artefacts. SMS-15 noted that the incremental development of ACs provides a continuous way to assess safety issues, thus helping track project progress.

EO2 – Increased quality of argumentation (5, 31.2%): Five studies noted the increase in quality of argumentation as a benefit from the incremental ACD. They argue that the argument construction and assessment benefit from the integration with SDLC, thus providing clear and complete argument structures that would facilitate the certification process (SMS-3, SMS-8, SMS-13, SMS-15). SMS-11 noted that a participatory review of assurance arguments would lead to reduced ambiguity.

EO3 – Better stakeholder communication (4, 25%): The enhancement of communication between internal and external stakeholders was indicated as a positive outcome by some studies. SMS-7 see the dependability case as an artefact for supporting communication and agreement with customers. SMS-11 proposed that non-experts should be engaged in AC reviews and account for its safety argumentation. Such accountability would raise the sharing of systems problems among stakeholders across the SDLC. SMS-14 noted that incremental ACD engages stakeholders towards a validated solution with an agreed safety argumentation.

Table 11 – Expected outcomes of incremental ACD approaches

Expected outcome	#	%
EO1 – Better project management	8	50.0%
EO2 – Increased quality of argumentation	5	31.2%
EO3 – Better stakeholder communication	4	25.0%
EO4 – Enhanced safety and risk awareness	3	18.7%
EO5 – Support for emergent design	2	12.5%
EO6 – Better knowledge management	2	12.5%

Source: The author (2023)

EO4 – Enhanced safety and risk awareness: The integration of ACD and the SDLC

could also enhance stakeholders' and team's awareness of safety and risk management. The team's work with ACs and communication with safety experts would increase the safety awareness of developers (SMS-12). Design and developers engaged in the definition and development of ACs would foster a more risk-oriented analysis (SMS-13) and safety thinking (SMS-15).

EO5 – Support for emergent design: SMS-5 addressed the assurance of autonomous systems (AS) and argued that an incremental development of AC should align with the emergent design approach that developers of AS usually conduct. SMS-2 demonstrated how the development of SCS and its companion AC could be performed together to help manage incomplete or uncertain requirements. Systems that present complex or emergent behaviour are designed through a series of refinements and prototypes, so the rationale from trial-and-error design and field data are key to structure safety argumentation.

EO6 – Better knowledge management: Last, two studies indicated ACs development as a potential trigger for knowledge management. That is a common benefit of sharing artefacts and guiding development towards them. In particular, SMS-1 indicated standardisation and reuse of ACs as benefits. SMS-9 noted that a systematic approach for the generation of AC could leverage the capability for standard-compliance review and certification.

3.2.5 Challenges and Needs (RQ5)

The motivation of this question is to identify the practical problems addressed by the approaches that promote continuous safety assurance alongside system development processes.

We identified many challenges and needs in the studies selected. Some studies addressed more than one challenge or need. Table 12 shows the challenges and the studies that addressed them.

C1 – Integrate safety assurance practices into the SDLC: The most noted challenge was how to find ways of integrating safety assurance practices into the SDLC. SMS-10 indicated that it is still unclear how to handle safety concerns during the development phases. SMS-3 pointed out that the gathering of evidence needs to occur throughout the SDLC. SMS-13 observed that current practices address the risk management processes separately from the SDLC, thus leaving vendors to define and manage the interactions between such processes. SMS-12 and SMS-16 noted that including AC construction in an agile process is a challenging and costly change. SMS-15 added as challenging and costly the incremental development of new features and certification of the modified system. SMS-14 indicated that many standards

Table 12 – Challenges addressed by incremental ACD approaches

Challenge	#	%
C1 – Integrate safety assurance practices into the SDLC	7	43.7%
C2 – Dealing with system complexity and uncertainty	4	25.0%
C3 – Reduce development risks and costs	3	18.7%
C4 – Reuse safety arguments/cases/documents	3	18.7%
C5 – Achieve balance between functionality and safety	2	12.5%
C6 – Assurance case construction	2	12.5%
C7 – Reaching agreement among stakeholders	2	12.5%

Source: The author (2023)

already require the integration of safety engineering activities within the SDLC in an early, iterative and continuous basis.

C2 – Dealing with system complexity and uncertainty: The second-most noted challenge acknowledges that systems complexity or uncertainty may impose great risks in the case of AC production separated from the SDLC. SMS-5 focused on autonomous systems and noted how difficult it is to assure the safety of unpredictable behaviour systems. It may be unfeasible to make a major up-front design. Also, it may only be possible to assess the system based on emergent behaviour and the rationale behind the system architecture. Such context calls for integrated engineering of safety assurance and software. SMS-10 and SMS-13 noted that the increasing complexity of systems affects the safety integrity of solutions and makes it difficult to properly analyse the multiple environmental conditions. SMS-3 considered demanding the coordination and organisation of the collection, review, and publication of certification evidence.

C3 – Reduce development risks and costs: Some studies were concerned with better management of risks and costs throughout the system and assurance case development. SMS-4 and SMS-2 pointed out that development errors that weaken the safety assurance argumentation may be discovered early in the process, providing that the argumentation development occurs alongside the software. Finding errors as early as possible reduces risks and late rework efforts. SMS-9 reported that the postponement of AC construction is time-consuming and error-prone.

C4 – Reuse safety arguments/cases/documents: Three studies indicated that incremental AC could foster the reuse of ACs artefacts. SMS-6 focuses on the reuse of safety arguments across releases, suggesting an emergent specification approach. SMS-12 indicates

that existing AC holds quality and safety management argumentation that would not change substantially on following releases or products. SMS-14 noticed the difficulties in incorporating third-party components and their corresponding safety artefacts into the SCS project, highlighting the need to carefully manage assumptions and expectations across supply chains.

C5 – Achieve a balance between functionality and safety: Two studies addressed the impracticality of taking apart functionality and safety. SMS-2 noted that satisfying functional requirements goals is insufficient in the context of safety-critical development. On the other side, a system that is safe but fails to satisfy requirement goals is also unacceptable. SMS-8 stated that assuring all engineering elements in the lifecycle contributes to the assurance of system safety. From a value-based perspective, implemented functions without proper assurance argumentation are of marginal value.

C6 – Assurance case construction: Easing the composition of safety arguments and evidence was cited by two studies. SMS-1 highlighted the significant variation that Assurance Cases (ACs) go through across industries and the potential for generic approaches for certifications. SMS-7 pointed out that writing cases, decomposing arguments and choosing evidence are still challenging.

C7 – Reaching agreement among stakeholders: Two studies cited how to better use ACs to reach agreement among stakeholders. SMS-7 indicated that more active participation of stakeholders could improve the incremental review of ACs. SMS-11 noted that the involvement of non-expert stakeholder in the arguments review could be better investigated.

3.2.6 Techniques and Tools (RQ6)

We aim to identify the structuring and assessment techniques used by existing approaches and to analyse if the way of structuring and assessment of ACs impacts its continuous development. We also want to identify the level of tool support that those approaches provide.

An assurance case is a category of Argumentation-Induced evidence structuring technique (NAIR et al., 2014). There are many ways to construct and assess them. In this section, we overview the notation languages, assessment techniques, and tools used in the studies.

Table 13 depicts the notation languages adopted by the selected studies (each considers only one notation). Most works adopt GSN notation, which is indeed very popular in literature (NAIR et al., 2014). TCL, POE, D-Case and a textual notation proposed by (HOLLOWAY, 2008) were cited once each. TCL language is a CAE-like notation supported by the NOR-STA tool. D-

Case is based on GSN with some extensions to have a broader use for dependability cases. POE is a textual notation used to organise problem statements and their decomposition into sub-statements and evidence of solutions. Holloway's notation is one of the five textual templates proposed as alternatives to the graphical notations (HOLLOWAY, 2008). Three studies did not discuss a specific notation (SMS-1, SMS-5, SMS-8).

Table 13 – Structuring techniques of incremental ACD approaches

Notation	Notation type	Studies
GSN	Visual	SMS-2, SMS-6, SMS-9, SMS-10, SMS-11, SMS-13, SMS-14, SMS-15
DOORS / Trace-Line	Visual	SMS-3
POE	Textual	SMS-4
D-Case	Visual	SMS-7
Holloway's notation	Textual	SMS-12
TCL	Visual	SMS-16
Not discussed		SMS-1, SMS-5, SMS-8

Source: The author (2023)

Table 14 shows the classification of the assessment techniques according to the categories proposed by Nair et al. (NAIR et al., 2014). Most studies explicitly indicated the review of argumentation as a Qualitative assessment technique (10 studies – 67%). Four studies did not explicitly discuss any assessment technique. Such studies should address when, how and who would perform human-based reviews of the assurance cases.

Two studies discussed assessment techniques categorised as Checklists. SMS-2 proposed a process framework that prompts developers to review their candidate development choices using a set of criteria. This checking drives the developer to establish confidence in the argument or to improve it. SMS-8 introduced an analysis framework that treats a system and its assurance case as a single entity. The framework comprises a list of statements about the safety properties of the system; as the development progress, the statements serve as a checklist.

Only one study discussed a Logic-based technique. SMS-8 provided a notification mechanism that performs in-production monitoring of the safety properties of the system. The study also proposes an audit phase (Qualitative) in its comprehensive lifecycle, thus including three different assessment techniques. SMS-7 discussed dependability metrics (Quantitative) managed throughout the dependability cases development with the support of the AssureNote tool.

The study also proposed a qualitative review of the argumentation to complement the quantitative assessment. The advancement towards more non-qualitative techniques could foster a more systematic way of incremental development.

Table 14 – Assessment techniques and categories (RQ5).

Technique	Category	Studies
Argument review	Qualitative	SMS-1, SMS-3, SMS-4, SMS-7, SMS-8, SMS-9, SMS-11, SMS-12, SMS-13, SMS-14, SMS-15
Development choice criteria	Checklist	SMS-2
Analysis framework	Checklist	SMS-8
Monitoring mechanism	Logic-based	SMS-8
Dependability metrics	Quantitative	SMS-7
Not discussed		SMS-5, SMS-6, SMS-10, SMS-16

Source: The author (2023)

Three tools appeared in the studies. SMS-7 used the D-Case Editor⁶, a visual tool that supports GSN and D-Case descriptions. SMS-16 used the NOR-STA⁷ tool that is a management system for supporting conformance with norms and standards; it supports assurance case documentation in CAE notation. SMS-11 used the AssureNote⁸, a GSN authoring tool. The other 12 studies did not disclose the tool support used (if any). Lack of tool support is still noticed as a gap between theory and practice (NAIR et al., 2015), although there were found 46 assurance case tools in a recent survey (MAKSIMOV et al., 2018). There is a need to provide more empirical data and discussion towards tools for assurance cases.

3.2.7 Discussion

In this section, we discuss the results found to answer the research questions presented in the previous section.

Development processes are indeed very particular from company to company (even among teams or projects within a company). Thus, when practitioners seek new practices in literature, they have to identify similarities between their environment and those of the research reports. The presentation of novel practices as *Tailoring methods*, as proposed by SMS-16, could better facilitate the industry adoption. The authors proposed a set of practices, each one composed

⁶ https://github.com/d-case/d-case_editor

⁷ <http://www.nor-sta.eu/en>

⁸ <https://github.com/AssureNote/AssureNote>

of objectives, steps, inputs, outputs, and a safety argument template. This eases the process improvement in a stepwise integration of single practices (DIEBOLD; THEOBALD, 2018).

The studies mapped as *Agile* (4, 25%) all prescribed *Blended* and *Scope-driven* development of the AC. This combination is a promising approach, fitting ACD into the agile culture in two ways. First, agile culture emphasises that the whole team is responsible for the deliverables, so blending ACD tasks into the main flow is key to raising team awareness towards safety assurance. Second, the *Scope-driven* development of ACs is aligned with the way agile development produces artefacts such as requirements specification and testing cases, for example. Although no study considers *Parallel* development combined with *Agile*, this could also be achieved with ACD running in a distinct flow and synchronising with the main development flow regularly. We see it as unlikely that *Agile* model could be effectively combined with *Argument-driven* development, as the nature of agile settings is to work on priority scope and to not put up-front effort on future features.

The studies mapped as *V-model* were more diverse in their combination with ACD, but most associated with *Parallel* task integration and *Phased* argument development. This reflects the nature of the V-model, in which the main flow could be accompanied by parallel, supporting flows (RUPARELIA, 2010). However, one study blended the ACD into the main development flow, interleaving ACD tasks with requirements analysis, safety analysis, design and coding SMS-13. This approach may raise more awareness of ACD for the entire team.

The majority (3/4) of the studies mapped as *Incremental* were directly influenced by the *Argument-driven* development approach. In making AC the main driver of the development, the work starts by developing the skeleton of safety arguments, and then proceeding to other software development activities. It is like the SDLC has been merged into the ACD process. It is hard to envision a whole complex product being developed in this manner. Still, it is an interesting approach to emphasise up-front risk/safety analysis and design, and it could be adopted in the development of specific parts of a system.

Our findings indicate that the SDLC model could be a factor that influences the choices of approaches for Task Integration and Argument Development, although it does not constrain them. Also, the specific choice for *Argument-driven* approach could influence the choice of the SDLC model.

The development of a preliminary AC, that break downs key safety strategies for the system under development, is a fundamental part of *Argument-driven* and *Phased* approaches, and it is also identified in *Scope-driven* approaches. This finding indicates that the preliminary AC

should be available as soon as the implementation phase begins. (KELLY, 1999) recommends this initial AC to be produced after the definition and review of the requirements specification. However, this kind of milestone may not apply to SDLC models such as Incremental and Agile. There is no empirical research on how much up-front effort should be put into the development of a preliminary AC.

We found that development-related roles are gaining importance in ACD. Since 2015, there were six (out of nine) studies indicating development personnel for ACD tasks (SMS-8, SMS-9, SMS-11, SMS-12, SMS-14, SMS-15). We also found that most of the studies (4 out of 6) classified as *Scope-driven* indicated development roles for AC building (SMS-5, SMS-9, SMS-12, SMS-15), which was once a task associated only with safety/assurance experts (HATCLIFF et al., 2014). Bridging the software engineering and safety engineering knowledge areas is an open challenge (STEGHÖFER et al., 2019). The direct involvement of developers in producing and reviewing ACs is a promising strategy to bridge software engineering and safety engineering. The *Blended* approach for task integration could be a way to foster team awareness of safety assurance concerns.

Regarding the expected outcomes found, we see a mix of expectations from software and safety perspectives, and the integration of ACD into SDLC is expected to deliver results for both sides. In spite of such expectations, we could not identify, in the studies, concrete discussions towards metrics to evaluate the approaches, neither of a quantitative nor qualitative nature. This makes the comparison of research results unfeasible.

Many studies did not evaluate their proposals in real settings, as they presented illustrative scenarios or a single case study. Process improvements are better evaluated when applied in multiple settings (multiple companies, teams, or projects), as they determine many factors that influence the outcomes of a process. However, evaluations performed in multiple settings are difficult as they would require access to various partners willing to participate in a research project.

Finally, our findings show that agile methods are gaining more adhesion in ACD research. However, none of the agile approaches present in the selected studies was evaluated in the industry context. One possible reason is that although agile adoption in SCS has increased, it is not the main development model used (DIEBOLD; THEOBALD, 2018). Also, we found only six (37.5%) studies in industry contexts, so there is a lack of studies on incremental ACD in the industry.

3.3 SMS THREATS TO VALIDITY

We discuss some threats to the validity of the SMS. We followed the guidelines defined by (KITCHENHAM; CHARTERS, 2007).

Publication bias. We started the SMS without a pre-selected list of venues or journals on the topic of safety-critical development and certification. As we were focusing on a topic that connects two established research areas (software engineering and safety certification), pre-selecting potential venues/journals would result in an extensive initial list of papers. In Table 7, we indicate the most important venues in these two research areas covered by the database engines used. To avoid such bias, we decided to perform an automatic search of studies, complemented by a snowballing procedure. The snowballing helps mitigate the risk of missing relevant studies that are not indexed, and the procedure found two studies from a non-indexed conference and an open journal.

Search string. We seek to avoid missing studies that have as secondary the contribution towards incremental ACD. Thus, we decided to build the query with a focus on general terms from software development and AC. We constructed the search string iteratively by performing various pilot searches. The duality of research areas also impacted the construction of the search string. We decided not to include in the search string words related to “incremental”, “integration”, “together”, “-driven” or “-based”. In some key studies used as the control sample to develop the search string, these terms do not appear in the title or abstract (SMS-2, SMS-4). Although this decision increases the number of retrieved articles, we avoided missing studies that have as secondary the contribution towards incremental ACD. Table 15 describes how integration between ACD and SDLC was referred to in all the selected studies.

Selection of primary studies. Only the first author performed the selection and extraction, whilst all authors revised the extraction. As there is the risk of missing relevant studies, we sought to design explicit inclusion and exclusion criteria, and we adopted the screening of full text for the exclusion criteria. In fact, few studies indicated in the title or abstract a focus on the subject of this mapping study, and some studies do not address the incremental ACD as their primary contribution.

Table 15 – Terms used to denote development of ACs alongside the SDLC.

Study	Title or/and abstract excerpts
SMS-1	“common evolutionary process”
SMS-2	“Assurance Based Development”
	“synergistic construction”
	“Co-developing the system and its assurance case”
SMS-3	“Evidence-Based Development”
	“approach to progressive system assurance (...) integrated with the development process”
SMS-4	“Assurance Driven Design”
	“how software and assurance argument can be built together”
SMS-5	“evidence sources that can be used at the various stages of the lifecycle”
SMS-6	“develop both a software system and a safety argument iteratively”
SMS-7	“promote dependability through the lifecycle”
SMS-8	“safety case becomes the focus of safety engineering throughout the system lifecycle”
SMS-9	“how an assurance case can be generated when a software process is employed”
SMS-10	“Safety Case Driven Development”
SMS-11	“Continuously Revised Assurance Cases”
	“developing assurance cases (...) at every stage of the system lifecycle”
SMS-12	“certification process to be done in parallel with development”
SMS-13	“Evidence-oriented V-model Methodology”
	“seamless and unified development process”
SMS-14	“assurance drives the engineering process”
SMS-15	“incrementally applied safety-related activities”
SMS-16	“incorporating agile practices into critical software development”

Source: The author

3.4 CHAPTER SUMMARY

The integration of safety engineering and software engineering is an ongoing research area. In this context, ACs are a trending approach for the certification of systems in some regulated domains. The incremental development of ACs alongside the SDLC has been a long-term topic of interest for researchers and practitioners. This chapter presented a Systematic Mapping Study on approaches that promote incremental development of ACs throughout the software development process. We identified 16 papers dating from 1999 to 2018, showing that the topic has kept the interest of researchers since the establishment of ACD as a research area.

We found a variety of ways the integration of ACD and SDLC could occur. Six different SDLC models were identified: Agile, V-model, Incremental, Spiral, Unified and Waterfall.

We identified that the SDLC model adopted could influence how ACD tasks and software development tasks are integrated.

We identified many studies in which developers play an active role in AC construction (SMS-2, SMS-4, SMS-5, SMS-7, SMS-8, SMS-9, SMS-11, SMS-12, SMS-14, SMS-15) and also in AC assessment (SMS-2, SMS-5, SMS-11, SMS-12, SMS-15). This is an interesting finding, as such involvement can help close the gap between software and safety areas. With regard to expected outcomes, the two most identified were the enhancement of project management (e.g., reducing cost) and the increase of quality of safety argumentation.

In future work, we propose further investigation on how practitioners are integrating ACD into the SDLC, and which factors drive their choices of SDLC models, Task Integration and Argument Development approaches. In addition, we see a need for more empirical evidence to map out the adequate profile characteristics of such professionals, as well as their required capabilities and previous education. Also, future studies should discuss how much safety accountability is to be assigned to each role involved in safety-critical development and certification.

The lack of case studies with multi-case designs and multiple units of analysis indicates that the topic still needs more empirical studies to evaluate assumptions and insights stated by researchers and gathered from practitioners. Also, surveys could be used to gather qualitative data from practitioners regarding their perceptions of the usefulness and ease of use of the approaches.

4 REQUIREMENTS ENGINEERING AND ASSURANCE CASE DEVELOPMENT INTERPLAY

In this research step, we reach out to practitioners to understand how Requirements Engineering (RE) and Assurance Case Development (ACD) activities currently interplay. As previous research discusses in Chapter 2 and our mapping study presented in Chapter 3 suggest, there are potential benefits in integrating ACD into the development process instead of handling it disconnectedly.

In this survey, we focus on the requirements activities and investigate how the integration of ACD and RE activities occur in practice and how practitioners perceive its benefits in SCS development. What is the current practice regarding Assurance Case Development? What are the current practices towards the integrated development of Assurance Case and Requirements? What is the view of practitioners on integrating Requirements Engineering and Assurance Case Development processes? This empirical study aims to answer these and other questions that the current literature could not fully answer. Therefore, we designed and executed a survey to draw the current state of practice regarding the interrelation between RE and ACD. Moreover, we aim to gather practitioners' views on this interrelation.

This survey will leverage how a vital process area, RE, interplays with ACD, providing empirical evidence to support the formulation of proposals for the integrated development of AC during RE activities in SCS development projects. We discuss related works to this survey in Section 2.4.2.

The remainder of this chapter is structured as follows. In Section 4.1, we present the research questions and methodology. Our results and analysis are discussed in Section 4.2 for each research question. In Section 4.4, we discuss the threats to the validity of this survey. Finally, in Section 4.5, the conclusions are presented.

4.1 SURVEY METHODOLOGY

4.1.1 Research Questions

In this survey, we seek to answer the following research questions:

Survey-RQ1. What is the current practice regarding Assurance Case Development?

We seek to characterise the experience of our participants concerning ACD. We currently

lack empirical evidence on the motivation to develop ACs, the roles involved in ACD, how ACD integrates to SDLC, and the structure of AC reports concerning assurance argument.

Survey-RQ2. What are the current practices towards the integrated development of Assurance Case and Requirements?

We seek to understand if there is any integration of ACD and RE activities. The goal is to determine if ACs are consulted during RE activities, if safety assurance is a concern addressed during RE activities and if there is cross-collaboration between RE and safety professionals.

Survey-RQ3. What are the views of practitioners on the integration of Requirements Engineering and Assurance Case Development processes?

We aim to gather practitioners' opinions on the feasibility and benefits of integrating RE and ACD. In this question, we want to collect the views of participation in the following aspects: perceived benefits of the cross-collaboration, if the development of AC during RE is potentially a good practice, if the integration could mitigate recurring problems in SCS, and which would be the RE activities appropriate for that integration.

4.1.2 Survey Design

Our methodology followed the guidelines from Kasunic (2005), Linäker et al. (2015), Molléri, Petersen and Mendes (2020) and relied on the reuse of previous experiences of the authors. We started by establishing the objectives of our survey based on the gaps identified in the literature discussed in Sects. 1 and 2. The objectives and research questions are presented in Sects. 1 and 3, respectively.

Our survey is descriptive and collected evidence regarding the state of the practice in the interplay of RE and ACD. It is a cross-sectional survey, as data is collected at one point in time. It was self-administered through a web-based questionnaire for the following reasons. First, if we only considered practitioners from our close professional network to ask for interviews, we may not reach a broader number of participants. Also, an online questionnaire is a good instrument for those seeking anonymity. This choice impacts the types of questions used and the length of the instrument.

We used the LimeSurvey tool to implement our questionnaire. This mature web survey platform provides a user-friendly interface and built-in features for questionnaire structuring, response collection/analysis, and anonymisation.

4.1.3 Pilot Application

We conducted a pilot study with representative practitioners with experience in both fields (RE and ACD). We invited six practitioners, pointing them to a landing page and the web questionnaire. The landing page clarified the scope of our survey and the expected profile of respondents. Four practitioners agreed to participate and answered the questionnaire. One is a researcher with experience in both fields and participation in long-term research and development projects in the area of assurance and certification. Another one has experience in research and industry consulting. The third participant has long experience with the consulting sector in the field of safety-critical software development and certification. Finally, our fourth participant is a software engineering director working with SCS development.

We asked the pilot participants to review the survey's questionnaire (by themselves without any mentoring) and to provide open-ended feedback about the following issues:

- Usability, structure, and length;
- Inappropriate/intrusive/unethical questions;
- Comprehensibility of questions and answer options; and
- Any additional comments.

Based on their feedback, we corrected some issues, such as improving the comprehension of questions/options, adding answer options, and improving web form navigability. We wrote the questions to be self-contained so that respondents may not need further explanations or supporting material. None of the pilot participants indicated the need for supporting information.

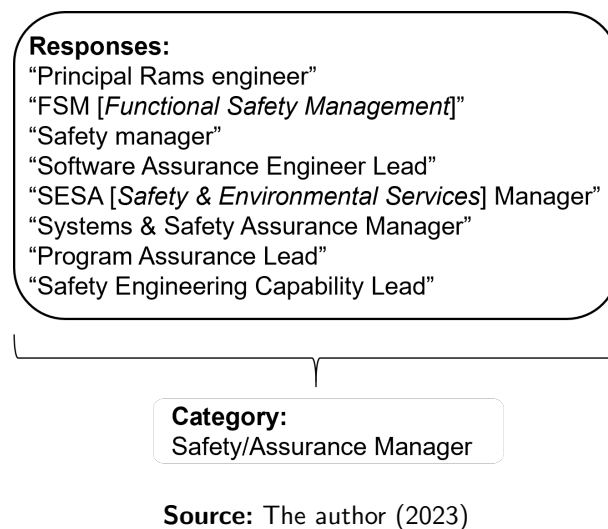
4.1.4 Survey Instrument

We defined a set of 21 survey questions to address the research questions and profile the respondents' experience and practice. Table 16 shows the survey questions, and in the Appendix, the complete instrument is presented. For questions related to RQ1 and RQ2, we explicitly asked the participants to consider their experience in the past five years.

Many questions are multiple-answer questions to allow the participants to choose more than one answer. Note here that some of the multiple-answer questions allow participants to include

information that is not included in the answer options (Other option). Thus, participants may provide answers not envisioned in the survey design and preliminary evaluation with this alternative option. Also, some questions offered participants the opportunity to provide open-ended responses. We manually coded the answers using a data-driven approach to analyse these answers, in which categories are constructed a posteriori based on terms found in the answers' texts (POPPING, 2015). One author coded the answers, and the other revised and agreed on the most accurate interpretation of the open-ended responses (see an example in Figure 10).

Figure 10 – Coding example from analysis of Survey Q3



Regarding questions that aim to identify practices currently in use (RQ1 and RQ2), we decided to use frequency scales to compare frequencies among the practices. We were interested in identifying the most used practice for some questions, so we explicitly asked respondents to indicate the most used alternatives. Similarly, we used agreement scales in questions asking for participants' opinions (RQ3). Frequency and agreement scales are ordinal response scales that are well suited to determine the frequency of behaviour and the intensity of a belief (KASUNIC, 2005).

4.1.5 Survey Sampling and Application

This survey's target population was researchers and professionals involved in developing critical systems and with any experience/knowledge about RE and ACD. As ACD is still an emergent approach, the respondent's target population is relatively narrow. We decided

Table 16 – Survey questions

Id	Survey RQ	Question
Q1	Profile	How did you find this survey?
Q2	Profile	How long have you been working on the development of safety-critical systems?
Q3	Profile	What is your current main role in the organisation?
Q4	Profile	How many projects that involved ACs for safety-critical systems have you participated in?
Q5	Profile	In which country(ies) have you principally worked upon AC or RE for SCS?
Q6	Profile	What is the organisation's main role for which you have worked regarding the development of SCS?
Q7	RQ1	Is the development of Assurance Cases a mandatory, recommended, or optional activity?
Q8	RQ1	How often have you been involved in the following ACD activities?
Q9	RQ1	Which of the following statements apply to how ACs are developed in your projects?
Q10	RQ1	Which is the main notation used to construct the Assurance Arguments?
Q11	RQ1	Which of the following scenarios best describes how AC Reports are organised, considering only one system to be certified?
Q12	RQ1	Is there any kind of automation in the creation or maintenance of Assurance Arguments and Reports?
Q13	RQ2	Considering the management of traceability among requirements and safety analysis information, which of the following scenarios best describes the most used practice in your projects?
Q14	RQ2	How often are Assurance Cases consulted during Requirements Engineering activities?
Q15	RQ2	How often does your team consider safety assurance when performing Requirements Engineering activities?
Q16	RQ2	Which of the following collaborations between RE and safety specialists occur, and in which frequency?
Q17	RQ3	What are the benefits of RE engineers' participation in ACD or safety analysis development activities?
Q18	RQ3	What are the benefits of Safety Engineers' participation in RE activities?
Q19	RQ3	How much do you agree with the development of ACs during RE activities?
Q20	RQ3	To what extent do you believe the development of ACs during RE activities could mitigate the following problems in SCS development?
Q21	RQ3	Which RE activities do you think the development or sketching of ACs could be combined with?

Source: The author (2023)

to include researchers because they perform services similar to consulting in the industry. Also, researchers and consultants most likely work with multiple companies, so they bring observations and opinions that serve this study purpose to be an exploratory view on the topic. Many questions were designed to capture observed practices in SCS development that could be answered by external roles participating in a project. Finally, we think the aggregation of the opinions of researchers (RQ3) is as important as the opinions of practitioners in such an emergent topic of study.

The unit of analysis for the survey sample is the practitioner who works on projects that perform RE and ACD activities. The survey participation is individual and anonymous. We did not record any personal data of the participants. We chose a non-probabilistic sampling, as the actual population of practitioners involved in RE and ACD is unknown. Non-probabilistic sampling refers to any approach in which participants are not randomly selected (LINÄKER et al., 2015). We constituted the sampling by convenience, through our personal contacts, and by publicising our survey on online social groups/lists. As ACD is still an emergent approach, the respondent's target population is relatively narrow.

We distributed our survey through advertisements, invitations, and snowballing. We publicised our survey to five groups on LinkedIn related to safety/assurance cases, safety-critical systems development, and requirements engineering. We directly invited practitioners and researchers from our personal networking (+10 invitations) and actively searched in professional social networks such as LinkedIn (+100) and Research Gate (+40).

We also invited authors of papers in selected venues (+80). We searched for articles published in these venues in the last five years, including requirements engineering and assurance cases. The choice of venues and search criteria is not meant to be exhaustive, yet it comprises the authors from the academy and industry who publish works in some of the most suitable venues to date. The chosen venues were the following:

- International Conference on Computer Safety, Reliability and Security (SAFECOMP) and associated workshops;
- International System Safety Conference (ISSC);
- International Conference on Dependable Systems and Networks (DSN);
- Requirements Engineering Conference (RE);
- Reliability Engineering and System Safety Journal (RESS);

- Journal of Reliable Intelligent Environments (JRIE);
- Requirements Engineering Journal (REJ).

Last, we asked participants to recommend other potential participants. We received four recommendations from practitioners, which were invited by email. We cared to send each invitation in a separate message and mention why they had been selected (indication or curriculum). Direct and open invitation messages comprised a brief explanation of the survey objectives and estimated time to complete.

Our survey was available online between May 2020 and November 2020. Our survey has received 31 different responses. Forty-one (41) responses were initiated but not submitted. We discarded these incomplete responses. After the survey was closed, we extracted the responses. We also performed the descriptive analysis and manual coding of free-text answers using a spreadsheet tool (descriptive statistics and manual coding) and R language code (Likert scale charts).

4.2 SURVEY RESULTS AND ANALYSIS

In this section, we present the data gathered for each RQ. We provide more detailed information on the instrument used in the supplementary material (Appendix B).

4.2.1 How did you find this survey? (Q1)

The sharing method with more responses was Invitation via Email (39%), followed by Invitation via LinkedIn (26%) and Post on a LinkedIn group (26%). Post on Mailing List comprises 6% of returned responses. Post on Research Gate comprises only 3% of returned responses. Thus, the direct, individual invitation provided a better return of responders.

4.2.2 How long have you been working on the development of safety-critical systems? (Q2)

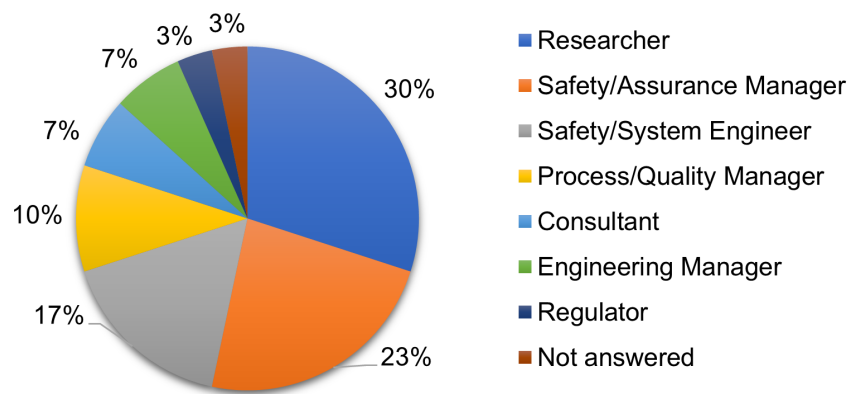
Most participants (78%) have 10+ years of SCS development experience, and 16% of the participants have 6 to 10 years. Participants with three to five years of experience were 3%, and those with one to two years were 3%. As SCS tends to be a long-term endeavour,

professionals with few years of experience are likely to have not yet walked through many phases of the SCS development lifecycle. Our sample contains many participants that most likely have experienced various phases of SCS development, although we have not directly asked them about this.

4.2.3 What is your current main role in the organisation? (Q3)

Participants provided open-text answers, coded into categories of roles (using the method described in Section 4.1.2). We identified seven category roles, presented in Figure 11.

Figure 11 – Category roles in the organisation



Source: The author (2023)

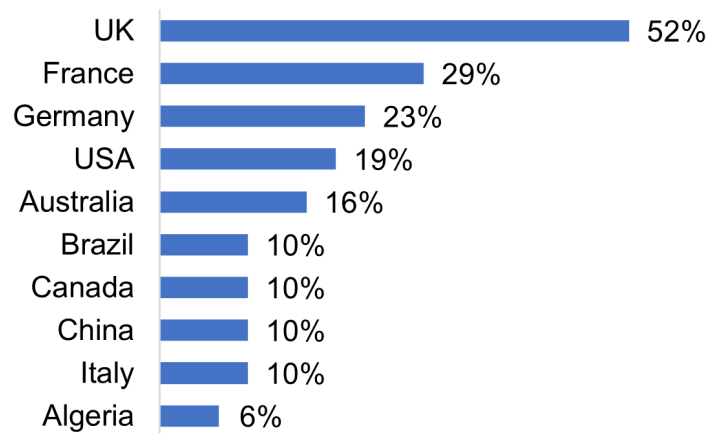
4.2.4 How many projects that involved Assurance Cases for safety-critical systems have you participated in? (Q4)

This question is intended to identify how many different projects the participants have worked on with ACs. The way ACs are organised and managed may vary between projects and domains. The number of projects indirectly indicates diversity in the participants' experience. 39% of the participants have worked with ACs in more than ten projects, 29% in five to ten projects, and 32% in fewer than five projects.

4.2.5 In which country(ies) have you principally worked upon Assurance Case or RE for safety-critical systems? (Q5)

We seek to identify countries in which the participants work. Figure 12 shows the most indicated countries. In total, participants indicated 20 different countries. Some countries were indicated only once each (3% each): Argentina, Croatia, Greece, Poland, Singapore, Sweden, Bulgaria, Qatar, Denmark, and Ukraine.

Figure 12 – Countries in which participants have worked



Source: The author (2023)

4.2.6 What is the main role of the organisation for which you have worked regarding the development of safety-critical systems? (Q6)

The main roles of the organisations in which participants worked were the following: Consultancy (32%), Developer/manufacturer of final systems (29%), Research institutions (23%), Certification Authority (7%), Component supplier (3%), Main system integrator (3%) and Systems engineering management (3%).

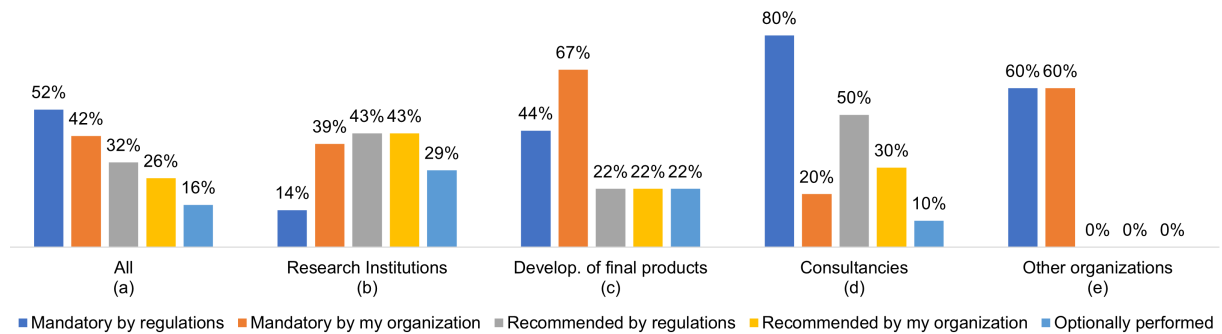
4.2.7 Is the development of Assurance Cases a mandatory, recommended, or optional activity? (Q7)

We aim to identify the nature of the need for AC development. It is a multiple-choice question because a single participant may work on projects that develop AC because of different motivations. The overall distribution is depicted in Figure 13–a.

The distribution varied if grouped by the role of the organisation of the participants (Section 4.2.6). Practitioners at Developer/manufacturer of final systems organisations (Figure 13–c) have more frequently addressed AC in a mandatory setting, whereas those at Research institutions (Figure 13–b) face ACD as a recommended or optional practice. This difference may be explained as researchers tend to be involved in projects experimenting on new technologies or practices that could later be incorporated into organisations policies and standards.

Consultancy organisations (Figure 13–d) mainly address ACD in settings driven by regulations (as mandatory or recommended), which could be explained by the fact that consultancy services are more focused on bringing the organisations the external expertise towards compliance with regulations.

Figure 13 – Nature of need for AC development for all participants and grouped by role of organisation



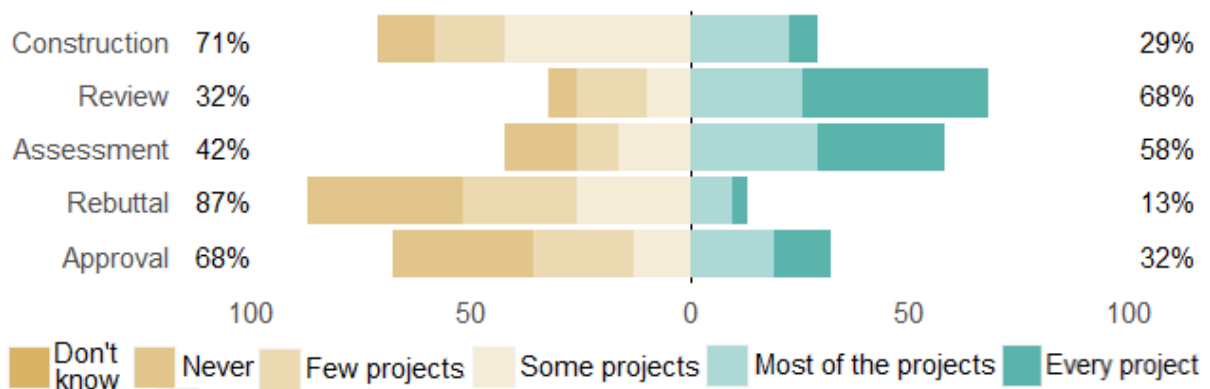
Source: The author (2023)

4.2.8 How often have you been involved in these following Assurance Case Development activities? (Q8)

We aimed to identify the ACD activities the participants are involved in and in which frequency. As Figure 14 shows, our participants are more frequently involved in Review and Assessment activities. The Rebuttal activity has few references in the literature, and it was also the least indicated activity.

We found differences in the participation frequencies among practitioners with different expertise in projects with AC (Section 4.2.4). Table 17 shows the frequencies of participation in ACD activities grouped by the expertise in projects using AC. More experienced professionals participate more in Review and Assessment activities than lesser ones. The most experienced engage more in Approval and Rebuttal activities while keeping frequent participation in all activities.

Figure 14 – Frequency of participation in ACD activities for all participants



Source: The author (2023)

Table 17 – Frequency of participation in ACD activities grouped by experience in ACD

ACD Activity	All	<5 projects	5 to 10 projects	>10 projects
Construction	29%	10%	56%	25%
Review	68%	20%	89%	92%
Assessment	58%	10%	89%	75%
Rebuttal	13%	0%	11%	25%
Approval	32%	0%	56%	42%

Sum of frequencies for “Most of the projects” and “Every project”

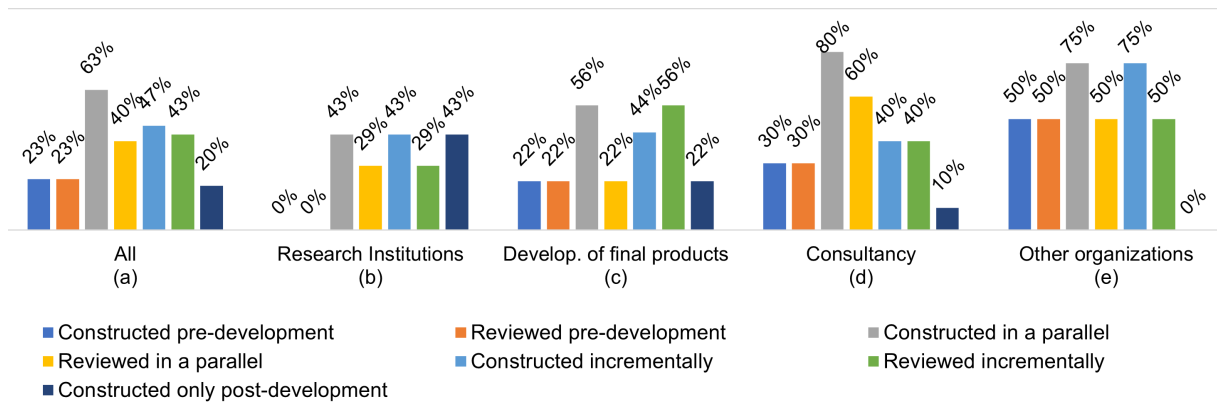
Source: The author (2023)

4.2.9 Which of the following statements apply to how Assurance Cases (ACs) are developed in your projects? (Q9)

We aimed to identify how the development of ACs integrated with the SDLC, so we sought to identify which practices are most common. Participants could indicate multiple practices, as some of them are not mutually exclusive and because projects may adopt different methods. One participant chose not to answer this question. The distribution based on 30 responses is presented in Figure 15–a.

The most common practice is the construction of ACs in a parallel process during the development (63%). The practice of reviewing ACs in a parallel process was reported by 40% of the participants. Incremental construction (47%) and review (43%) are more common than pre-development construction (23%) and review (23%) and also more common than ACs constructed only post-development (20%). We did not find significant variance in the frequency of practices among the different roles of organisations (Figure 15–b/c/d/e).

Figure 15 – ACD practices observed by all participants and grouped by the organisation's role



Source: The author (2023)

4.2.10 Which is the main notation used to construct the Assurance Arguments? (Q10)

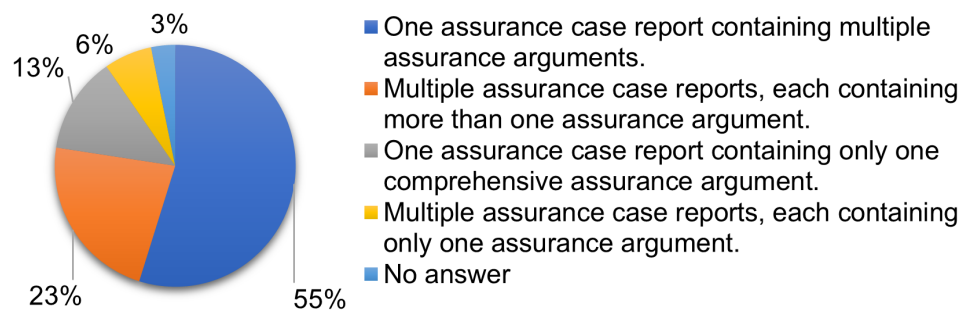
In a state-of-practice survey, Nair et al. (2015) identified the techniques to structure evidence in practice. In our survey focused on ACs, we break the questions into two aspects: notation used for argument construction (this question, Q10) and organisation of the AC artefacts (question Q11). We asked participants to inform the most used practice for each aspect.

The primary notation used to structure the assurance arguments is GSN (52%), followed by Structured textual (16%) and Textual/plain text (10%). Other notations received one mention (3% each): ASCAD, Bowtie, CAE, Claims table, SACM, Tabular, and Toulmin.

4.2.11 Which of the following scenarios best describes how Assurance Case Reports are organised, considering only one system to be certified? (Q11)

We asked participants about the most common scenario for AC report organisation. Figure 16 shows the distribution. The use of multiple assurance arguments to argue for a single system is the most used practice, either using a single AC report (55%) or various AC reports (23%). The use of one comprehensive assurance argument is less frequent, either in one assurance case report (13%) or split into multiple reports (6%).

Figure 16 – AC report organisation



Source: The author (2023)

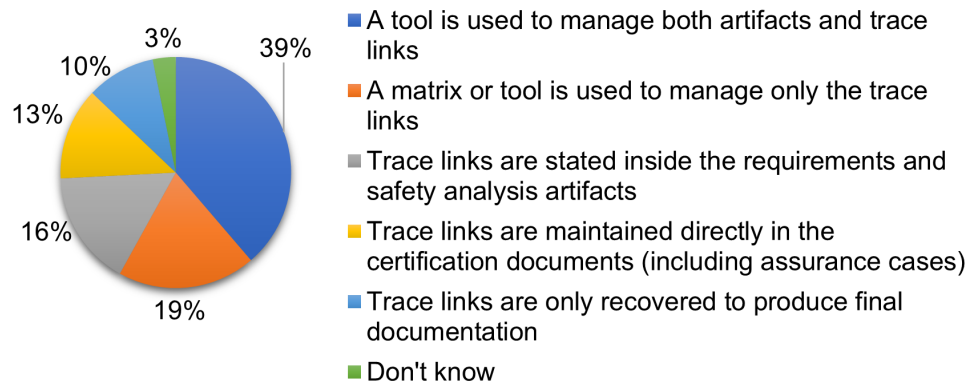
4.2.12 Is there any kind of automation in the creation or maintenance of Assurance Arguments and Reports? (Q12)

We seek to identify the level of automation in creating or maintaining ACs. If there is any automation, the participant was asked to briefly describe it using free text. Only eight participants answered, but two just indicated there is no automation. Four of them stated the existence of reporting features in the toolchain used. One participant noted using a tracing tool to assess the completeness of requirements involved in ACs. Another indicated only the tools used (SAP and Oracle Business) without describing the kind of automation provided by the tools. The six participants that indicated the existence of automation all have experience in five or more projects using AC.

4.2.13 Considering the management of traceability among requirements and safety analysis information, which of the following scenarios best describes the most used practice in your projects? (Q13)

We seek to identify the most used practices for traceability management of requirements and safety analysis information. Figure 17 shows the distribution. The good practice of managing artefacts and trace links in a single tool is the most common scenario (39%). However, the post-development recovery of links (10%) and the use of certification artefacts to hold traceability information (13%) are observed.

Figure 17 – Traceability management scenarios

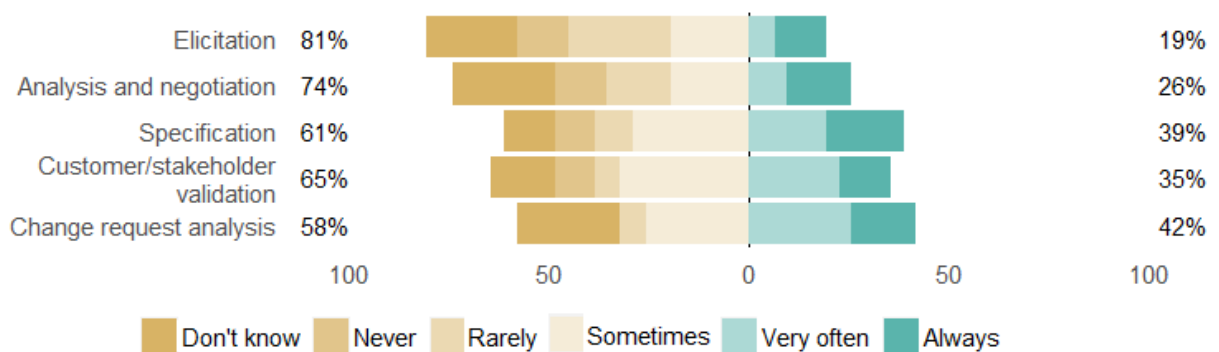


Source: The author (2023)

4.2.14 How often are Assurance Cases consulted during Requirements Engineering activities? (Q14)

As Figure 18 shows, ACs are artefacts regularly consulted during RE activities. In the initial activities of Elicitation and Analysis and negotiation, they are consulted less. In later activities, such as Specification and Change request analysis, ACs are consulted more often. Twenty-seven participants (87%) reported for at least one activity the frequency Sometimes or above. Table 18 shows the frequencies grouped by the role of the organisation of the participants (Section 4.2.6). Professionals from research institutions observed less frequent use of ACs in RE activities, while those from consultancies and other organisations observed more regularly.

Figure 18 – Frequency of which ACs are consulted during RE activities for all participants



Source: The author (2023)

Table 18 – Frequency in which ACs are consulted during RE activities for all participants and grouped by type of organisation

RE activity	All	RI	DFP	C	O
Elicitation	19%	14%	22%	10%	40%
Analysis and negotiation	26%	0%	33%	30%	40%
Specification	39%	14%	33%	50%	60%
Customer/stakeholder validation	35%	14%	33%	40%	60%
Change request analysis	42%	29%	44%	30%	80%

RI = Research institutions, DFP = Developers of final products, C = Consultancies, O = Other organisations
Sum of frequencies for “Very often” and “Always”

Source: The author (2023)

4.2.15 How often does your team consider safety assurance when performing Requirements Engineering activities? (Q15)

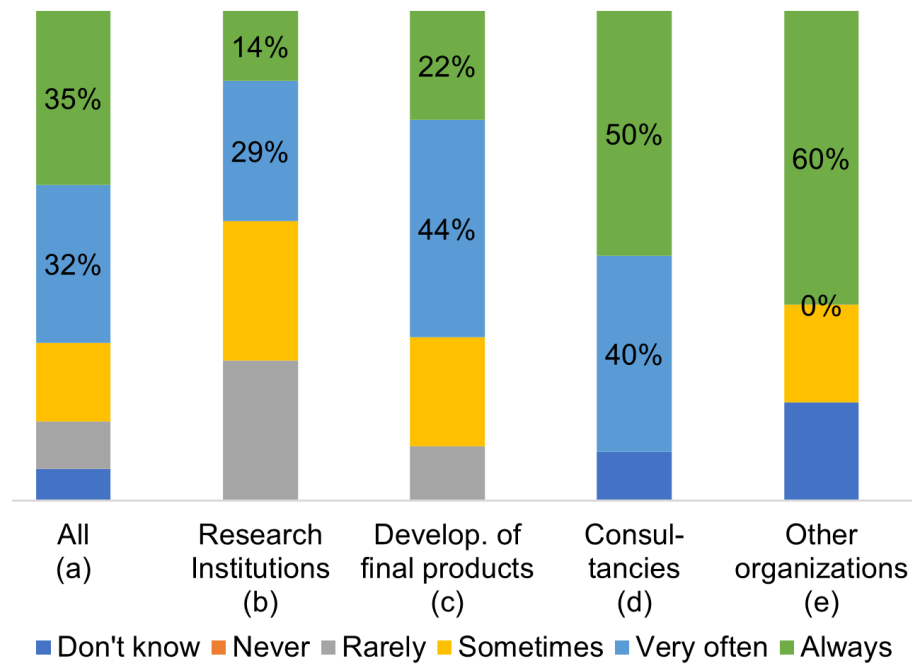
As Figure 19–a shows, most participants regularly consider safety assurance during RE activities (Always = 35% or Very often = 32%). Of them, 16% indicated Sometimes, and 10% indicated Rarely. Few participants reported Don’t know (3%), and none reported Never. Professionals from Research Institutions (Figure 19–b) observed lower frequencies of this practice, while those from Consultancies (Figure 19–d) observed higher frequencies compared with the entire sample.

4.2.16 Which of the following collaborations between requirements and safety specialists occur, and in which frequency? (Q16)

We chose to explore a variety of different scenarios of cross-collaboration. We presented participants with some scenarios and asked them how often they observed each project’s scenario. The scenarios and frequencies are depicted in Figure 20. Our results show that safety engineers frequently engage in RE activities (Every or Most of the projects = 58%) and artefacts review (Every or Most of the projects = 58%). Less frequently, Requirements engineers participate in ACD activities (Every or Most of the projects = 48%), in safety analysis (Every or Most of the projects = 35%), and review safety engineering artefacts (Every or Most of the projects = 29%). However, the level of frequency in which requirements engineers are engaged in safety-related activities is impressive.

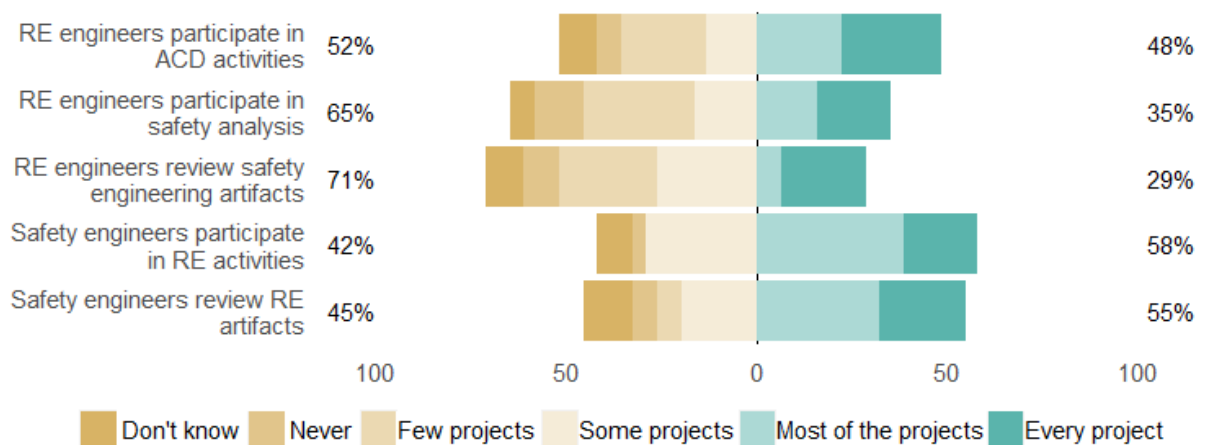
Table 19 shows the frequencies of collaborations grouped by the role of the organisation of

Figure 19 – Safety assurance consideration during RE activities for all participants and grouped by the organisation's role



Source: The author (2023)

Figure 20 – Collaboration scenarios and the frequency of occurrence for all participants



Source: The author (2023)

the participants (Section 4.2.6). The distribution by groups is similar, except for professionals from Research Institutions. It may be explained by the fact that researchers may not be involved in all kinds of activities.

Table 19 – Collaboration scenarios and the frequency of occurrence grouped by the organisation's role

Collaboration scenario	All	RI	DFP	C	O
RE engineers participate in ACD activities	48%	14%	56%	50%	80%
RE engineers participate in safety analysis	35%	0%	44%	40%	60%
RE engineers review safety engineering artefacts	29%	0%	33%	30%	60%
Safety engineers participate in RE activities	58%	29%	67%	60%	60%
Safety engineers review RE artefacts	55%	29%	67%	60%	60%

RI = Research institutions, DFP = Developers of final products, C = Consultancies, O = Other organisations
Sum of frequencies for "Most of the projects" and "Every project"

Source: The author (2023)

4.2.17 What are the benefits of Requirements Engineers' participation in Assurance Case Development or safety analysis development activities? (Q17)

We asked participants to provide the perceived benefits of the collaboration of requirements engineers in safety-related activities and reviewing safety artefacts and assurance cases.

The most indicated benefit (7 participants, 23%) was related to safety concerns awareness. In the participants' opinion, this collaboration fosters the balance between safety and non-safety requirements, as requirements engineers will better understand safety concerns and trade-off needs.

Another indicated benefit was related to traceability (6 participants, 19%) in two perspectives. First, requirements engineers have an early opportunity to review the traces between requirements and the other safety artefacts. Second, they act as knowledge sources in the process of elaborating safety validation plans and the development of ACs.

Raising awareness and shared review of requirements and safety concerns was indicated by four participants (13%). Three participants (10%) pointed out that this collaboration helps the development of reasonable, achievable, and correctly specified requirements. The anticipation of errors and conflicts, and the cost reduction associated with it, were indicated as benefits by three participants (10%).

4.2.18 What are the benefits of Safety Engineers' participation in Requirements Engineering activities? (Q18)

Participants were also asked to provide the perceived benefits for safety engineer collaboration in requirements-related activities and review of requirements artefacts.

The most indicated benefit (8 participants, 26%) was that safety engineers engaging in the RE process support the early consideration of safety concerns during requirements development rather than afterwards. As safety requirements drive many decisions in the concept of an SCS, the collaboration of safety engineers in RE is key.

Five participants (16%) highlighted that this cross-collaboration helps integrate the people and work from safety and requirements engineering areas.

Five participants (16%) pointed out that this collaboration helps to achieve viable requirements that are correctly specified and possible to be met (and feasible to be shown to have met the safety needs). Another benefit indicated is to raise the understanding of safety engineers on customer and product needs (4, 13%). Finally, this collaboration can help review the traceability of requirements and safety artefacts (3, 10%).

4.2.19 How much do you agree with the development of Assurance Cases during Requirements Engineering activities? (Q19)

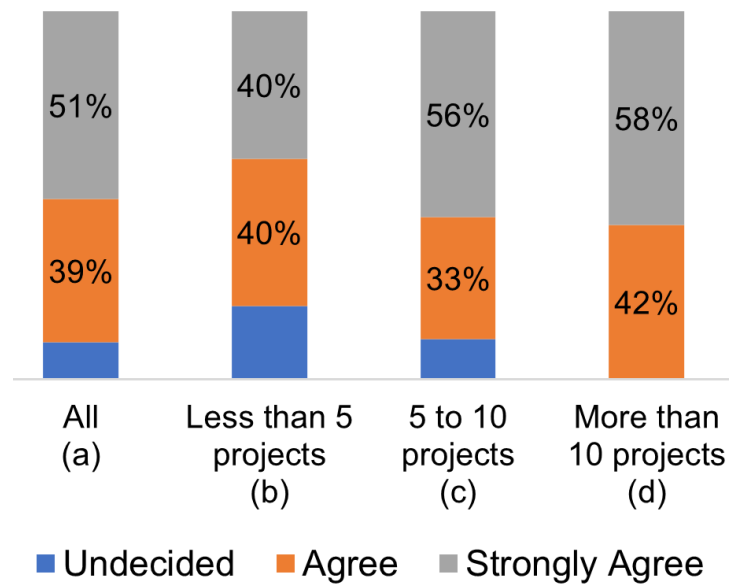
We also asked their opinion on the development of Assurance Cases during RE (Figure 21–a). None reported Disagree or Strongly Disagree. We found that professionals with more experience in ACD are less undecided and more confident that ACD can be integrated with RE activities (Figure 21–b/c/d).

4.2.20 To what extent do you believe the development of Assurance Cases during Requirements Engineering activities could mitigate the following problems in SCS development? (Q20)

We asked the participants to indicate to what extent some recurrent problems in ACD could be mitigated by integrating ACD into RE activities. We discuss the problems reported in the literature in Section 2.1.1.

Figure 22–a shows the five common problems and the distribution of participants' opinions.

Figure 21 – Opinion on ACD during RE activities for all participants and grouped by experience in ACD



Source: The author (2023)

We considered the answer *To a Great Extent* as a positive indication of the participant on the potential impact of the integration in mitigating the problems; the other options were *Somewhat*, *Very Little*, *Not at All*, and *Undecided*.

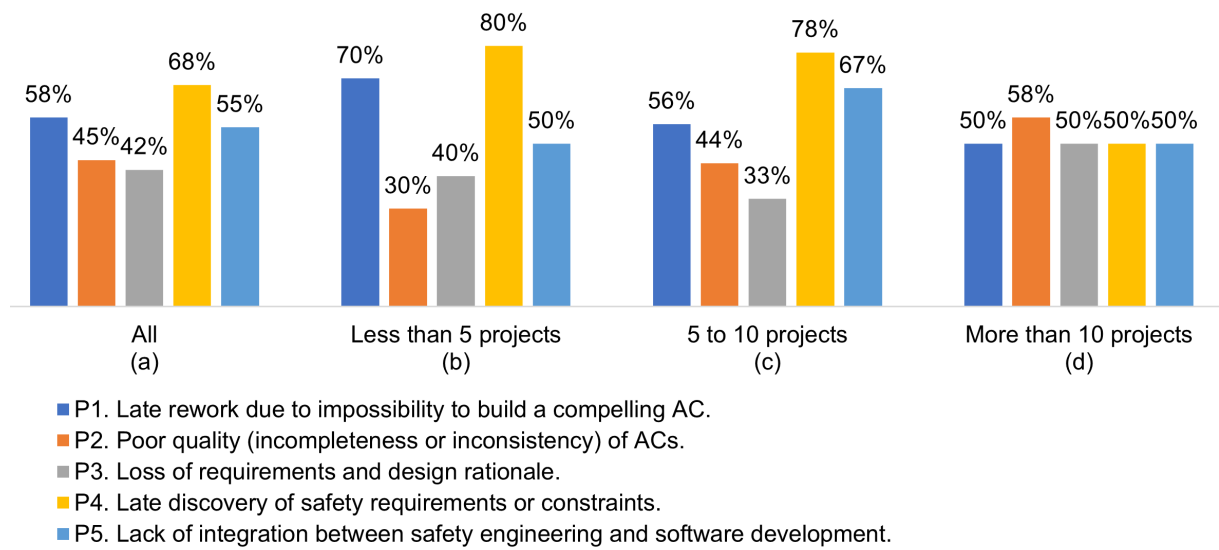
All problems received positive indications from at least 40% of the participants. P4 problem (68%) was pointed as the one that could most benefit from the development of ACs integrated with RE activities. Problems P1 (58%) and P5 (55%) also indicated from the majority that they could be mitigated. On the other side, the problems P2 (45%) or P3 (42%) were not regarded as extensively impacted by the integration of ACD and RE.

As Figure 22–b/c/d show, for problems P1 and P4, the less experienced, the more optimistic the participant is about the impact of integration. On the contrary, for problems P2 and P3, the most experienced group is the most confident that the integration could mitigate these problems.

4.2.21 In your opinion, with which Requirements Engineering activities do you think the development or sketching of Assurance Cases could be combined? (Q21)

Finally, we aimed to gather opinions on which RE activities would benefit more from the integration of AC development. We asked the participants to indicate to what extent some

Figure 22 – Common problems in SCS development that could benefit from RE and ACD integration

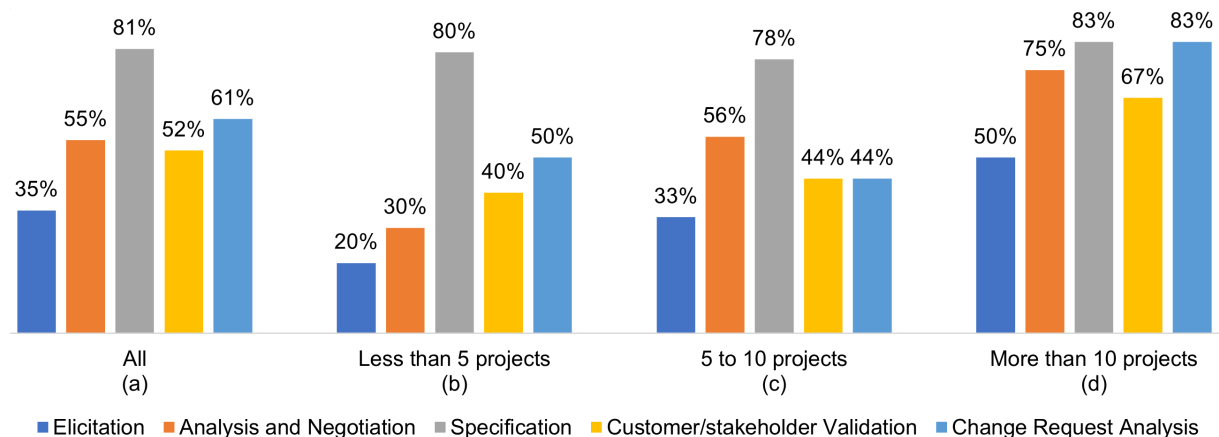


Source: The author (2023)

recurrent problems could be mitigated by integrating ACD into RE activities. The list of problems presented to the participants was identified through a literature review. We present and discuss the problems in Section 2.1.1.

As Figure 23–a shows, most participants (81%) agree the Specification activity is appropriate to be combined with ACD. Except for the Elicitation activity (35%), the other RE activities all received indications by at least half of the participants. Figure 23–b/c/d show the opinions grouped by experience in ACD. We found that, for all RE activities, the confidence in the benefit of integrating ACD and RE grows among the most experienced.

Figure 23 – Opinion on the integration of ACD with RE activities for all participants and grouped by experience in ACD



Source: The author (2023)

In Table 20, we compare the results of this question with the state of practice addressed

by question Q14 (Section 4.2.14). We see that the gap between participants' current practice and their opinions is higher for all activities.

Table 20 – Comparison of state of practice and opinion on the integration of RE and ACD

RE activity	State of practice ^a	Opinion on integration ^b	Diff.
Elicitation	19%	35%	+16%
Analysis and negotiation	26%	55%	+29%
Specification	39%	81%	+42%
Customer/stakeholder validation	35%	52%	+17%
Change request analysis	42%	61%	+19%

^aSum of responses "Very often" and "Always" for Q14, ^bResponses for Q21

Source: The author (2023)

Optionally, we asked participants to provide a comment for each activity they indicated as suitable for RE/ACD integration. One participant pointed out that the integration fosters the "*early identification and agreement of potential safety impact*" for the Analysis and negotiation activity. Another noted that it helps "*capture specific risk considerations and its implications*". Two participants indicated the integration as a way to link requirements analysis to "*safety reliability analysis*" and "*hazard analysis*", respectively. Also, a participant pointed out that the integration of activities may allow for considering the design options while considering the strategies for system assurance.

For the Specification activity, one participant indicated the ACD as a way to reach an "*understanding of what evidence is needed to show the specification is met*". Two participants pointed out the benefit of discussing the "*traceability between system layers level*" and "*link hazard analysis to requirements*", while the other regarded the integration as a helper to "*specifying an architecture that is assurable*".

For the Validation activity, one participant stated that the AC development helps to "*show that the requirements correctly reflect safety concerns*", while another considered an opportunity to "*link acceptance (evidence) to assurance cases*". Finally, one participant noted the integration as essential for the Change Request Analysis activity because "*ill-understood changes was a major factor in about half the fatal accidents I have investigated*". Another participant pointed it out as a moment to "*capture rationale and risk consideration*". Another one viewed the ACD as a supporting tool to analyse the "*impact on the requirements*" of the change requests. No comment was provided for the Elicitation activity.

4.3 SURVEY DISCUSSION

In this section, we highlight and discuss the key findings of this survey.

4.3.1 Participant Profiles

More than half of the participants worked in the United Kingdom, where the AC seminal approaches and the first standards initially appeared. Europe is the continent with the most countries indicated and most participants working with ACDs. Six continents were revealed, thus showing that ACD practice is widespread yet more adopted in Europe (Section 4.2.5).

Participants with 10 or more years of experience in SCS were 78%, and those that worked on five or more projects with ACD were 68% (Sects. 4.2 and 4.4). Thus, the survey's findings represent the practice and opinions of an experienced group of professionals.

We sought to analyse the industry practice from individual practitioners' perspectives (unit of analysis). Participants who work for multiple companies impact the generalisation of findings based on the sample's frequency. However, they hold knowledge on a broader range of practices, so their responses help identify practices used in the industry. Consultants and researchers most likely work with multiple companies (Section 4.2.3), so they bring observations and opinions that serve the study purpose to be an exploratory view on the topic.

4.3.2 State of ACD Practice

ACD is mainly driven by regulations, as 52% of the participants worked in projects where AC was mandatory by regulations; 32% of them were in projects in which regulations recommended the use of ACs (Section 4.2.7). Also, we found that ACD is not only performed for compliance. organisations have incorporated ACD as mandatory or recommended practice, indicating a move towards standardisation of ACD as an assurance practice across projects, regardless of regulatory needs. Additionally, 16% of participants experienced situations where AC's usage was an optional, team-level decision. The investigation of the ACD in non-obligatory settings could shed light on the most valuable activities/artefacts.

There is a lack of literature on the roles involved in ACD activities. Our survey brings new information on these topics, although our sample size hinders the generalisation of our findings. We identified differences in the participation of ACD tasks based on personal experience

(Section 4.2.8). The least experienced professionals work less frequently in any type of ACD activity, with more engagement in construction and review. The most experienced ones work on all types of activity, with more frequency in review, assessment, and approval. Finding professionals who work in all ACD activities is expected because ACs serve as the amalgamation of requirements, constraints, decisions, and all kinds of process/product evidence needed to demonstrate the safety of a system.

Our results indicate the integration of ACD into SDLC in many organisations (Section 4.2.9). We found that 61% of the participants observed the AC being constructed in a parallel process, and 39% observed the reviewing of AC in parallel. The incremental construction was 45%, and the incremental review was 42%. Various approaches support both parallel or incremental integration of ACD into the development process (ALMENDRA; SILVA; VILELA, 2020). Parallel development of ACs is mainly adopted with V-model lifecycle (the most-used lifecycle for SCS development (HEEAGER; NIELSEN, 2018)), while incremental development is more associated with agile and iterative lifecycle (ALMENDRA; SILVA; VILELA, 2020). The difference we found in our results between parallel and incremental usage probably reflects a higher adoption of V-model or Waterfall lifecycle models in the organisations. Noteworthy, the inadequate practice of postponement of AC construction to later stages is still observed by 19% of the participants.

Our results show more usage of the visual notation GSN than other textual notations (Section 4.2.10), which contrasts with previous works that pointed to textual notation as predominant in the industry (HOLLOWAY, 2008; NAIR et al., 2015). Even if we separate from our results the responses by researchers (9 out of 31), GSN is still the most used notation: GSN (50%), Textual (18%), and Structured text (14%). There is a need for more evidence towards this topic, particularly in investigating the factors that influence the choice of notation. Therefore, the support for multiple notations should be a key feature in future research and development of tools.

Participants reported low automation for SAC management (Section 4.2.12). The absence of responses from the majority of participants probably indicates that there is low automation in industry practice. On the other hand, the few answers we received meant that much of the current automation mainly focuses on reporting capabilities. None indicated the use of model-driven approaches, although its use in industry is acknowledged in other studies (VARA; RUIZ; ESPINOZA, 2018). Our results show a clear gap between the practice observed by the participants and the state of the art in AC automated development.

4.3.3 State of Integration

The most common scenario for traceability management is to use a tool to handle both artefacts and links (39%) (Section 4.2.13). Practices regarded as inadequate are still observed, such as creating trace links only at the end of the project (10%) and keeping trace links only in the certification documents (13%). The assurance information that comprises ACs includes more than just lists of requirements and trace links, so there is a need to capture the rationale for many decisions. The integrated scenarios for traceability management are better options for handling assurance information throughout the development lifecycle.

The discussion of how a system will be assured impacts project planning and should not be discussed after the requirements definition (KELLY, 2018). We found that safety assurance is frequently addressed during RE activities (Always and Very often = 68%) (Section 4.2.14). ACs are artefacts consulted during all RE activities, usually Specification (Always and Very often = 39%) and Change request analysis (Always and Very often = 42%).

Participants reported cross collaborations between safety and requirements professionals (Section 4.2.16). Safety engineers frequently engage in RE activities (Every and Most of the projects = 60%) and review RE artefacts (Every or Most of the projects = 59%). Such collaboration is expected to foster the early consideration of safety concerns during requirements development, the integration of engineering areas and help achieve viable requirements.

Less frequently, requirements engineers participate in ACD activities (Every and Most of the projects = 54%), in safety analysis (Every and Most of the projects = 41%), and review safety engineering artefacts (Every and Most of the projects = 32%) (Section 4.2.16). Such collaboration is expected to raise awareness of requirements engineers of safety concerns and necessary trade-offs. It also benefits traceability management as requirements engineers may review traces between requirements and the other safety artefacts while acting as knowledge sources in the safety activities.

Our findings show a practice gap in the interplay of RE and ACD activities and promising collaboration scenarios that could be further explored. Also, future research could investigate the observed asymmetry in which safety engineers participate more in requirements activities than otherwise.

4.3.4 Views of Practitioners on Integration

The benefits of RE and ACD integration are mainly the promotion of shared awareness of safety concerns and the early cross-collaboration in artefacts and traceability management (Sects. 4.17 and 4.18). Other benefits indicated by participants were the opportunity for safety engineers to understand the customer and product needs and also for requirements engineers to understand limitations imposed by hazards. The need to close the gap between safety engineering and software engineering is challenging (SIKORA; TENBERGEN; POHL, 2012), and the RE activities may be a way to connect these disciplines.

The proposal for performing ACD during RE activities has the agreement of 90% of the participants (Agree and Strongly Agree) (Section 4.2.19). It is a promising finding to foster more research towards integrating such areas.

Incorporating new practices has to tackle current issues to justify the process change effort. We identify in literature problems recurring in SCS development and asked their perceptions of the impact that the integrated development of requirements and ACs could have on those problems (Section 4.2.20). Most of the participants are confident that the following problems could be mitigated: late discovery of safety requirements or constraints; late rework due to the impossibility to build a compelling AC; and lack of integration between safety engineering and software development teams. Thus, participants consider the integration of ACD and RE as more about promoting discussion towards safety assurance rather than focusing on the final quality of certification artefacts. This perception may explain the number of participants who work with ACD in projects with no explicit obligation to produce an AC (Section 4.2.7).

The RE activities participants indicated as more suitable for ACD integration are Specification (81%) and Change request analysis (61%) (Section 4.2.21). The benefits of ACD integration into the Specification activity, characterised by the participants, are the early traceability elaboration among design decisions, requirements, hazards, and the grasp of what evidence is needed to show that the requirements specification is met. It's worth noting that the expectation towards integrating ACD into the requirements specification activity exceeds by far the current observed practice, which indicates a promising research gap to be addressed.

4.4 SURVEY THREATS TO VALIDITY

This section discusses the threats to validity concerning our survey's planning, design, and execution. We adopted the approach presented by Wohlin et al. (2012).

4.4.1 Construct Validity

The quality of the questions and answers is key to ensuring that a survey measures what it intends to measure. We took several steps to increase their quality. First, we reused questions and their answer options from a previously published survey for the profile questions (VARA et al., 2016). Second, we performed a pilot questionnaire application with four practitioners from the industry and academy (their profiles are described in Section 4.1.3). To evaluate the instrument's comprehensibility, the pilot participants answered the questionnaire by themselves without any mentoring.

Our survey is an exploratory study conducted as a self-administered questionnaire. So, we sought to reduce the number of open-ended questions, as they take more time from responders and could lead to more dropouts. Thus, we decided to present pre-defined scenarios or statements in some questions while leaving the possibility for the participant to include an open-ended text answer in the Other option. Few participants used this option, thus indicating that the set of answer options provided in the questionnaire was adequate to capture the variations in the participants' practices.

Another aspect is that participants may experience multiple varying practices simultaneously, be it in different projects or organisations. So, we designed some questions as multiple-choice to capture all current practices. We explicitly asked participants to consider only the last five years of experience to not include deprecated practices in the survey's responses.

We focus this study on discovering which integrated activities currently occur and their benefits perceived by practitioners. A complementary facet would be investigating the challenges and pitfalls in such integration, which we recommend be addressed in future research.

4.4.2 External Validity

This survey's findings cannot claim to cover the integration of RE and ACD as done by the whole population of practitioners working in SCS development. First, it is unclear what

would be the extension of such a population. Second, sampling by convenience is likely to attract more motivated people, which could not represent the whole population (KITCHENHAM; PFLEEGER, 2008). The personal interest in the two topics (RE and ACD) of our survey, which influenced the decision to participate, probably is reflected in the opinion questions (discussed in Sects. 4.17 to 4.21). Finally, our decision of sampling by convenience to target expert participants resulted in a small sample size ($n = 31$). However, our study provided interesting findings that help draw an initial characterisation of the interrelation between RE and ACD. We gathered participants with varying roles, without any dominant category of practitioners. Although a homogeneous sample could lead to a better characterisation of a specific group, a heterogeneous sample provides richer information to portray the state of the practice. Finally, as the instrument was designed to be self-administered, its replication is feasible. So, future replications of this study may expand the sample and allow for a historical analysis of trends.

4.4.3 Internal Validity

The selection of participants and how to treat their answers may affect the internal validity of this survey. To avoid responses from people outside the expected profile, we clearly explained the desired profile on the landing page (see Appendix). We also included survey questions towards the experience with SCS development and ACD. The participants' general profile comprises professionals with considerable expertise in SCS development and AC development. Regarding the experience in the different ACD activities, our sample is unbalanced towards people who work more frequently in Review, Assessment, and Approval activities rather than in Construction. It would be interesting for future research or replication of this survey to devise ways to reach practitioners who more frequently work in AC construction.

4.5 CHAPTER SUMMARY

This chapter has reported the design and execution of a practitioner survey to establish the state of the practice towards integrating Requirements Engineering (RE) and Assurance Case Development (ACD) activities during the development of safety-critical systems. Our survey attracted 31 participants working in 20 different countries, who represent a variety of profiles in terms of job roles, organisation roles, and years of experience.

The results obtained in this survey show that the development of ACs is being perceived as

beneficial beyond the simple need to pass a certification procedure. Also, our results indicate that ACD is being integrated into SDLC. The mixed observation of parallel and incremental integration approaches suggests that research on processes and tools should seek to support both paradigms combined.

We found evidence of cross-collaboration between requirements and safety teams, which furthers the need to investigate how the current state of practice regarding processes/tools impacts this cooperation. We identified that ACD is consulted during RE activities, so future investigations could aim to leverage the integration of assurance and requirements information.

Participants consider the integration of ACD and RE as more about promoting discussion towards safety assurance rather than focusing on the final quality of certification artefacts. This perception aligns with the number (almost half) of participants who reported that they worked with ACD on projects with no explicit obligation to produce an AC.

Our findings here provide perspectives on integrating ACD into the processes for SCS development, particularly for RE processes. Throughout the SDLC, requirements development needs to address how the requirements will be assured and what evidence is required. This integration is especially interesting, and the current state of the practice already presents the first step: the people responsible for AC and RE already cross-collaborate. There are, however, challenges regarding the level of automation and integrated use of tools to support such integration.

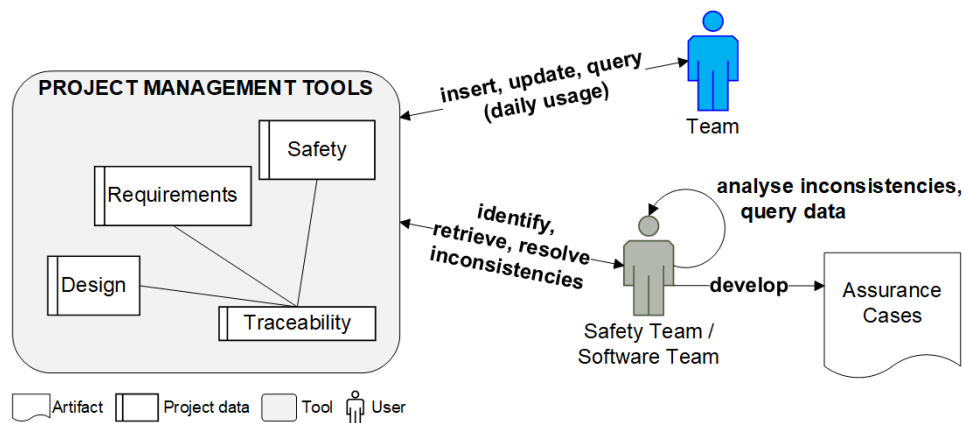
We identified good topics for future research. Mainly, replication studies could be carried out as self-administered questionnaires or based on interviews to strengthen the generalisation of the findings. The use of interviews would bring richer data for the open-ended questions. Further research could also investigate the factors that drive the non-mandatory adoption of ACD by organisations. We also seek to investigate different characteristics of the interplay of RE and ACD at the project level, which could leverage how the requirements and assurance practices are used together. In particular, investigating the challenges and pitfalls of such interplay is an exciting topic to explore. Finally, the expectation towards integrating ACD into requirement specification activities exceeds the current observed practice, indicating a promising research gap to be addressed.

5 FRAMEWORK DESIGN

In this chapter, we present the conceptualisation and objectives of the ARCADE framework and the development of the Safety Assurance Ontology.

We envision safety assurance and software teams working together in an integrated repository where each team can manage the project information under their responsibility and collaboratively create and review the associated traceability. Our target scenario is any team/-company that already manages regular project information inside an Issue Tracking System (ITS) or similar tool, that is, they manage requirements, design definitions, tests, and other project items somehow inside tools. Figure 24 illustrates our target scenario.

Figure 24 – Illustrative scenario for collaborative work between safety and software teams.



Source: The author (2023)

We aim to leverage project management tools, allowing them to serve as a collaborative place where the interrelations between safety assurance and regular project information can be explicitly registered and managed together early and continuously throughout the development lifecycle. To achieve this vision, some challenges arise. First, as discussed previously in chapters 1 and 2, regular project information items (e.g., requirements, design, tests) are most likely managed separately from safety-related information (e.g., hazards/risk analysis, safety requirements, safeguard measures, rationales, assumptions, etc.), and only bound inside AC artefacts and during ACD activities.

The adoption of information models supports the developers in managing artefacts and tracing information required for the certification (MÄDER et al., 2013). Such underlying models help organise and interrelate the information stored in project management tools such as ITS, application lifecycle management and requirements tools.

In the safety domain, the set of information elements that may comprise the specification of an SCS is large. In Vilela et al. (2018), the Safe-RE metamodel was devised to support requirements activities in the early phases of the development process. Safe-RE comprises concepts and relationships representing the inputs and outputs of safety-related activities required by a set of ten standards. Not all the constructs are expected to be needed in the specification of a single system, as some may not be necessary or not applicable to the target standard or domain. Also, it may be a choice of the teams that a kind of information is not tracked as single items in project management tools. For example, a team may choose to track hazards individually in an ITS, but with regard to the tracking of risks, the choice is a single “Risk Analysis” artefact that refers to the group of all the risks. Thus, underlying models are greatly beneficial but need to be tailored for better adoption (CLELAND-HUANG et al., 2014).

5.1 GOALS

We formulated the following goals for the ARCADE framework:

- G1** Support the integrated management of assurance and regular project information.
- G2** Support the automated assessment of integrated assurance and regular project information.
- G3** Support the automated generation of assurance case fragments.
- G4** Support the customisation of the framework for its adoption in different SCS project settings.

To achieve these high-level goals, we defined the specific sub-goals:

- SG1** Support the use of safety-related information directly inside general-purpose project management tools (supports **G1**, **G3**).
- SG2** Provide automated analysis of traceability among project items retrieved from project management tools, that can be requested anytime during the project lifecycle (supports **G2**).
- SG3** Provide automated assurance case generation that can be requested anytime during the project lifecycle (supports **G3**).

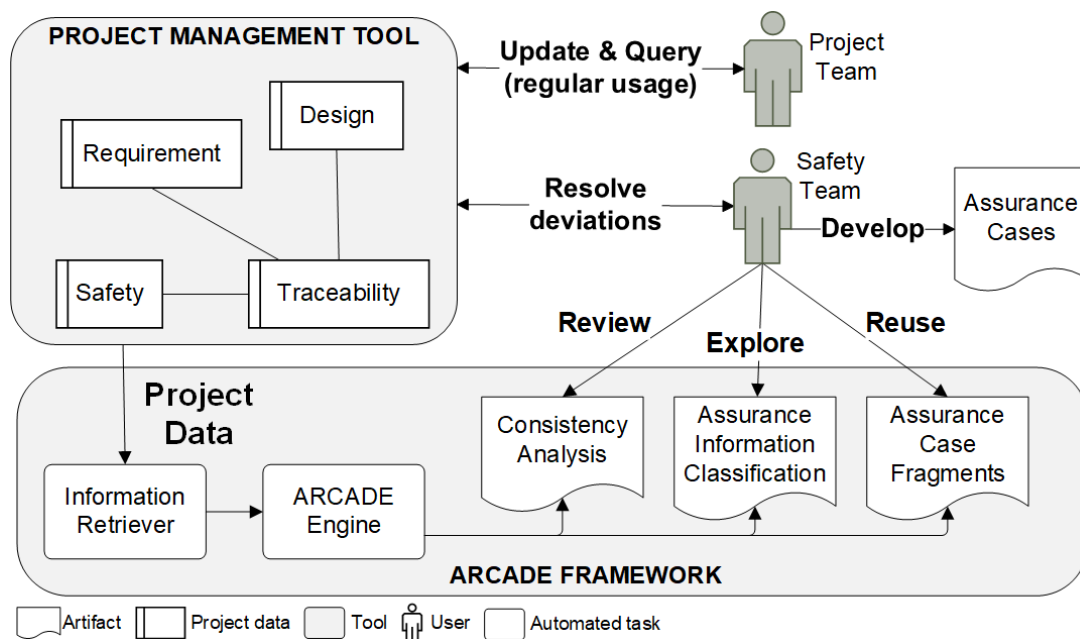
SG4 Support early stages of systems development, when there are naturally many gaps in the project dataset (supports **G1**).

SG5 Provide a base information model that could be extended for singularities of each project (supports **G4**).

5.1.1 Framework Usage Overview

The ARCADE framework proposes a mechanism to automatically retrieve and process regular and assurance-related project information from issue trackers, to help teams in the identification of gaps and inconsistencies that could hinder certification procedures. Figure 25 shows an overview of the ARCADE framework.

Figure 25 – Overview of assurance case development supported by ARCADE



Source: The author (2023)

ARCADE performs automatic reasoning to check for inconsistencies, and performs inference to discover implicit information. The framework also generates assurance case fragments based on the provided and inferred data, thus supporting early and continuous assessment of the quality of assurance information recorded in the project management tools. Before using the framework web tool, it is necessary to tailor the Information Retriever component to connect and extract data from an external project management tool. Then, the usage of the framework is simple: an user should provide an unique identifier of the project (inside the

external tool) and request the analysis of the data. The framework main engine performs the following steps in the following sequence:

- Triggers the Information Retriever feature to gather the data for the selected project identifier, and translate to the ARCADE interchange format;
- Triggers the Consistency Analysis feature passing the retrieved data as input; this feature performs automatic reasoning, inference, and prepare the Consistency Analysis report;
- If inconsistencies are found, the computation of explanations is triggered and the sequence stops. If not, the sequence continues;
- Triggers the Knowledge Discover feature to query provided and inferred data and prepare the Assurance Information Classification report; and
- Triggers the Assurance Case Generator feature to construct fragments based on the provided and inferred data; the sequence ends.

5.1.2 Design Decisions

We discuss below three aspects that led to key design decisions during the inception of the ARCADE framework.

Integrated data representation. Current modern project management tools provide features to query and extract project data. In particular, ITSs are widely used and could record much knowledge about the system under development. Each ITS has its schema for representing project assets, and many allow users to create customised concepts and relationships among them. A design challenge was to deal with the different ways to organise project data we could encounter, yet we needed to build a reusable, easy-to-adapt solution. We start from the premise that a company already manages regular project information inside an ITS or similar tool. This means that requirements, design definitions, tests, and other project items are somehow stored in the ITS.

Making teams manage safety assurance information using their well-established, daily-used project management tool may leverage a system's safety analysis and the development of SACs. In this context, we designed an information model that combines concepts and relationships from three domains: a reference model for requirements traceability (RAMESH; JARKE, 2001), a safety requirements metamodel based on industry standards (VILELA et al., 2018), and the

Goal Structuring Notation (GSN) standard for Assurance Case (AC) representation (GOAL..., 2021). The information model is used to:

- Map the information schema used by a company to the ARCADE information model;
- Guide how to customise the company's information schema to represent the concepts and relationships related to safety assurance;
- Provide semantics to the interchange format used to gather project data and input into ARCADE Engine.

In this scenario, a company that adopts our approach may keep its current way of representing project information, only needing to enrich its tool with new concepts and relationships related to safety assurance. It is noteworthy that a specific project information retriever tool needs to be developed to extract and transform the project data into the input format of ARCADE.

This design decision addresses the sub-goals **SG1**, **SG2** and **SG3**.

Interfaces for data extraction. After enabling data integration inside a management tool, there is a need to retrieve the data for processing. This retrieval poses a design challenge, as each tool has proprietary interfaces and data formats. We defined an ARCADE interchange format to which the company project data should be transformed. This interchange format serves as input in the ARCADE Engine component. This decision allows the development of a reusable engine, easing the adaptation of the framework to any project management tool.

We resorted to semantic web technology and choose semantic triples as the syntax of our interchange format (see Section 2.3), while the semantics are provided by the ARCADE Information Model (AIM). We later discuss how this interchange format based on RDF allows previewing the benefits of ARCADE for adoption by a company.

This design decision addresses the sub-goals **SG1** and **SG5**.

Open-world assumption and extensibility. We decided to address the consistency analysis of the project's knowledge using a system of logic that makes the Open-World Assumption (OWA). OWA is useful for scenarios in which knowledge is managed as it is discovered, that is, the complete information is not available at once. This is the scenario in the early stages of a software project system development when much information needed for the final certification procedures is yet to be produced.

We chose to use description logic (DL) as our logic system. It is sufficiently expressive to represent project data repositories, and the core reasoning for DLs is mostly decidable (i.e. reasoner will not run forever) (W3C, 2023a). As efficient reasoner tools are available (W3C, 2023a), we can rely on this technology to build predictable tools for general-purpose uses. The Ontology Web Language (OWL) is one of the most common languages used for ontologies related to software engineering (DERMEVAL et al., 2016) and is compatible with DL.

We designed the Safety Assurance Ontology (SAO), an operational ontology implemented in OWL that models the concepts, relationships and constraints of the AIM. Its DL expressivity¹ is $\mathcal{ALCRIF}(\mathcal{D})$. Some consistency rules are implemented using SPARQL, a semantic query language that can be applied in RDF. Using semantic web tools/libraries (later detailed), ARCADE Engine performs consistency checking, implicit knowledge inference and knowledge extraction.

The base version of the SAO provides a lean (yet precise) set of definitions and rules that address core aspects of safety-critical software projects. To have practical applicability, this technology needs to be extensible to address the idiosyncrasies of a company development process. Extensibility is inherent to semantic web technologies; thus, our ontology base version can readily adapt to a company's needs (or even use different versions across projects).

This design decision addresses the sub-goals **SG2**, **SG3** and **SG4**.

Framework architecture. We decided to deliver the framework as an independent web application that can be integrated into any development operations (DevOps²) pipeline. Our main motivation is to allow ARCADE to be easily integrated into any development process and to be able to connect with any ITS, ALM, or repository tool.

We chose the Java platform³ because of its flexibility and extensive availability of third-party libraries. Noteworthy, many third-party libraries related to the semantic web are only available for the Java platform.

This design decision addresses **all the sub-goals**.

¹ Description logic - Naming convention. Available at: <https://en.wikipedia.org/wiki/Description_logic#Naming_convention>. Accessed in: July 1, 2023.

² DevOps. Available at: <<https://en.wikipedia.org/wiki/DevOps>>. Accessed in: May 1, 2023

³ Java. Available at: <[https://en.wikipedia.org/wiki/Java_\(software_platform\)](https://en.wikipedia.org/wiki/Java_(software_platform))>. Accessed in: May 1, 2023

5.1.3 Expected Benefits

The design of the ARCADE framework was based on needs and challenges identified in the literature review and in our previous studies reported in chapters 3 and 4. Below, we list the expected benefits of the ARCADE alongside the studies that motivated them:

- B1** Improve team communication towards safety assurance (ALMENDRA; SILVA; VILELA, 2020; ALMENDRA et al., 2022);
- B2** Reduce effort to retrieve and assess safety assurance information (CHENG et al., 2018; ALMENDRA et al., 2022); and
- B3** Reduce effort to build safety assurance cases (DENNEY; PAI, 2012; ALMENDRA; SILVA; VILELA, 2020).

These are very ambitious benefits, and we do not profess that our solution thoroughly addresses each aspect. Indeed, our framework supports new or improved team activities that leverage these benefits.

B1. Instead of managing safety assurance information in separate documents or tools, we support integrated management with software project management tools (ALMENDRA; SILVA, 2020). A unified place for storing, retrieving and cross-referencing eases traceability management, thus facilitating cross-collaboration (ALMENDRA et al., 2022).

B2. Our framework provides automated extraction and analysis, helping users to review inconsistencies and incompleteness in the project data.

B3 The maintenance of standalone AC artefacts may be cumbersome as project data evolves. We argue that during most of the project lifecycle, there is no strong need to build complete AC. Our framework provides automated generation of AC fragments in multiple perspectives, which can be useful for safety assurance engineers to assess the quality of evidence throughout the development process. When a complete AC is needed, the fragment generated can be used as starting point.

The adoption of ARCADE is, however, not trivial. First, it requires that the project team agrees on using an ITS as a unified repository for traceability, design rationale and assurance information. Second, a customised project information retriever needs to be implemented according to the mapping of AIM to the ITS. Finally, the framework provides a core ontology

that may need to be extended to address the safety regulations that apply to the project. Although these two last actions only need be carried out once per project, they require personnel acquainted with semantic web technology.

In the next section, we present the design and specification of the ontology that backs the ARCADE framework.

5.2 SAFETY ASSURANCE ONTOLOGY

In this section, we introduce the Safety Assurance Ontology which is the pillar of the ARCADE framework. We present the methodological steps, objectives, design and specification of the ontology.

5.2.1 Methodology

The methodological steps below were based on discussions of ontology building in Gómez-Pérez and Benjamins (1999), knowledge engineering principles in Schreiber (2008) and the METHONTOLOGY process in Fernández-López, Gómez-Pérez and Juristo (1997). The steps are the following:

- Requirements specification: Define the scope of ontology and its intended use;
- Knowledge gathering: Conduct a literature review to identify knowledge sources;
- Implementation: Conduct iterative cycles of conceptualisation, integration, implementation and verification of ontology prototypes; and
- Evaluation: Demonstrate ontology usage through an illustrative scenario.

It is important to point out that this is not a linear, cascading process. Each step requires a review of definitions made in previous steps, so it is an iterative and incremental process.

5.2.2 Specification

5.2.2.1 Purpose and Scope

The purpose of an ontology is to integrate common and interrelated knowledge of two systems engineering practices: safety engineering and software engineering. We refer to the conceptual model of our ontology as “information model”, which is developed alongside the ontology specification. The intended use of this ontology is to serve as a base system to integrate outputs of safety and software engineering activities performed by all kinds of personnel involved in safety-critical system development (e.g., developers, safety/assurance engineers). To support users’ needs, we formulated competency questions that the framework should answer concerning the project information provided.

5.2.2.2 Intended Users and Scenarios of Use

The intended use of this ontology is to serve as a base system to integrate outputs of safety and software engineering activities performed by all kinds of personnel involved in SCS development (e.g., developers, safety engineers, assurance engineers). Intended users and their goals are the following:

- **Software engineers** working on requirements, design, tests and other kinds of artefacts who need to assess the consistency of software specifications and completeness of traceability with safety assurance artefacts.
- **Safety engineers** working on safety analysis who need to assess the consistency of safety specifications and completeness of traceability with software specifications.
- **Assurance engineers** producing assurance cases or other certification documents who need to retrieve project information and assess its correctness and completeness with regard to safety assurance argumentation.

The scenarios of use are all related to actions users can perform in the knowledge repository:

- Register software project information items (e.g., requirements, design definitions);
- Register safety analysis and assurance argument information items (e.g., hazards, environmental contexts, assumptions);

- Register attributes associated with items;
- Register traceability between items;
- Review provided (explicit) item classifications and relationships;
- Review inferred (implicit) item classifications and relationships;
- Assess correctness;
- Assess completeness;

This ontology is classified as a *Task-Specific Ontology* because it embodies just the conceptualisations that are needed for carrying out a particular task (SCHREIBER, 2008).

5.2.2.3 Sources of Knowledge

The knowledge gathering comprises a literature review to identify knowledge sources, such as foundational ontologies, related domain-specific ontologies, international standards, datasets, and other metamodels. We chose two metamodels (RAMESH; JARKE, 2001; VILELA et al., 2018) and a standard (GOAL..., 2021) that serve as domain-specific ontologies for connecting our ontology to both safety assurance and software traceability domains. As basic vocabulary, we used the RDS Schema, a data-modelling vocabulary for RDF data (BRICKLEY; GUHA, 2014). We also used assurance case datasets and pattern catalogues as sources. These sources are presented in Table 21.

Table 21 – Knowledge sources for Safety Assurance Ontology

Source		Main contributions	Reference
Traceability model	reference	Core traceability concepts	(RAMESH; JARKE, 2001)
Safety metamodel	requirements	Core safety concepts applied to software systems	(VILELA et al., 2018)
GSN standard		Assurance case concepts and rules of formation	(GOAL..., 2021)
RDF Schema		Basic vocabulary to represent objects, object attributes and links in RDF	(BRICKLEY; GUHA, 2014)
Open Patient-Controlled Analgesia Pump		Dataset with assurance cases and project information	(HATCLIFF et al., 2018)
Generic Patient-Controlled Analgesia Pump	Patient-Analgesia	Dataset with assurance cases and project information	(Generic Infusion Pump Research Project, 2023)
Generic Insulin Infusion Pump	Insulin Infusion	Dataset with project information	(Generic Infusion Pump Research Project, 2023)
Kelly's catalogue		Assurance case argument patterns	(KELLY; MCDERMID, 1998)
Szczygielska's catalogue		Assurance case argument patterns	(SZCZYGIELSKA; JARZĘBOWICZ, 2018)

Source: The author (2023)

Ramesh and Jarke argue that rationale management demands specialised concepts and relationships, and also proper tool support to foster rationale capture (RAMESH; JARKE, 2001). The authors provide a reference model for requirements traceability, and we chose it because of its simplicity and extendability. Their perspective simplifies traceability into three fundamental concepts: *Stakeholders*, *Objects* and *Sources*. *Stakeholders* manages *Sources* and has roles in *Objects* lifecycles. *Sources* are artefacts or intangible references to people or undocumented knowledge, they document *Objects*. Finally, *Objects* are the inputs and outputs of the development process; they trace to each other. We chose this reference model to organise the concepts related to project information because of its simplicity. Mapping these concepts to an ITS, *Stakeholders* are users, *Objects* are all the managed items and configurations (e.g., issue, components, and fields), and *Sources* are links to external information (e.g., weblinks and attachments).

The Vilela et al. safety metamodel supports the analysis and specification of safety requirements and safety analysis information (VILELA et al., 2018). It provides support for fine-grained specification, thus allowing its use both for simple and sophisticated levels of traceability. The level of detail used to build SACs may vary; thus, ARM was designed to support multiple levels

of refinements for the two main concepts, Requirements and Hazards. We selected high-level concepts of Safe-RE to be stated as Objects for the sake of traceability inside an ITS. The selection was based on the analysis of the granularity of requirements stated in open datasets such as (WEINSTOCK; GOODENOUGH, 2009).

In ALM tools, Objects and Sources are more likely to be managed seamlessly. For example, a customised issue could record a high-level requirement for a system and be decomposed into smaller requirements for development. This requirement will have trace links to the derived low-level requirements. In ARM, we kept Object and Source as separate concepts, as in the original reference model (RAMESH; JARKE, 2001). Such separation works well when a Source is intangible (e.g., a person), just a textual reference to external artefacts, or a URI (Universal Resource Identifier) to another project management tool, such as a wiki or test repository. The Source concept comprises the input and outputs of Safe-RE that are more likely to be handled as external artefacts, not managed by the ITS, such as a Hazard Analysis Report or an Architecture Specification.

5.2.2.4 Competency Questions

A competency question is a natural language question that the ontology should be able to answer. Questions and their responses serve as a requirements specification against which the ontology can be evaluated (SUÁREZ-FIGUEROA; GÓMEZ-PÉREZ; VILLAZÓN-TERRAZAS, 2009). Thus, the questions were specified towards the informational needs of team members reviewing assurance information or constructing assurance cases. Also, the questions encompassed traceability gaps that impact the quality of the arguments. They were identified by analysing each kind of relationship that each concept may or should have in a consistent and complete model instance. Therefore, these questions guided the implementation of the internal data model and the data queries.

Table 22 lists some of the competency questions that comprise the specification of the ARCADE ontology. This set of questions specifies types of items, relationships and attributes that need to be handled. Additionally, we extracted rules and constraints from the metamodels on which our information model is based.

Table 22 – Competency questions for Safety Assurance Ontology

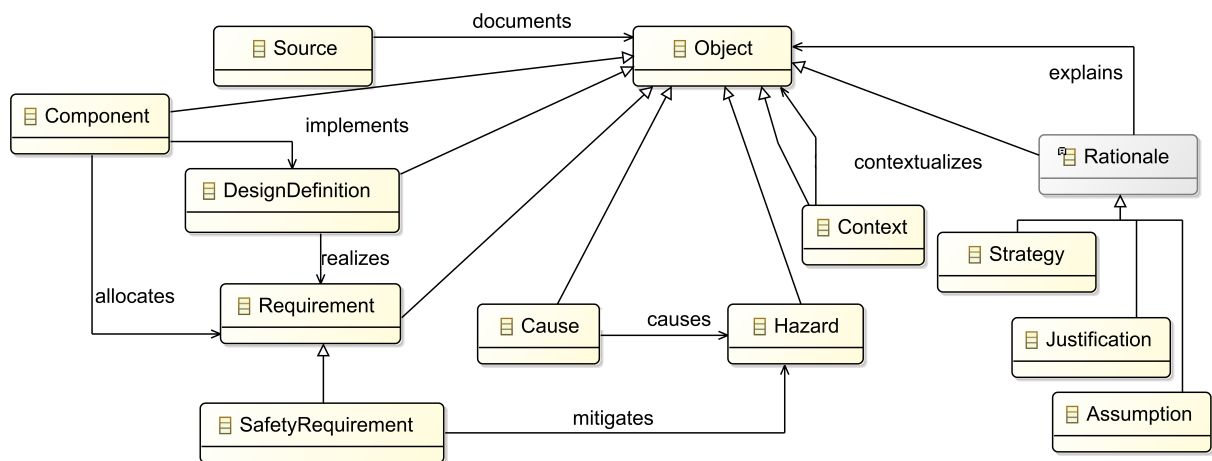
Basic (“Which are the...”)
... Objects and their classifications?
... Relationships associated with an Object?
... Hazards?
... Safety Requirements?
... Requirements?
... Design Definitions?
... Causes?
... Contexts?
... Components?
... Assumptions?
... Justifications?
... Strategies?
... Sources?
Traceability gaps (“Which are the...”)
... Requirements that are not allocated in any Component?
... Safety Requirements that do not mitigate any Hazard?
... Safety Requirements that are not explained by any Rationale?
... Hazards that are not mitigated by any Safety Requirement?
... Causes that are not cause of any Hazard?
... Hazards that have no Cause associated?
... Hazards that are not explained by any Rationale?
... Hazards that are not contextualized by any Context?
... Rationales that do not explain any Object?
... Rationales that are not classified as any of the subclasses?
... Contexts that do not contextualize any Object?
... Components that are not allocated to any Requirement?

Source: The author (2023)

5.2.2.5 Information Model

The ontology handles the project information at the granularity of single, uniquely identified items such as an issue in an ITS. The ontology also handles the traceability between the project items. It is out of the scope of this work to discover or handle knowledge inside the natural language texts of the project items.

Figure 26 – Information metamodel of Safety Assurance Ontology



Source: The author (2023)

Table 23 – Information model concepts and relations of Safety Assurance Ontology

Concept	Description
Object	Input or output of the development process, in any level of granularity, that is managed in the ITS.
Source	An external artefact to the ITS.
Rationale	An explanation that provides additional information for an Object.
Strategy	A statement behind a project's decision. It can be a design decision, a requirements decision, or a mitigation strategy selected for a hazard.
Justification	A statement that provides an explanation for a project decision.
Assumption	A statement that reflects a common situation in the domain so ingrained into stakeholders' general beliefs, that it is unquestioned.
Hazard	A single hazard or category of hazards.
Requirement	A system requirement.
SafetyReq.	A requirement related to system safety. It represents actions and constraints that should or should not be performed to maintain the system in a safe state.
Context	An environmental or operational condition that may occur and is relevant to safety.
Component	A unit of architecture, design or code of the system.
Relationship	Description
<i>documents</i>	It links a Source to Objects that it documents.
<i>refines</i>	It represents a refinement of Requirement or Hazard.
<i>mitigates</i>	It links a SafetyRequirement to the Hazards that it addresses.
<i>causes</i>	It links a Hazard to Objects identified as its cause.
<i>explains</i>	It links a Rationale to Objects it provides an explanation.
<i>realizes</i>	It links a Component to Requirements allocated to it.
<i>contextualizes</i>	It links a Context to Objects.

Source: The author (2023)

Besides the definition of concepts and relationships, the ontology specifies the rules listed in Table 24. Rules from 1 to 20 are not restrictions, but instead, classifications that will be inferred if the ontology reasoner finds items/links matching these conditions. Finally, rules 21 and 22 are restrictions, and if an object is classified as more than one of the disjointed classes, then an inconsistency should be triggered by the framework.

Table 24 – Formation and classification rules of Safety Assurance Ontology

Rule	Description
1.	Object that <i>allocates</i> other is a Component
2.	Object that <i>is allocated in</i> other is a Requirement
3.	If A <i>refines</i> B and B <i>is allocated in</i> C, then A <i>is allocated in</i> C
4.	Object that <i>causes</i> other is a Cause
5.	Object that <i>is caused by</i> other is a Hazard
6.	If A <i>contextualizes</i> B and B <i>is refined by</i> C, then A <i>contextualizes</i> C
7.	Object that <i>contextualizes</i> other is a Context
8.	Object that <i>documents</i> other is a Source
9.	Object that <i>explains</i> other is a Rationale
10.	Object that <i>implements</i> other is a Component
11.	Object that <i>is implemented by</i> other is a Design Definition
12.	Object that <i>documents</i> other is a Source
13.	Object that <i>mitigates</i> other is a Safety Requirement
14.	Object that <i>is mitigated by</i> other is a Cause or Hazard
15.	If A <i>mitigates</i> B and B <i>causes</i> C, then A <i>mitigates</i> C
16.	If A <i>refines</i> B and B <i>mitigates</i> C, then A <i>mitigates</i> C
17.	Object that <i>realizes</i> other is a Design Definition
18.	Object that <i>is realized by</i> other is a Requirement
19.	Object that <i>refines</i> or <i>is refined by</i> a Hazard is a Hazard
20.	Object that <i>refines</i> or <i>is refined by</i> a Requirement is a Requirement
21.	Object can only be classified as one of Component , Design Definition , Hazard , Rationale or Requirement
22.	Object can only be classified as one of Assumption , Justification or Strategy

Classes are highlighted in **bold** and relationships in *italics*.

Source: The author (2023)

5.2.3 Implementation

We resorted to semantic web technologies and choose OWL/DL as our logic system (W3C, 2023b), as discussed in Section 2.3. We used Protégé⁴ as a supporting tool for the iterative cycles of specification, implementation and evaluation, and we used a set of third-party libraries to run the ontology inside our framework (discussed further in Chapter 6).

The dialect used to specify our ontology is the OWL 2 EL⁵, which is sufficiently expressive to represent project data repositories (as discussed in Section 2.3), and the core reasoning is decidable (i.e., the reasoner will not run forever). Also, the use of OWL 2 EL brings some capabilities for inferring implicit classification based on inward and outward relationships, based on a chain of relationships and based on attributes. Finally, complex rules/constraints can be implemented using SWRL, and ontology data can be queried using SPARQL (W3C, 2023b). This technology stack supports many ways to extend the framework to specific teams' needs.

The entire OWL specification (TBox) for the Safety Assurance Ontology is presented in Appendix D. We illustrate how the information model (Figure 26) and the rules (Table 24) are translated as OWL specifications in Source Code 1. The specification of the relationship *mitigates* and its associated rules read as follows:

- Relationship *mitigates* is implemented in lines 1 to 22 as an object property.
- Rule 13 is implemented in line 2 as a property domain.
- Rule 14 is implemented in lines 3 to 10 as a property range.
- Rule 15 is implemented as a property chain in lines 11 to 14.
- Rule 16 is implemented as a property chain in lines 15 to 18.

Source code 1 – OWL specification for *mitigation* relationship and associated rules

```

1 <owl:ObjectProperty rdf:about="dare:arcade/sao#mitigates">
  <rdfs:domain rdf:resource="dare:arcade/sao#SafetyRequirement"/>
3   <rdfs:range>
    <owl:Class>
5     <owl:unionOf rdf:parseType="Collection">
      <rdf:Description rdf:about="dare:arcade/sao#Cause"/>
7     <rdf:Description rdf:about="dare:arcade/sao#Hazard"/>

```

⁴ Protégé ontology editor. Available at: <<https://protege.stanford.edu/>>. Accessed in: May 1, 2023.

⁵ OWL 2 EL. Available at: <https://www.w3.org/TR/owl2-new-features/#OWL_2_EL>. Accessed in: May 1, 2023.

```

    </owl:unionOf>
9    </owl:Class>
    </rdfs:range>
11   <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="dare:arcade/sao#mitigates"/>
13        <rdf:Description rdf:about="dare:arcade/sao#causes"/>
    </owl:propertyChainAxiom>
15   <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="dare:arcade/sao#refines"/>
17        <rdf:Description rdf:about="dare:arcade/sao#mitigates"/>
    </owl:propertyChainAxiom>
19   <rdfs:comment xml:lang="en">It links a SafetyRequirement to the Hazards that it
        addresses</rdfs:comment>
        <rdfs:label xml:lang="en">mitigates</rdfs:label>
21 </owl:ObjectProperty>

```

Source: The author (2023)

The loading of data into an OWL ontology requires the preparation of a set of individuals (ABox). We specified individuals directly in the user interface of Protégé throughout the iterative development of the ontology specification. However, for the automated import of data into the framework, we need a machine-readable format. We defined an interchange format to allow the framework to handle a single input format, regardless of the origin of the data.

The syntax of our interchange format is RDF Turtle (BECKETT et al., 2014). Turtle is a textual syntax for RDF that allows the specification of a data graph in a compact and natural text form. In this way, Turtle syntax is both machine-readable and human-readable and can be handled by many third-party tools and libraries. Regarding the semantic schema, the interchange format uses the ARCADE ontology and the RDF Schema (BRICKLEY; GUHA, 2014). Source Code 2 shows an example of project data represented in the interchange format.

Source code 2 – Example of data represented in the ARCADE interchange format

```

1  :R1 rdf:type owl:NamedIndividual .
   :R1 rdf:type sao:Requirement .
3  :R1 rdfs:label "Requirement 1" .
   :R1 rdfs:comment "This is a requirement" .
5  :H2 rdf:type owl:NamedIndividual .
   :H2 rdf:type sao:Hazard .
7  :H2 rdfs:label "Hazard 2" .
   :R1 sao:mitigates :H2 .

```

Source: The author (2023)

This interchange format serves as an integration point between ARCADE and any project

management tool. Instead of extending the core implementation of the framework to the specifics of each management tool, the responsibility of transforming the project data into the interchange format is delegated.

We expect variability in how teams and companies represent and manage their development data inside project management tools (ALMENDRA; SILVA, 2020). Imposing a rigid information model may hinder the adoption of any proposed solution.

Thus, teams may use different tools (ITSs, ALMs), but the concepts and relationships underlying their project data have to comply with regulations. Different information retrievers will need to be implemented based on the exporting interfaces of each tool. In another scenario, teams may use the same tool in different ways. For example, let two teams use an ITS, Team A chooses to represent hazards as a customised issue type, while Team B chooses to represent them as an issue which contains a label “hazard”. We need two different information retrievers to extract this type of data from these two settings.

This intermediary transformation step led to the need for the component “Information Retriever” discussed in the next chapter.

5.2.4 Evaluation

The evaluation of an ontology on the implementation level seeks to verify if the specification is consistent (BAIL, 2013). There are three steps to verify an ontology implementation:

1. If the set of axioms (TBox) is consistent;
2. If the set of axioms combined with valid individuals (ABox) is consistent;
3. If the set of axioms combined with invalid individuals is inconsistent.

The Protégé tool provides support for all these types of verification. Ontology engineers can specify axioms and assertions (known as individuals) inside the editor and also trigger the execution of OWL reasoners. In steps 1 and 2, the expected result is that no inconsistency is found. In step 3, the expected result is that the reasoner returns an inconsistency. Step 3 is intricate as the engineer needs to elaborate assertions for each aspect of the axioms that will be tested (SCHEKOTIHIN et al., 2018).

For example, to verify the proper implementation of Rule 13 (Table 24), we need to feed our ontology with some data items and check if the items are correctly classified. The following

project data can verify the rule:

- X is a project item of any type (except “SafetyRequirement”);
- Y is another project item of any type;
- Y mitigates X.
- Expected result after reasoning: X is classified as Safety Requirement.

We adapted two datasets for use during the ontology implementation and as illustrative data in the evaluation with experts (see Section 7.2 for more information).

5.3 CHAPTER SUMMARY

This chapter presents the main objectives and design goals of the ARCADE Framework. We also discuss the design and implementation of the Safety Assurance Ontology. The ontology supports the representation, assessment and classification of assurance and regular project information. The ontology comprises an information model, constraints, rules of formations, and competency questions. The main purpose is to support teams to continuously evaluate the quality of project information and its traceability.

6 FRAMEWORK IMPLEMENTATION

In this chapter, we present the results of the development of the ARCADE framework. We discuss the steps taken, the technologies chosen, how the ontology was operationalised and the set of features that compose the current version of the framework. This section is useful to understand the current limitations of the framework implementation and to understand the evaluation studies detailed in Chapter 7.

ARCADE Framework is implemented as a web system that provides a web user interface to demonstrate its features. It can retrieve data automatically from an Issue Tracking System (ITS) or load data from files using the interchange format. It automates the data retrieval, loading, reasoning, querying, and presentation of the project data and the generation of Assurance Case (AC) fragments.

The remainder of this chapter is structured as follows. Section 6.1 presents our development methodology steps. Section 6.2 presents the requirements. We discuss from Section 6.3 to Section 6.6 each main component's features and design details.

6.1 DEVELOPMENT METHODOLOGY

The framework development followed an iterative and incremental approach¹ grouped in phases of development loosely based on the Unified Process.² Below, we list the development phases and main results achieved in each phase:

- Inception phase results
 - Definition of the framework goals and sub-goals (Section 5.1);
 - Definition of the requirements (Section 6.2);
 - Selection of semantic web as underlying technology;
 - An overview of the framework conception is published in Almendra, Barros and Silva (2019).
- Elaboration phase results

¹ Iterative and incremental development. Available at: <https://en.wikipedia.org/wiki/Iterative_and_incremental_development>. Accessed in: May 1, 2023

² Unified Process. Available at: <https://en.wikipedia.org/wiki/Unified_Process>. Accessed in: May 1, 2023

- Ontology development using Protégé;
 - Selection and adaptation of Atlassian Jira as target project management tool;³
 - Conduct spikes⁴ to select third-party libraries for manipulation, reasoning and querying of OWL and RDF data;
 - Conduct spikes to select a third-party library for rendering SVG images (used to display AC fragments);
 - A preliminary version of the information model and the adaptation of Jira is published in Almendra and Silva (2020).
- Construction phase results
 - Development of the Information Retriever (IR);
 - Development of the Consistency Analysis (CA);
 - Development of the Knowledge Discovery (KD);
 - Development of the Assurance Case Development (ACD);
 - Refinement of the information model and ontology specification;
 - Deployment of staging version⁵ for evaluation purposes;
 - An overview of the current design and implementation is published in Almendra and Silva (2023).

At the time of the publication of this thesis, we have not initiated the Transition phase, in which we deploy the tool in a production environment for end users.

6.2 REQUIREMENTS

Table 25 lists the main requirements for the ARCADE framework tool. The following sections discuss design decisions and the current features implemented in each main component. Requirements, design and implementation decisions were based on the experience of the authors, to address the goals stated in the Section 5.1.

³ Atlassian Jira instance adapted for integration with ARCADE tool. Available at: <<https://arcade-dare.atlassian.net/>>. Accessed in: July 30, 2023

⁴ Spike solution. Available at: <<http://www.extremeprogramming.org/rules/spike.html>>. Accessed in: May 1, 2023

⁵ ARCADE tool demo. Available at: <<https://arcade.cin.ufpe.br/>>. Accessed in: July 30, 2023.

Table 25 – ARCADE requirements

Id	Requirement description (<i>The system shall...</i>)	Comp.
R1	... permit user access from the internet using desktop browsers	G
R2	... operate on a server running Apache Tomcat 9 and Java 11	G
R3	... not persist any project data loaded	G
R4	... allow the user to directly upload files with project data in the interchange format	IR
R5	... allow the user to request retrieval of project data from the adapted Jira by informing the project identification	IR
R6	... import project data through Jira REST API	IR
R7	... allow the user to reset any loaded project data	IR
R8	... run reasoner to evaluate ontology (TBox+Abox) consistency upon any project data loading	CA
R9	... run the computation of explanations if the ontology is inconsistent	CA
R10	... transform the axioms of the explanations into natural language statements	CA
R11	... list all the provided project items and their associated attributes and links	KD
R12	... list all the inferred project items and their associated attributes and links	KD
R13	... retrieve and list the answers to all the competency questions from project data	KD
R14	... generate assurance case fragments based on different argumentation patterns	ACG
R15	... generate assurance case fragments using GSN visual notation	ACG

Components: G = General; IR = Information Retriever; CA = Consistency Analysis; KD = Knowledge Discovery; ACG = Assurance Case Generator.

Source: The author (2023)

6.3 INFORMATION RETRIEVER

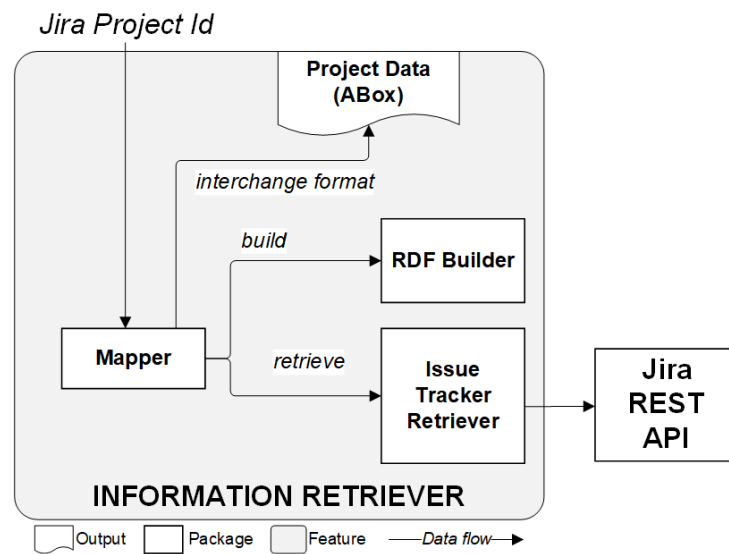
The task of the retriever is to transform the project data extracted from the management tool into semantic triples. The retriever is not meant to have a single implementation for all kinds of tools, as the data in the tools will not necessarily conform to the metamodel used by ARCADE. The information retriever encapsulates all the logic necessary to parse and map the project data (in the proprietary format of the project management tool) into the ARCADE interchange format. Using the semantics of our ontology and the RDF syntax, we provided a canonical interchange format to receive data from different tools.

There are two ways to ingest project data into ARCADE: implementing the automated retriever or manually preparing project data based on the interchange format. The first case is the expected scenario of integration to achieve the continuous analysis provided by the framework. The second case is suitable for one-off analysis. In both cases, a necessary step is to map the ARCADE information model to the representation schema used by the target project. We discuss the adaptation of the Atlassian Jira tool to ARCADE in Section 7.1.

Each new project management tool adapted to ARCADE will require a new retriever to be developed. ARCADE provides a reference implementation for the information retriever, developed in Java and tailored to extract data from the Jira REST API (see Figure 27). In case there are multiple information retrievers (for multiple tools), the output of each retriever shall be concatenated as the RDF turtle syntax represents each semantic triple in a text line.

Figure 27 depicts the workflow of information inside the engine.

Figure 27 – Information Retriever internal workflow



Source: The author (2023)

There are some minimum requirements for a project management tool to be adapted for use with ARCADE:

- The tool shall support the unique identification of objects (items, issues etc) that are tracked, and provide a way to associate a type to that object;
- The tool shall support the recording of a relationship between two objects, and provide a way to associate a type or a text field to that relationship;
- The tool shall support the customisation of fields; and
- The tool shall support the automated exporting of object and relationships data.

6.4 CONSISTENCY ANALYSIS

The main objective of an ontology is to give users the answers to the competency questions. However, before delving into the questions, users need to know if there are any inconsistencies regarding the data. A recurring problem in traceability practice is the difficulty of communicating the traceability model and enforcing people to keep it updated (CLELAND-HUANG et al., 2014). Thus, this component features the analysis of the core traceability information provided in the project data retrieved.

Upon loading project data through the Information Retriever or file uploading, the ARCADE evaluates the input data (ABox) against the constraints and rules defined (TBox).

If the engine finds inconsistencies, it explains the data entries and axioms that lead to the inconsistency. The engine presents the explanations showing the data entries and axioms involved separately. The axioms are termed *Rules* and data entries as *Project items* in the web interface to ease understanding by users (see Figure 28).

Figure 28 – Inconsistency Explanation

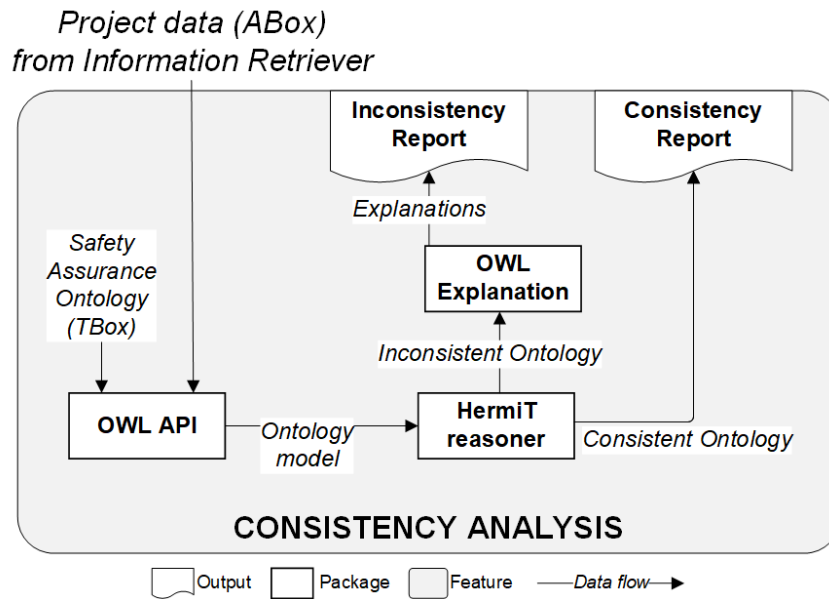
#	Items/Links involved	Rules impacted
Expl. #1	<p>GPCA-1 is a Requirement</p> <p>GPCA-10 refines GPCA-1</p> <p>GPCA-10 is a DesignDefinition</p>	<p>– Everything that refines or is refined by a Requirement, it's also a Requirement.</p> <p>– Project items can be classified to only one of these classes: Component, Context, DesignDefinition, Hazard, Rationale or Requirement (mutually exclusive).</p>

Source: The author (2023)

How is an inconsistency fixed? It is expected that team members using the ARCADE web tool have some level of access to the project management tool. So, they can either fix the inconsistencies directly in the tool or notify those responsible for it. ARCADE does not act back on the management tool to update project items. After correction, the inconsistencies are no longer found the next time the project data is loaded in the framework.

We used the OWL API (PALMISANO, 2023) for manipulating OWL specification and RDF data input, the HermiT reasoner (Data and Knowledge Group, 2023) for ontology reasoning, and the OWL Explanation (HORRIDGE; SPERO, 2023) to compute explanations in case of inconsistencies. Figure 29 depicts the internal workflow of the Consistent Analysis.

Figure 29 – Consistency Analysis internal workflow



Source: The author (2023)

6.5 KNOWLEDGE DISCOVERY

After the consistency analysis, if the ontology and project data are consistent, the framework provides a complete report of provided and inferred project items and links (see Figure 30). The engine computes inferences that are implicit knowledge hidden in the project data. The inferences are based on the rules of formation and classification stated in Table 24. For example, based on Rule 13, if an object A retrieved from project data has a relationship of the type *mitigation* to another object B, then the engine automatically classifies A as a *Safety Requirement*.

This implicit knowledge is needed to support the elaboration of ACs but may not be essential for other development activities. It is up to users to decide if an implicit piece of information (e.g., a hidden class/relationship of an item) should be registered explicitly in the project management tool.

This listing provides the answers for the first two questions (see Table 22), thus providing an overview of the entire knowledge database. The list is sorted by item name, presenting the item attributes (data properties). In the rest of the tool, every mention of items is linked back to this listing for easy navigation.

Additionally, the engine executes queries to gather answers for the other competency questions. *Basic* questions aim to find items of specific classes and are expected to be non-empty.

Figure 30 – Project data provided and inferred

Project Items		Item classification		Item relationships	
Item_ID		provided	inferred	provided	inferred
▼ GPCA-10		SafetyRequirement		mitigates GPCA-19	
Name:	Basal flow rate tolerance	Requirement		isAllocatedIn	
Description:	The pump shall deliver basal infusion at the prescribed basal rate within a basal infusion flow tolerance of tolerance = 0.5 ml/hour of prescribed basal rate.			Component10005	
				mitigates GPCA-7	
				refines GPCA-1	
				isExplainedBy	
				GPCA-10_justification	

Source: The author (2023)

Traceability Gap questions aim to find inconsistencies and are expected to be empty, that is, there should be no project items that fit the query. We decided not to address these traceability gaps as axioms in the OWL specification because these are not strong flaws in the data; they may be related to temporary incompleteness related to the early stages of development. For each question, the answers (if any are found) in the data are presented, alongside an indication if the answer is according to the expectation (empty or non-empty). Figure 31 depicts how the answers are shown for users.

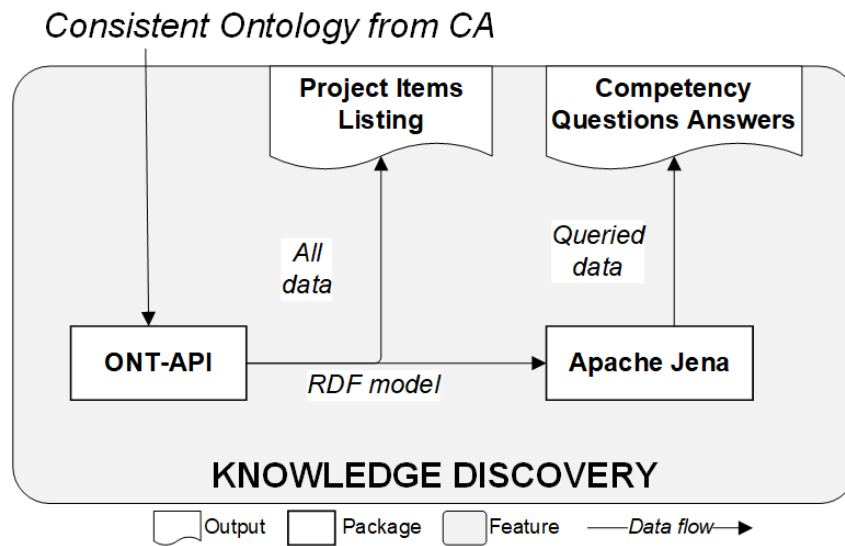
Figure 31 – Answers for competency questions

Traceability Gap Questions	Status	Items founds
CQ13. Which are the Requirements that are not allocated in any Component	Gaps found!	GPCA-1 GPCA-2
CQ14. Which are the Safety Requirements that do not mitigate any Hazard	No gaps found.	

Source: The author (2023)

We used the ONT-API (SSZ et al., 2023) for manipulating OWL axioms as RDF and the Apache Jena for querying the ontology with SPARQL (Apache Software Foundation, 2023). Figure 32 depicts the internal workflow of the Knowledge Discovery component. Source code 3 depicts the SPARQL queries for Basic and Traceability Gap competency questions.

Figure 32 – Knowledge Discovery internal workflow



Source: The author (2023)

Source code 3 – SPARQL queries for two competency questions

```
# Question "Which are the Hazards?"
2 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  PREFIX sao: <dare:arcade/sao#>
4 SELECT ?h WHERE {
    ?h rdf:type sao:Hazard
6     }

# Question "Which are Hazards that are not mitigated by any Safety Requirement?"
8 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  PREFIX sao: <dare:arcade/sao#>
10 SELECT ?h WHERE {
    ?h rdf:type sao:Hazard
12     MINUS {?sr sao:mitigates ?h}
    }
```

Source: The author (2023)

6.6 ASSURANCE CASE GENERATOR

Developers face difficulty selecting the proper project data and structuring the argumentation (HAWKINS et al., 2013). Also, there are many different argumentation patterns for structuring ACs, even though their information source is the same (KELLY; MCDERMID, 1998; SZCZY-GIELSKA; JARZĘBOWICZ, 2018). If developers manually produce an AC using one pattern, they have to create another AC to view the same project data from another perspective.

Using the classified data computed by the other components of the framework, the AC

generator creates multiple AC fragments based on patterns available in the literature. These AC fragments provide users multiple argumentation perspectives based on the same project data.

Each pattern is implemented as a heuristic that queries the ontology model to build the fragment in GSN notation (GOAL..., 2021), a popular argumentation notation used in academy and industry (KELLY, 2018; ALMENDRA et al., 2022). The fragments are generated in GSN notation (GOAL..., 2021), a popular argumentation notation used in academy and industry (KELLY, 2018; ALMENDRA et al., 2022). Figure 33 depicts the design pattern used to structure the AC generator. Each argumentation pattern is implemented as a heuristic embedded in a builder, following the Builder design pattern.⁶ Each pattern has its input needs and expected output, but the common steps are the following:

- Query the ontology data to retrieve relevant semantic triples;
- Iterate over the triples to gather the data;
- Perform complementary queries to the ontology data (if needed);
- Build the parts of the argument during the iteration;
- Build the whole argument in DOT language;⁷ and
- Generate SVG file based on the digraph.

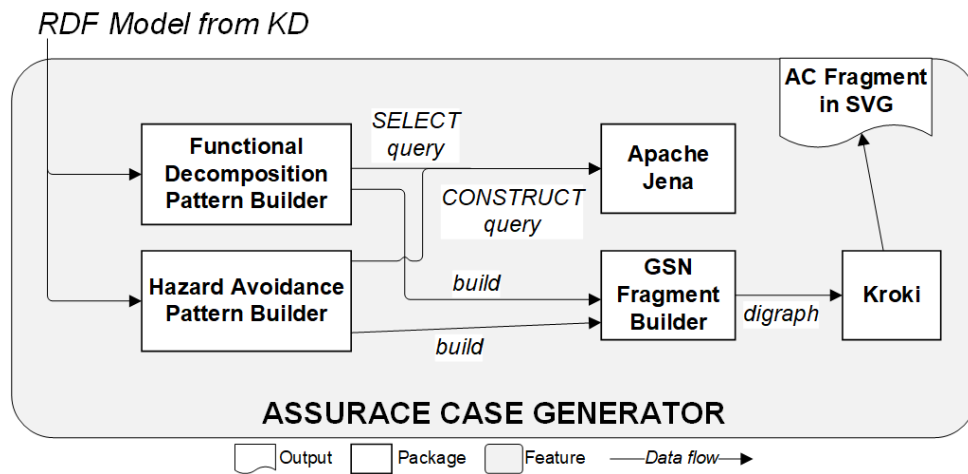
Whenever the engine processes new project data, it generates AC fragments based on the loaded data and the implemented argumentation patterns. Figures 34 and 35 show parts of the fragments generated based on the two patterns currently implemented in ARCADE: Functional Decomposition and Hazard Avoidance (KELLY; MCDERMID, 1998). The fragments were generated for the Open PCA Pump dataset adapted for the ARCADE.⁸ The former breaks down the project data starting from high-level requirements (functionality) that need to be satisfied. At the same time, the latter organises the project data towards the high-level hazards that the system requirements and design must mitigate.

⁶ Builder pattern. Available at: <https://en.wikipedia.org/wiki/Builder_pattern>. Accessed at: July 30, 2023

⁷ DOT (graph description language). Available at: <[https://en.wikipedia.org/wiki/DOT_\(graph_description_language\)](https://en.wikipedia.org/wiki/DOT_(graph_description_language))>. Accessed at: July 30, 2023

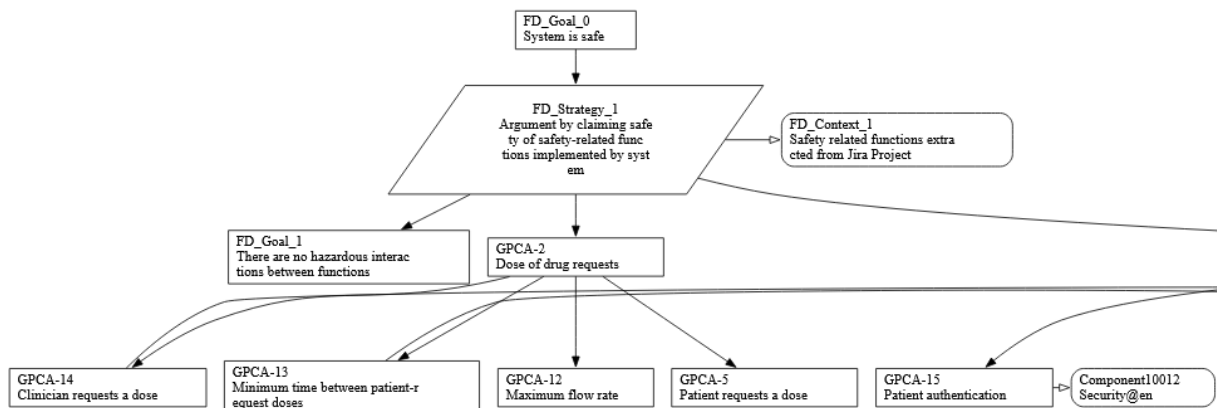
⁸ Open PCA Pump dataset. Available at: <<https://arcade-framework.github.io/openpca/>>. Accessed at: July 30, 2023

Figure 33 – Assurance Case Generator workflow



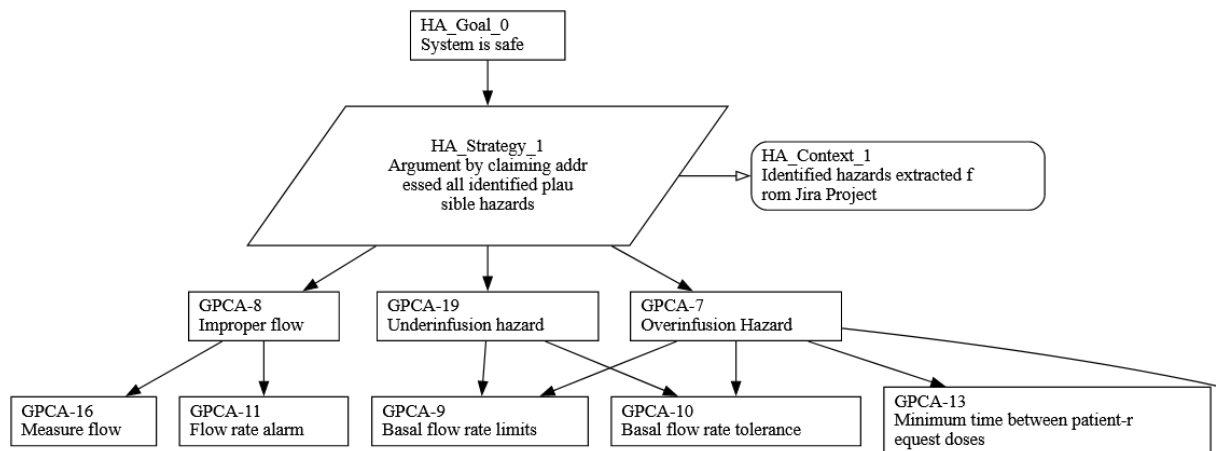
Source: The author (2023)

Figure 34 – Assurance case fragment for Functional Decomposition pattern in GSN notation



Source: The author (2023)

Figure 35 – Assurance case fragment for Hazard Avoidance pattern in GSN notation



Source: The author (2023)

Source code 4 depicts the SPARQL query to retrieve the data needed to produce the Functional Decomposition Pattern. The pattern starts with identifying the root requirements, so the query looks for any requirement that does not refine another.

Source code 4 – SPARQL query for Functional Decomposition Pattern

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  PREFIX sao: <dare:arcade/sao#>
3 SELECT ?r WHERE {
      ?r rdf:type sao:Requirement
5     MINUS {?r sao:refines ?x}
}
```

Source: The author (2023)

6.7 ADOPTING ARCADE

The ARCADE framework was designed to be easily integrated to any enterprise environment. There are two approaches to start using the ARCADE in a project.

In the one-off analysis approach, the project data is extracted and mapped (manually or supported by scripts) to the ARCADE interchange format. This task requires the involvement of people acquainted with the project information model already in use, and people knowledgeable of the ARCADE information model and RDF schema. Once the project data is in ARCADE format, the framework web tool can be used to ingest the data and perform the analysis and assurance case generation.

In the complete automation approach, the Information Retriever feature deployed in the ARCADE reference implementation code needs to be tailored to extract data from the project-specific management tools. If multiple tools are used by the team, each tool will require specific tailoring. Then, the ARCADE framework tool (with the new code of the IR component) is rebuilt and deployed. When a project identifier is entered in the web interface, the data from the project management tool is gathered and the ARCADE features performed.

In Section 7.1, we discussed in detail the steps to integrate the framework with the Jira issue tracker. These steps may vary for other tools.

6.8 CHAPTER SUMMARY

The envisioned benefits of ARCADE framework are the following: **(B1)** Improve team collaboration towards safety assurance; **(B2)** Reduce effort to retrieve and assess assurance information; **(B3)** Reduce effort to build ACs; **(B4)** Increase argumentation quality. ARCADE web tool supports new or improved team activities that leverage these benefits, as detailed in the following.

B1. Instead of managing safety assurance information in separate documents or tools, we support integrated management with project management tools. A unified place for storing, retrieving, and cross-referencing eases traceability management, thus facilitating cross-collaboration across requirements engineering and other activities through the development lifecycle.

B2. Our framework provides automated extraction and analysis, helping users review the project data's consistency and completeness. Also, by means of the interchange format for data input, the framework can work with data retrieved from different management tools.

B3 and B4. The maintenance of standalone ACs may be cumbersome as the project evolves. We argue that there is no strong need to build complete ACs during most of the project lifecycle, and preliminary and interim ACs (KELLY, 2018) could be generated automatically from data available in management tools. Thus, ARCADE provides automated generation of AC fragments in multiple perspectives, which can be useful for safety engineers to assess the quality of evidence throughout the development process. When a complete AC report is needed, the fragment generated can be used as starting point.

7 EVALUATION

7.1 ILLUSTRATIVE INTEGRATION WITH AN ISSUE TRACKER

This illustrative scenario shows how to support our ARCADE Information Model (AIM) using Atlassian Jira (ATLASSIAN, 2023) – the most used agile project management tool nowadays (ONE, 2019; DIGITAL.AI, 2022). Jira provides some predefined issue types to handle software projects and also allows the creation of custom issue types and custom fields.

Tables 26 and 27 depict how Jira was customised to map our information model concepts and relationships, using built-in features and new custom types of issues, issue links and fields. The adaptation comprised new issue types, additional custom fields, new issue link types, and configurations. Once this adaptation is in place, it is possible to record the safety assurance information, thus enriching the existing project information.

Table 26 – Mapping of AIM concepts into Jira.

Model concept	Representation in Jira
Assumption	Assumption field (custom)
Cause	Cause issue (custom)
Component	Component configuration
Context	Context issue (custom)
Design Def.	Design Definition issue (custom)
Hazard	Hazard issue (custom)
Justification	Justification field (custom)
Object	Any issue, labels and components
Requirement	Story issue
Safety Requirement	Story issue with Label “Safety”
Source	Any weblink/attachment associated to an issue
Strategy	Strategy issue (custom)

Source: The author (2023)

Once this adaptation is in place, a team can record assurance information, thus enriching the existing project information. Figure 37 illustrates how Jira elements are filled to comply with project and assurance rationale information of the running example in Figure 36. The main benefit of a customised ITS is to provide an easy way to explicit knowledge alongside the development process.

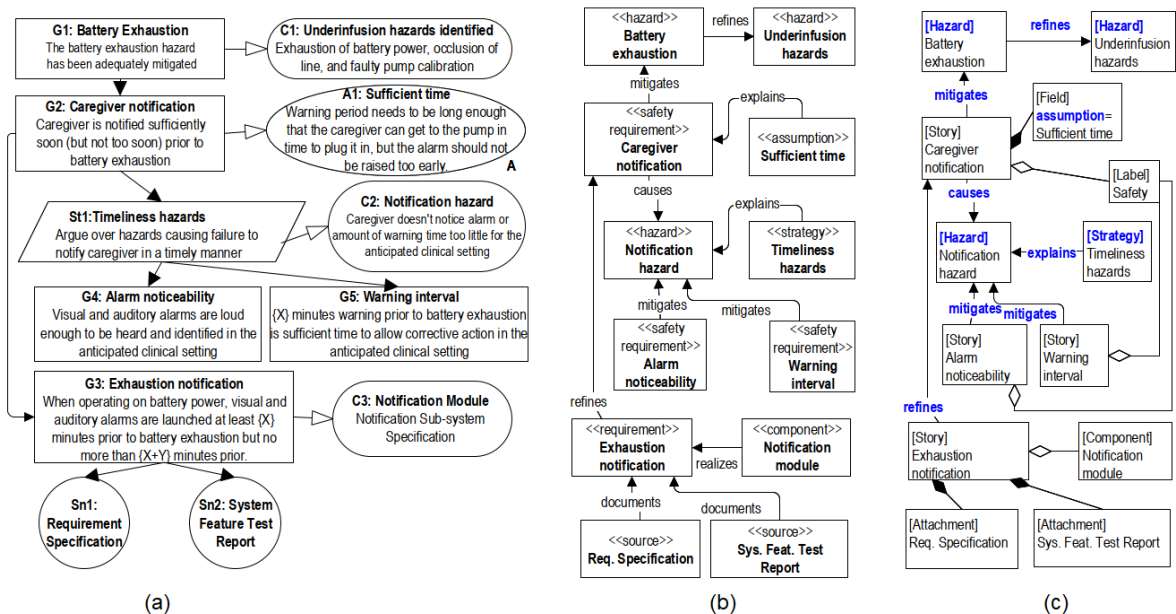
The basic ITS customisation feature to support our model is the option to create custom fields. For example, we may have represented the relationship *mitigates* in another way, creating

Table 27 – Mapping of AIM relationships into Jira.

Model relationship	Representation in Jira
<i>causes</i>	causes issue link (custom)
<i>contextualizes</i>	Context issue linked by relates to to any issue
<i>documents</i>	Weblinks/attachments associated with an issue
<i>explains</i>	explains issue link (custom)
<i>refines</i>	refines issue link (custom)
<i>mitigates</i>	mitigates issue link (custom)
<i>realizes</i>	Component associated to Story issue
<i>implements</i>	Component associated to Design Definition issue

Source: The author (2023)

Figure 36 – (a) Assurance case fragment. (b) Assurance and project data represented in AIM. (c) Same data mapped to Jira.



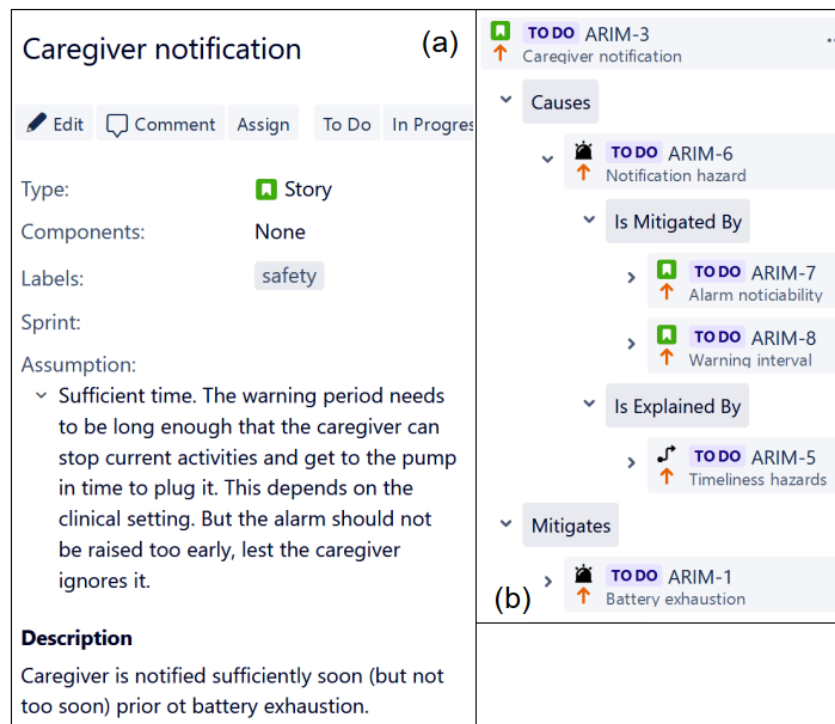
Source: The author (2023)

a custom field with the name “mitigates” which receives an issue identification as value. Custom fields are a very common customisation option in ITSS¹, and there are multiple settings a team could choose to represent their data.

To promote lightweight tailoring, we reused features of Jira, adding meaning to existing relationships. For example, we reused the Jira feature that allows Components to be associated with any issue type. If a Component is associated with a Hazard, it means the Component “causes” the Hazard. Differently, if a Component is associated with a Story (Requirement),

¹ Comparison of issue tracking systems, Wikipedia. Available at: <https://en.wikipedia.org/wiki/Comparison_of_issue-tracking_systems>. Accessed in: May 1, 2023

Figure 37 – (a) Safety requirement represented in Story issue. (b) Issue link tree visualisation.



Source: The author (2023)

it means the Requirement is realised by the Component. This approach has led to more fragmentation in the way the relationships are represented. Thus, reusing existing features may ease the customisation of the Issue Tracking System (ITS) but can confuse teams due to multiple forms of association.

Adapting Jira also exposes recurring mapping patterns, for example, using a custom issue type to represent a concept (e.g., Hazard and Context concept). Another interesting aspect is that we can represent a single concept in multiple ways. Considering the mapping options shown in Table 26, in which we choose to represent Safety Requirements as a Story issue with a label, we could also create a custom issue type “*Safety Story*” and keep the option to label Stories. These two ways of representation may co-exist, and when data is extracted, they both lead to Safety Requirements instances in our information model. This redundancy can be useful if teams want to customise their workspace differently or if there is a need to transition from one form of representation to another without losing pre-existing data.

In Appendix E, we provide some screenshots of the customised Jira instance populated with the dataset we discuss in the next section.

7.2 ILLUSTRATIVE SCENARIO WITH PUBLIC DATASETS

One of the challenges of this thesis work was the access to datasets. Although there are publicly available requirements datasets and examples of Assurance Cases (ACs), there are few datasets comprising safety requirements and the associated ACs. We discuss two public datasets below that were used to evaluate our framework.

7.2.1 Open Patient-Controlled Analgesia Pump Project

The Open PCA Pump Project (OpenPCA) is an open-source, safety-critical medical device project developed to provide a public resource for researchers and practitioners (HATCLIFF et al., 2018). The project illustrates a suite of realistic development artefacts comprising requirements, architecture, source code, tests, risk management, and ACs. An illustration of the available data regarding requirements and ACs is depicted in Appendix C.

To adopt this dataset in our work, we manually extracted the requirements, hazards and traceability information from two artefacts: Open PCA Pump Requirements Specification (LARSON; HATCLIFF, 2018) and the Open PCA Pump Assurance Case (LARSON, 2014). The adapted dataset was recorded in a Jira project, and the representation using the ARCADE interchange format is available at <https://arcade-framework.github.io/openpca>.

The dataset comprised the following provided and inferred data metrics:

- Items provided: 94
- Item links provided: 203
- Item attributes provided: 266
- New classifications for items inferred: 49
- New item links inferred: 105

For all Basic competency questions, ARCADE found at least one item as the answer. For the following Traceability Gap Questions, ARCADE found items with possible inconsistencies:

- Which are the Requirements that are not allocated in any Component?
- Which are the Safety Requirements that do not mitigate any Hazard?

- Which are the Safety Requirements that are not explained by any Rationale?
- Which are the Hazards that are not mitigated by any Safety Requirement?
- Which are the Hazards that are not explained by any Rationale?
- Which are the Contexts that do not contextualize any Object?
- Which are the Hazards that are not contextualized by any Context?

7.2.2 Dronology Project

Dronology Project provides an open-source framework for controlling and coordinating the flight of individual Unmanned Aircraft Systems, formations, and swarms to support applications such as search-and-rescue, surveillance, and scientific data collection. The project establishes a research environment for studying various software and systems engineering aspects for cyber-physical systems — for example, runtime monitoring, safety analysis, and product line development (CLELAND-HUANG; VIERHAUSER; BAYLEY, 2018).

To adopt this dataset in our work, we semi-automatically extracted the requirements, hazards, design definitions, packages (components) and other items from spreadsheets and JSON files available in the dataset. The adapted dataset was recorded in a Jira project, and the representation using the ARCADE interchange format is available at <https://arcade-framework.github.io/drone>.

The dataset comprised the following provided and inferred data metrics:

- Items provided: 155
- Item links provided: 334
- Item attributes provided: 552
- New classifications for items inferred: 64
- New item links inferred: 22

For the following Basic Questions, ARCADE did not find any items (empty answer):

- Which are the Causes?
- Which are the Justifications?

- Which are the Strategies?
- Which are the Sources?

For the following Traceability Gap Questions, ARCADE found items with possible inconsistencies:

- Which are the Requirements that are not allocated in any Component?
- Which are the Safety Requirements that are not explained by any Rationale?
- Which are the Hazards that are not mitigated by any Safety Requirement?
- Which are the Hazards that are not explained by any Rationale?
- Which are the Rationales that do not explain any Object?
- Which are the Hazards that are not contextualized by any Context?

In this section, we demonstrated that the Safety Assurance Ontology encompasses the concepts and relationships available in two public datasets. We also demonstrate that the use of the ARCADE framework could retrieve relevant project information, infer implicit information, and highlight traceability gaps in these datasets. These findings allowed us to adopt these datasets in the empirical evaluation described in the next section.

7.3 EXPERT SURVEY EVALUATION

We conducted a study to evaluate the framework from the point of view of industrial practitioners and researchers. In this section, we detail the study methodology, the results and analysis of the application and the threats to validity.

7.3.1 Expert Survey Methodology

7.3.1.1 Research Questions

This study with experts is classified as **survey research** (EASTERBROOK et al., 2008) based on a **self-administered questionnaire** (KITCHENHAM; PFLEEGER, 2008). The objective is to gather the perception of experts about the suitability of the framework with regard to the

expected benefits outlined in Section 5.1.3. We defined the following Expert Survey Research Questions (ES-RQ) to guide the evaluation of the framework:

ES-RQ1 How useful and easy to use is the ARCADE information model to facilitate the organisation of assurance-relevant information in the regular project repositories?

ES-RQ2 How useful is the ARCADE information model to represent core concepts/relationships related to assurance and project information?

ES-RQ3 How useful and easy to use is it to explore assurance-relevant information (retrieved from the project repository) using the ARCADE web tool?

ES-RQ4 How useful are the ARCADE web tool features (Consistency Analysis, Knowledge Explorer, Assurance Case Generator)?

ES-RQ5 Does the ARCADE web tool bring the expected benefits of our design goals?

- B1 – Improve team communication towards safety assurance
- B2 – Reduce effort to retrieve and assess safety assurance information
- B3 – Reduce effort to build safety assurance cases

We devised a study protocol comprising short video lectures, tasks and opinion questions. The short video lectures serve to leverage some basic concepts on the domain areas involved and to present the framework characteristics and features. The tasks complement the short lectures and provide participants with a better grasp of the framework. Finally, the opinion questions collect experts' judgement on the framework.

7.3.1.2 Short Video Lectures

Our study involves three main topics in the domain of safety/software engineering: issue tracker systems, information models for safety-critical systems, and ACs. We do not limit the practitioner sampling to persons acquainted with all three topics, so we prepare companion material in the form of short lectures to leverage the participants. These short lectures were produced:

- Safety Assurance Information Model;

- Issue Tracking Systems;
- Safety Assurance Cases;
- ARCADE Framework.

Each short lecture comprises presentation slides and a recorded video. The slides were written in English, and the video lectures were spoken in Brazilian Portuguese. The video lectures were published on a YouTube site.² After the pilot application, the final video lectures were published with English subtitles available for non-Portuguese speakers. The video lectures will serve as user documentation of the framework.

The following is a list of tools used to prepare the video lectures:

- OBS Studio³ for screen recording;
- MS Photos⁴ for video editing;
- VLC⁵ and Reaper⁶ for audio/video editing;
- Adobe Podcast Enhance Speech⁷ for voice enhancement; and
- YouTube Studio⁸ for transcription generation/translation.

7.3.1.3 Participant Procedure

We adopted in this survey the Technology Acceptance Model (TAM) evaluation approach, which provides a model for measuring user acceptance of technologies (DAVIS, 1989). TAM supports collecting opinions from two perspectives: Perceived Usefulness and Perceived Ease of Use. Usefulness is defined in TAM as “*the degree to which a person believes that using a particular system would enhance his or her job performance*” (DAVIS, 1989). Ease of Use is defined as “*the degree to which a person believes that using a particular system would be free of effort*” (DAVIS, 1989). According to Davis (1989), these two measurements significantly correlate with technology’s future use.

² Short Lecture – ARCADE Evaluation. Available at: <<https://www.youtube.com/playlist?list=PLGfIBBin8ovbRpYNPw3VMiAGWH5BGBN8K>>. Accessed in: July 1st, 2023

³ <https://obsproject.com>

⁴ <https://www.microsoft.com/en-us/windows/photo-movie-editor>

⁵ <https://www.videolan.org/vlc>

⁶ <https://www.reaper.fm/>

⁷ <https://podcast.adobe.com/enhance>

⁸ <https://studio.youtube.com>

TAM can be applied for studies in which users are briefly introduced to a technological solution, perform tasks, and are then questioned on the usefulness/ease of use of the technology (DAVIS, 1989). In our study, the short lectures serve as learning material, the tasks serve as a hands-on experience, and the questionnaire collects participants' opinions on the perceived usefulness and ease of use of the ARCADE framework.

The participants should go through the short lectures and tasks, and then answer the opinion questions. We decided to interleave tasks and opinion questions throughout the instrument to reduce bias. Also, it could be demanding for participants to interrelate the opinion questions to the tasks performed if the questions were presented only after all tasks.

The participant procedure is divided into two parts. Part I focuses on the ARCADE information model and the use of the adapted issue tracker (Jira), and relates to ES-RQ1 and ES-RQ2. Part II focuses on the ARCADE web tool and overall evaluation of the framework and relates to ES-RQ3, ES-RQ4 and ES-RQ5. The complete instrument distributed to participants is in Appendix G.

Task design. The main motivation for the tasks is to prompt participants to use the components of the framework. However, as the instrument is self-administered, we have to collect data to assess the correctness of the task execution. Therefore, we defined some measurements based on the guidelines for controlled experiments in (JURISTO; MORENO, 2013).

We elaborated tasks in which participants had to identify and list elements in the datasets. For each task, there are known elements in the dataset that match the task. Based on the known elements and the stated elements by participants, we collected and analysed the following metrics for each task:

- CSE – correctly stated elements divided by the total number of known elements of the task;
- ISE – incorrectly stated elements divided by the total number of known elements of the task;
- MSE – missing elements divided by the total number of known elements of the task.

Opinion question design. The opinion questions were planned to be quantitative and qualitative. Each task presents the participant with a scale and with an open-ended comment box. The scales vary depending on the kind of question, as follows (VAGIAS, 2006):

- Level of Difficulty: Very Difficulty=-2, Difficulty, Neutral, Easy, Very Easy=+2.

- Level of Support: Strongly Oppose=-2, Somewhat Oppose, Neutral, Somewhat Favour, Strongly Favour=+2, Undecided=?;
- Frequency of Use: Never Applied=-2, Few Projects, Some Projects, Most of the Projects, Every Project=+2, Don't Know=?.
- Relevance: Not at All=1, Very Little, Somewhat, To a Great Extent=4, Undecided=?.
- Likelihood: Not at All=1, Very Little, Somewhat, To a Great Extent=4, Undecided=?.

7.3.1.4 Pilot Application

We performed a pilot application of the study protocol with two participants. One has large development industry experience in many fields and more recently in the safety critical systems domain (Participant I). The other participant has solid experience in RE research (Participant II). Both participants hold PhD degrees. Table 28 shows their personal profile answers.

Table 28 – Pilot participants' profile

Question	Pilot Partic. I	Pilot Partic. II
How long have you been working on development and/or research of safety-critical systems?	3 to 5 years	1 to 2 years
What is your current main role in the organisation?	Team leader	SE Professor
What is your level of familiarity (knowledge or practice) of Safety Assurance Cases?	Slightly familiar	Moderately familiar
How many projects (concluded or in-progress) that involved Assurance Cases for safety-critical systems have you participated in?	None	1 to 4 projects
What is your level of familiarity (knowledge or practice) with ontology specification?	Not at all familiar	Somewhat familiar
What is your level of familiarity (knowledge or practice) with Description Logic reasoning or use of semantic reasoners?	Not at all familiar	Slightly familiar
What is your level of familiarity (knowledge or practice) in using Product/Application Lifecycle Management and/or Issue Tracking tools?	Extremely familiar	Extremely familiar
What is your level of familiarity (knowledge) in configuring/administering Product/Application Lifecycle Management and/or Issue Tracking tools?	Somewhat familiar	Moderately familiar

Source: The author (2023)

One of the participants noted the absence of a question to collect familiarity with GSN notation, resolved by including it in the final questionnaire.

Regarding the procedure, both participants had session expiration problems and lost data. The pilot was distributed using LimeSurvey. We ran some tests in Google Forms and detected the same pattern. As the tasks and the short lectures made the procedure lengthy, we changed the presentation of the instrument to an editable document (Word format). Each participant filled out the document as they moved through the procedure. This approach avoided the risk of session expiration and loss of data. The informed consent and personal profile were kept as online questionnaires because they could be quickly filled out.

Regarding the length of the overall protocol, both participants expressed concern about the time needed to complete the steps. We decided to shorten the video lectures and exclude some tasks/questions. We shortened the four videos from a total of 76 minutes to about 52 minutes through better planning of speech script and video editing/cutting. We also removed some redundant tasks and removed some questions. For example, the first version of the procedure had tasks and opinion questions related to each Competency Questions (CQs). In the final version, we kept tasks for three CQs and one opinion question for each group of CQs (Basic and Traceability Gap).

7.3.1.5 *Sampling*

This expert survey's target population was researchers and professionals involved in developing critical systems and with any experience/knowledge about RE or ACD. As ARCADE is designed for various roles in the development lifecycle, we do not restrict the population to be knowledgeable of ACD or RE. It is a target population similar to that of the survey we reported in Chapter 4.

The unit of analysis for the survey sample is the practitioner who works on SCS projects. The survey participation is individual and anonymous. Although we did not use an anonymous online questionnaire for the reasons discussed in the pilot application, we took care to avoid linking any personal data of the participants to their responses.

We chose a non-probabilistic sampling, which refers to any approach in which participants are not randomly selected (LINÄKER et al., 2015). We used convenience sampling as a strategy with our collaboration network. We sent an invitation by email to our contacts explaining the evaluation objectives and how to perform it. We also prepared a landing page website⁹ with

⁹ ARCADE Framework Evaluation. Available at: <<https://arcade-framework.github.io/>>. Accessed at: May 1, 2023

more details on the evaluation.

7.3.2 Expert Survey Results and Analysis

In this section, we present each participant's profile and discuss the results of the procedure steps grouped by research questions.

We received very few responses, only four partners. Their profiles are depicted in Table 29. Besides the short sample, they are very representative of the target population. All participants have more than six years of experience in SCS projects and are at least moderately familiar with ALM/ITS tools. They represent the target public of ARCADE, which are people involved in SCS development and acquainted with project management tools, but not necessarily knowledgeable of ACD.

Table 29 – Participants' profile

Question	Partic. 1		Partic. 2		Partic. 3		Partic. 4	
Safety-critical experience	+10 years		+10 years		6 to 10 years		+10 years	
Current role	Researcher		Senior Software Specialist		Engineer		Senior Design Assurance and Quality Engineer	
Assurance Case Projects w/ ACs	Extremely familiar	fa-	Somewhat familiar	fa-	Moderately familiar	fa-	Somewhat familiar	fa-
GSN notation	Extremely familiar	fa-	Not at all familiar		Not at all familiar		Moderately familiar	fa-
Ontology spec.	Extremely familiar	fa-	Moderately familiar	fa-	Slightly familiar		Not at all familiar	
Description logic	Somewhat familiar	fa-	Moderately familiar	fa-	Not at all familiar		Not at all familiar	
ALM/ITS use	Moderately familiar	fa-	Extremely familiar	fa-	Extremely familiar	fa-	Extremely familiar	fa-
ALM/ITS config	Somewhat familiar	fa-	Extremely familiar	fa-	Slightly familiar		Moderately familiar	fa-

Source: The author (2023)

7.3.2.1 ES-RQ1 – How useful and easy to use is the ARCADE information model to facilitate the organisation of assurance-relevant information in the regular project repositories?

The tasks and opinion questions for this ES-RQ are listed in Table 30. Each task refers to a competency question defined in the ARCADE ontology specification (Table 22). In these tasks, participants have to use the Jira web interface to search/explore the issues, as they would if working on a project that manages the assurance and project information together within the issue tracker. For each of the four tasks, there is an associated opinion question with the purpose of collecting the perceived level of difficulty. Finally, the last opinion question of this group asks the level of support of the participants for the overall approach of integrating assurance information with regular project information.

Table 30 – Task and opinion questions for Part I – ES-RQ1

Procedure step	TAM Persp.	Step type	Answer type
1.4.1 Identify and list below which are the Hazards that are not mitigated by any Safety Requirement of the project	Usefulness	Task	Open-ended
1.4.2 How difficult was it to retrieve the information requested in step 1.4.1?	Ease of use	Opinion	Level of difficulty
1.4.3 Identify and list below which are the Safety Requirements that are not explained by any Rationale (Assumption, Justification or Strategy) of the project	Usefulness	Task	Open-ended
1.4.4 How difficult was it to retrieve the information requested in step 1.4.3?	Ease of use	Opinion	Level of difficulty
1.4.5 Regarding the issue OPENPCAB-40, which are the project items that contextualize and provide assurance evidence to it?	Usefulness	Task	Open-ended
1.4.6 How difficult was it to retrieve the information requested in step 1.4.5?	Ease of use	Opinion	Level of difficulty
1.4.7 Regarding the issue OPENPCAB-70, which are the project items that contextualize and provide assurance evidence to it?	Usefulness	Task	Open-ended
1.4.8 How difficult was it to retrieve the information requested in step 1.4.7?	Ease of use	Opinion	Level of difficulty
1.5.1 To what extent do you support the approach to manage safety assurance information together with regular project information inside project management tools?	Usefulness	Opinion	Level of support

Source: The author (2023)

We summarise the tasks and opinion question answers in Table 31 and refer to them in the remainder of this section.

Table 31 – Participant answers for steps 1.4.1 to 1.4.8 (ES-RQ1)

Participant	CE	ME	IE	EU
Task 1.4.1 (KE=1)				Opinion 1.4.2
P1	1.0	.0	.0	+2
P2	.0	1.0	.0	0
P3	1.0	.0	.0	+1
P4	1.0	.0	.0	0
Task 1.4.3 (KE=12)				Opinion 1.4.4
P1	.0	1.0	.0	+1
P2	.0	1.0	.0	-1
P3	1.0	.0	.75	-1
P4	.0	1.0	.25	+2
Task 1.4.5 (KE=6)				Opinion 1.4.6
P1	.0	1.0	.0	+1
P2	.17	.83	.0	-1
P3	.17	.83	.0	0
P4	.5	.5	.0	+1
Task 1.4.7 (KE=3)				Opinion 1.4.8
P1	.0	1.0	.0	+1
P2	.67	.33	.0	0
P3	.67	.33	.0	0
P4	.67	.33	.0	+1
Average	.3	.7	.14	.44

KE – Known elements in the dataset, **CSE** – correctly stated elements divided by KE, **ISE** – incorrectly stated elements divided by KE, **MSE** – missing stated elements divided by KE, **EU** – Ease of use in Likert scale (Very Difficult=-2, Difficult, Neutral, Easy, Very Easy=+2).

Source: The author (2023)

Task 1.4.1 relates to “Which are the Hazards that are not mitigated by any Safety Requirement?” traceability gap competency question, and the opinion question 1.4.2 is about the difficulty to retrieve such data. P2 did not identify the known elements and marked the difficulty as *Neutral*. P1, P3 and P4 find all elements and mark the difficulty as *Very Easy*, *Easy* and *Neutral*, respectively. P4 was able to create a filter in Jira to retrieve the answer and “classified it as *Neutral*, because it was not difficult, but it is also not so straightforward to search”.

Task 1.4.3 relates to “Which are the Safety Requirements that are not explained by any Rationale?” traceability gap competency question, and the opinion question 1.4.4 is about the difficulty to retrieve such data. P1 marked as *Easy* but found no known element. In the free comment, P1 pointed out that “*didn’t manage to find any filter to retrieve only safety*

requirements that fulfil the condition", which indicates an expectation to have available custom filters to retrieve the traceability gaps.

P2 marked as *Difficult* and found no element. Based on their free comment, we observed that the relationships among issues (Story, Hazard, etc.) and rationales (Justification, Assumption and Strategy) were unclear for P2. P3 marked as *Difficult* and found all known elements but indicated some incorrect elements. In the free comment, we observed that the mapping between Requirement and Story issue type was unclear for P3.

P4 marked as *Very Easy* but found no known element and three incorrect elements. P4 described the filter used, and we observed a misunderstanding in the question statement.

Tasks 1.4.5 and 1.4.7 relate to the same Basic competency question ("*Which are the relationships associated with an Object?*") but point to different project items (issues) in the Open PCA dataset. The opinion questions 1.4.6 and 1.4.8 are about the difficulty to retrieve such data. Similarly to the previous task, P1 marked as *Easy* but did not inform any known element for tasks 1.4.5 and 1.4.7.

P2 marked as *Difficult* and *Neutral* for tasks 1.4.5 and 1.4.7, respectively. P3 marked both questions as *Neutral*. For tasks 1.4.5 and 1.4.7, participants P2 and P3 only stated items that were explicitly visible in the "Linked Issues" visualisation in the Jira interface (see Figure 37). For both tasks, they miss components associated, rationales in the issue and items linked to other issues up in the refinement hierarchy.

P4 marked as *Easy* for tasks 1.4.5 and 1.4.7. P4 found half the known elements in the task 1.4.5 and two thirds in task 1.4.7. P4 only missed the implicit elements up in the refinement hierarchy.

Finally, opinion question 1.5.1 is about the overall approach of the ARCADE framework regarding the integration of assurance and project information within the same tool. The answer type is Level of Support, and the participant answers and comments were as follows:

- P1: *Somewhat Favour* with comment, "*Having all the Information in a unified repository is positive.*"
- P2: *Strongly Favour* with comment, "*Safety related requirements should be identified in the safety assessment and included in the system/software requirements for implementation.*"
- P3: *Strongly Favour*.

- P4: *Strongly Favour* with comment, “Based on Jira tool presented it seems to be possible to manage the safety information together with regular project information, and it seems to increase the ability to reuse the project information by linking Jira items.”

Discussion for ES-RQ1 Results. We observed that participants had difficulty retrieving the planned information from the issue tracker. In some tasks, only explicit information was retrieved. This indicated that the issue tracker, although largely used, lacks support to handle implicit relationships among project items. On the other hand, this may indicate that participants did not completely understand the ARCADE information model.

They expected filtering features to retrieve the data, and some participants tried to build filters to retrieve the issues. The availability of custom filters may be an interesting approach to communicate the information model to teams and to support the collaborative traceability management.

Regarding the deviations in P1’s response, P1 probably understands the concept of *assurance evidence* differently from what was designed into our ontology. As P1 has extended experience in safety assurance research and development, this mismatch indicates a point for further investigation.

Besides the difficulties of retrieving data and understanding the way information was organised in the issue trackers, participants demonstrated support for the overall approach.

7.3.2.2 ES-RQ2 – *How useful is the ARCADE information model to represent core concepts/relationships related to assurance and project information?*

The opinion questions for this ES-RQ are listed in Table 32. These questions aim to gather evidence on the utility of the ARCADE information model in terms of terminology coverage (questions 1.5.2 and 1.5.3) and positive and negative aspects (questions 1.5.4 and 1.5.5). Table 33 shows the answers for opinion question 1.5.2.

From the answers in Table 33, we observed that P3’s answers deviate from the others. P3 either works in very specific niches in the development process or is familiar with another terminology. Even after going through the short lectures and the tasks, P3 could not establish a mapping among ARCADE concepts and those of their work experience. P3’s role is Engineer, and the gap between safety engineering and software engineering may explain the mismatch (HATCLIFF et al., 2014).

Table 32 – Opinion questions for Part I – ES-RQ2

Procedure step	TAM persp.	Step type	Answer type
1.5.2 For each concept of the Safety Assurance Information Model, please indicate if it applies to the projects you participated in the last 5 years (for each concept, a frequency is asked)	Usefulness	Opinion	Frequency
1.5.3 From your experience, please indicate any important concept which is not covered in the AIM model	Usefulness	Opinion	Open-ended
1.5.4 Suppose the following scenario: (<i>omitted</i>) In your opinion, which are the positive and the negative aspects of using Jira adapted to the AIM to accomplish the tasks?	Usefulness	Opinion	Open-ended
1.5.5 Do you have any additional comments regarding the use of AIM?	Usefulness	Opinion	Open-ended

Source: The author (2023)

Table 33 – Participant answers for question 1.5.2 (ES-RQ2)

ARCADE Concept	P1	P2	P3	P4	Average
Requirement	+2	+2	0	+2	1.5
Safety Requirement	+2	+2	-2	+2	1.0
Hazard	+2	+2	-2	+2	1.0
Cause (of Hazard)	+2	+2	-2	+2	1.0
Design Definition	+2	+2	+1	+2	1.75
Component	+2	+2	?	+2	2.0
Context	+2	+1	?	+2	1.67
Assumption	+1	+2	+1	+2	1.5
Justification	+1	+2	+1	+2	1.5
Strategy	0	+1	+1	?	.67
Object	+2	+1	?	+2	1.67
Source	+2	+1	?	+2	1.67

Frequency of use in projects= Never=-2, Few=1, Some=0, Most=+1, Every=+2, Don't know=?.

Source: The author (2023)

From P1, P2 and P4 answers, we observed some high frequency of use of fundamental concepts such as Requirement, Safety Requirement, Hazard, Cause, Design Definition and Component. This is expected, as these concepts appear in various regulations (VILELA et al., 2018). The terms we brought from AC domain (Context, Assumption, Justification, Strategy) have mixed frequencies reported, which may relate to the fact that ACs are not demanded by all regulations.

In question 1.5.3, we asked for concepts not covered in the ARCADE information model.

One participant pointed out that ARCADE's focus is on the early requirements and development process, but as safety and assurance engineering pervades the entire product lifecycle, the model should cover more concepts related to evidence types and other kind of artefacts such as *"lifecycle plan"*. In fact, the ARCADE model currently handles any kind of evidence or external artefacts just as *Source*. Another participant indicated the concept of *"Operation"* as important to make the ARCADE model more complete. This information can be managed as a special kind of *"Context"*, but this was not clear to the participant. Two participants did not indicate any missing concept.

In question 1.5.4, we asked for positive and negative aspects of using the adapted Jira in a scenario of producing ACs. The answers are listed below:

- Positive aspects
 - *"Centralised Information management (including traceability management)"*
 - *"Can be useful for small projects, with limited number of requirements"*
 - *"Jira is an well-known tool (faster learning curve)"*
 - *"It can be easily integrated to source control (e.g., Git) making it easier to track requirements into the source code"*
 - *"By creating the traceability using Jira, it is possible to take the advantage of having bidirectional traceability, trust in the data consistency, and use filters to recover data based on specific characteristics."*
- Negative aspects
 - *"Specific filters would be useful"*
 - *"It doesn't ensure correctness-by-construction of ARCADE model instances"*
 - *"I do not know the way Jira is implementing the ARCADE Model will work with thousands of requirements"*
 - *"A generic tool may constrain the application to certain assurance cases (not certain about it)"*
 - *"I don't see clearly a negative aspect of using Jira, maybe the fact that it was necessary to attach some files (e.g., safety analysis), but this could be easily mitigated by adding the reference and version of the artefacts that are created outside Jira."*

For question 1.5.5, we asked for additional comments on the ARCADE information model. We received two answers:

- *“More clearly defining its scope could be useful. In the safety case development example, it seems to me that the requirements specification is not used as evidence in a valid way. It can show that the supported claim was considered, but not that it is true.”*
- *“The way the information is presented is confusing. Different types of objects are presented together. It is difficult see the information organisation. For example, I assumed that when one requirement refines the other, this requirement belongs to a lower level, however this hierarchy is not easy to see. Maybe Jira provides resources to create views.”*

Discussion for ES-RQ2 Results. Our model was designed to be lean and be extended for application in real-world settings (as discussed in Section 5.1.2). We received an indication that the concepts in the current version of ARCADE are useful.

Regarding important concepts not covered, we expected to receive more specific indications of missing concepts. One of the participants indicated a category of concept (evidence types), for which in a recent literature survey more than 40 different types of evidence were found (NAIR et al., 2014). Another participant indicated the concept of operation, for which the Safe-RE metamodel (VILELA et al., 2018) provides the enumeration “Context Type” (with values *Operational* and *Environmental*) to be used as an attribute in “Context” instances.

Bringing all these types and enumerations suggested by the participant may inflate the model. However, the inclusion of some evidence types (the most common) serves to show how the ARCADE is able to handle specific needs in a real-world setting, thus improving the readiness to adopt the framework.

The deviation in the frequency of use of the concepts, observed among one participant to the others, is a good example of the challenge of bridging different areas of the development towards a single repository. In spite of years of experience in safety-critical systems (SCS), practitioners may work in specific roles or parts of a project that may either not manage some kind of artefacts or use different terminology. The adoption of ontology technology can handle this challenge, as we can extend the ontology specification to include different terminologies for the same concept/relationships.

7.3.2.3 *ES-RQ3 – How useful and easy to use is it to explore assurance-relevant information (retrieved from the project repository) using the ARCADE web tool?*

Before this part of the procedure, participants are asked to watch the last short lecture, which presents and demos the ARCADE framework and the web tool. The remainder of the procedure is related to tasks and opinions about the ARCADE web tool. The tasks and opinion questions for this ES-RQ are listed in Table 34. The tasks 2.2.2, 2.2.3, 2.2.5 and 2.2.6 mirror the tasks 1.4.1, 1.4.3, 1.4.5 and 1.4.7 from ES-RQ1. However, they will explore the Dronology dataset for these tasks, instead of the Open PCA dataset for previous tasks. Therefore, participants see how ARCADE brings together the same kind of traceability information they were asked to collect manually in the previous tasks.

Table 34 – Task and opinion questions for Part II – ES-RQ3

Procedure step	TAM Persp.	Step type	Answer type
2.2.2 Identify the Hazards that are not mitigated by any Safety Requirement of the project	Usefulness	Task	Open-ended
2.2.3 Identify the Safety Requirements that are not explained by any Rationale (Assumption, Justification or Strategy) of the project	Usefulness	Task	Open-ended
2.2.4 How difficult was it to retrieve and explore the information requested in steps 2.2.2 and 2.2.3?	Ease of use	Opinion	Level of difficulty
2.2.5 Regarding the project item DRONE-33, which are the project items that contextualize and provide assurance evidence to it?	Usefulness	Task	Open-ended
2.2.6 Regarding the project item DRONE-136, which are the project items that contextualize and provide assurance evidence to it?	Usefulness	Task	Open-ended
2.2.7 How difficult was it to retrieve and explore the information requested in steps 2.2.5 and 2.2.6?	Ease of use	Opinion	Level of difficulty
2.2.8 Explore the assurance case fragments (<i>omitted</i>)	Usefulness	Task	N/A
2.2.9 How difficult was it to retrieve and explore the information requested in step 2.2.8?	Ease of use	Opinion	Level of difficulty

Source: The author (2023)

We summarise the answers for tasks and opinion questions from 2.2.2 to 2.2.7 in Table 35 and refer to them in the remainder of this section.

Task 2.2.2 relates to “*Which are the Hazards that are not mitigated by any Safety Requirement?*” traceability gap competency question, and the opinion question 2.2.4 is about the difficulty to retrieve such data. P1 and P3 identified all the known elements, but P2 referred

Table 35 – Participant answers for steps 2.2.2 to 2.2.7 (ES-RQ3)

Participant	CE	ME	IE	EU
Task 2.2.2 (KE=3)				Opinion 2.2.4
P1	1.0	.0	.0	+2
P2	.0	1.0	.0	+2
P3	1.0	.0	.0	+1
P4	1.0	.0	.0	+2
Task 2.2.3 (KE=13)				Opinion 2.2.4
P1	1.0	.0	.0	+2
P2	1.0	.0	.0	+2
P3	1.0	.0	.0	+1
P4	1.0	.0	.0	+2
Task 2.2.5 (KE=6)				Opinion 2.2.7
P1	.0	1.0	.0	+2
P2	1.0	.0	.0	+2
P3	.83	.17	.0	+1
P4	1.0	.0	.0	+2
Task 2.2.6 (KE=5)				Opinion 2.2.7
P1	.2	.8	.0	+2
P2	1.0	.0	.0	+2
P3	.8	.2	.0	+1
P4	1.0	.0	.0	+2
Average	.86	.14	0	1.75

KE – Known elements in the dataset, **CE** – correctly stated elements divided by total numbers of known elements, **IE** – incorrectly stated elements divided by total numbers of known elements, **ME** – missing elements divided by total numbers of known elements, **EU** – Ease of use in Likert scale (Very Difficult=-2, Difficult, Neutral, Easy, Very Easy=+2).

Source: The author (2023)

to a different competency question in the ARCADE interface. It looked like a minor mistake, as P2 managed to find all the known elements in the subsequent tasks.

For task 2.2.3, which relates to “*Which are the Safety Requirements that are not explained by any Rationale?*” traceability gap competency question, all participants identified all known elements.

P1, P2 and P4 indicated the tasks 2.2.2 and 2.2.3 as *Very Easy* in the opinion question 2.2.4. P3 indicated the tasks as *Easy* and suggested as an improvement the implementation of more elaborate filters. In fact, the current version of the ARCADE web interface comprises a simple HTML page and lacks advanced filters to explore the data.

Tasks 2.2.5 and 2.2.6 relate to the same Basic competency question (“*Which are the relationships associated with an Object?*”) but point to different project items (issues) in the

Dronology dataset. The opinion question 2.2.7 is about the difficulty to retrieve such data. P2 and P4 stated all known elements for these tasks, and P3 only missed one element in each.

Similar to the answers in tasks 1.4.3 and 1.4.5, P1 did not indicate the known elements except for one item correctly stated in task 2.2.6. Based on their comment in question 1.5.3, in which the lack of “*evidence type*” concept was pointed out, we think the wording used in the question statement (“and provide assurance evidence”) was not appropriate; we expected participants to state assurance-related information, not only concrete evidence.

In task 2.2.8, we asked participants to explore the AC fragments and find some elements. Then, we asked the opinion question on the perceived ease of use of the fragments. The answers were as follows:

- P1: *Easy*=+1 with comment, “*Not sure how easy it would for larger safety cases*”;
- P2: *Very Easy*=+2;
- P3: *Very Easy*=+2;
- P4: *Very Easy*=+2;
- Average=1.75.

Discussion for ES-RQ3 Results.

We observed an increase in the total number of correctly stated elements for all tasks using ARCADE (86%) compared with the results of tasks using Jira (30%). We also observed a decrease in missing elements (ARCADE=15%, Jira=70%) and a decrease in incorrectly stated elements (ARCADE=0%, Jira=14%). Regarding the ease of use, the average of opinions in Jira tasks was 0 (*Neutral*), and in the tasks using ARCADE, it was 1.75 (close to *Very Easy*).

For the task 2.2.8 related to the exploration of the AC fragments, the opinion question 2.2.9 average was 1.75. However, one participant was concerned that larger fragments may be difficult to explore. The current visualisation of the fragments in SVG was planned as a form to demo the feature of generating the AC fragments in multiple argumentation patterns. In future work, we plan to generate the fragments in a format to be viewed by a third-party tool.

7.3.2.4 ES-RQ4 – How useful are the ARCADE web tool features (Consistency Analysis, Knowledge Explorer, Assurance Case Generator)?

The opinion questions for this ES-RQ are listed in Table 36. These questions aim to gather evidence on the utility of the ARCADE's competency questions, the consistency analysis and the AC fragment generation. Table 37 shows the answers for these opinion questions.

Table 36 – Opinion questions for Part II – ES-RQ4

Procedure step	TAM Persp.	Step type	Answer type
2.3.1 To what extent do you find relevant the ARCADE feature that organises all project items by their types/-classifications (the “Basic” competency questions)?	Usefulness	Opinion	Relevance
2.3.2 To what extent do you find relevant the ARCADE feature that reveals gaps in traceability (the “Traceability Gaps” competency questions)?	Usefulness	Opinion	Relevance
2.3.3 To what extent do you find relevant the ARCADE feature that checks consistency and generates explanations in case of inconsistent project data?	Usefulness	Opinion	Relevance
2.3.4 To what extent do you find relevant the ARCADE feature that generates AC fragments in multiple perspectives using the same project data?	Usefulness	Opinion	Relevance

Source: The author (2023)

Table 37 – Participants' answers for steps 2.3.1 to 2.3.4 (ES-RQ4)

Question	P1	P2	P3	P4	Average
2.3.1 Relevance of “Basic” competency questions	4	4	4	4	4.0
2.3.2 Relevance of “Traceability Gaps” competency question	4	4	4	4	4.0
2.3.3 Relevance of Consistency Analysis feature	3	4	4	4	3.75
2.3.4 Relevance of Assurance Case Generator feature	3	4	4	4	3.75

Relevance in Likert scale (Not at all=1, Very Little=2, Somewhat=3, To a Great Extent=4).

Source: The author (2023)

Discussion for ES-RQ4 Results.

From the answers in Table 37, we observed that ARCADE was perceived as useful by the participants.

For question 2.3.1, we received the comment: “*ARCADE helps to reunite all the necessary information necessary to perform the analysis and identify the gaps that must be covered, in a very easy and logical way.*” For question 2.3.2, we received the comment: “*This helps ensure*

that no gaps will be missed by the engineer using ARCADE, it is really a supporting tool and not something to automate the analysis.” Both comments are aligned with the objectives of ARCADE.

For question 2.3.3, we received the comment: “*Most of/all the checks would not be needed if ARCADE model instances were created with a tool that ensures correctness-by-construction.*” Truly, a model-based approach for AC development can ensure correctness during the construction of model instances (VARA et al., 2022). However, one motivation of ARCADE is to investigate how we could bring the support of consistency analysis to project data repositories in which we cannot (or do not want to) enforce constraints in the create/update actions.

For question 2.3.4, we received the comment: “*Not sure how useful it will be for larger projects. It seems that the layout could be improved.*” Future versions of the ARCADE web tool need to address support to explore large datasets.

7.3.2.5 ES-RQ5 – Does ARCADE tool bring the expected benefits of our design goals?

The opinion questions for this ES-RQ are listed in Table 38. These questions aim to gather evidence on the perceived likelihood that the adoption of ARCADE could offer the expected benefits. Table 39 shows the answers to these opinion questions.

Table 38 – Opinion questions for Part II – ES-RQ5

Procedure step	TAM Persp.	Step type	Answer type
2.3.5 To what extent do you believe the adoption of the ARCADE Framework could foster the cross-collaboration between safety and software teams towards shared management of safety assurance information?	Usefulness	Opinion	Likelihood
2.3.6 To what extent do you believe the adoption of the ARCADE Framework could reduce the effort to review safety traceability information?	Usefulness	Opinion	Likelihood
2.3.7 To what extent do you believe the adoption of the ARCADE Framework could reduce the effort to develop safety assurance cases?	Usefulness	Opinion	Likelihood
2.3.8 Do you have any additional comments on ARCADE Framework?	N/A	Opinion	Open-ended

Source: The author (2023)

Discussion for ES-RQ5 Results.

From the answers in Table 39, we observed that ARCADE was perceived as beneficial by

Table 39 – Participants' answers for steps 2.3.5 to 2.3.6 (ES-RQ5)

Question	P1	P2	P3	P4	Average
2.3.5 Likelihood to improve cross-collaboration	3	3	3	4	3.25
2.3.6 Likelihood to reduce traceability review effort	3	3	3	4	3.25
2.3.7 Likelihood to reduce effort to develop ACs	3	4	3	4	3.5

Likelihood in Likert scale (Not at all=1, Very Little=2, Somewhat=3, To a Great Extent=4).

Source: The author (2023)

the participants, but not to a great extent.

For question 2.3.5, we received the comment *"It will depend on the way the organisation works and the regulations/standards they have to comply with."* This is a limitation of the current version of the ARCADE, as we did not choose a specific regulation to address thoroughly.

For question 2.3.6, we received the comment *"It can be used to support a requirements review but, unless it is qualified, the reviewer cannot rely on the information provided by the tool."* In the process of customising ARCADE to be used in a real-world setting, the fine-tuning of the information model and rules shall validate the semantics of the traceability with project stakeholders.

For question 2.3.8, we received the following comments:

- *"This initial version of Arcade seems to be very helpful and really improves the Safety Engineer capability of analysing the safety data and creating the safety assurance cases."*
- *"Overall, I think the main ideas are useful. It'd be interesting to study/discuss the benefits beyond the current state of the practice and related tools."*
- *"Not familiar with other competitor framework (if there is any)."*

These last two comments express concern about how the ARCADE framework positions in comparison with competitors and the state of the practice. We position the ARCADE framework as a tool to work alongside already existing project management tools so that teams could improve traceability management and aggregate assurance-related information into these tools. We discuss the related works in Section 2.4.

7.3.3 Expert Survey Threats to Validity

This section discusses the threats to validity concerning our expert survey's planning, design, and execution. We adopted the approach presented by Wohlin et al. (2012).

Construct validity.

In this study, we prompted participants to watch short lectures and perform tasks, so that they could evaluate the characteristics of the ARCADE framework. We balanced the content and length of the lectures, to increase engagement without taking too much time from the participants. Pilot participants reported that the lectures were sufficient to understand the procedure and to use the tools involved (Jira and ARCADE web tool).

In a survey, the quality of the questions ensures that the instrument measures what it intends to measure. As we adopted TAM, our main questions aimed to collect opinions. For each single feature of the ARCADE framework, we co-designed tasks and opinion questions. We interleaved tasks and questions throughout the procedure so that participants could clearly understand for what they were providing opinions. To evaluate the instrument's comprehensibility, pilot participants performed the procedure by themselves without additional mentoring beyond the short lectures.

One limitation is that all tasks correspond to information retrieval activity within the context of a development project. We consider that tasks for creating or updating information could be lengthy or require extended training. We also consider that tasks aiming at the building of ACs would fall into the same problems. A case study would be a more appropriate kind of study to run a comprehensive evaluation of the ARCADE.

The procedure comprises tasks using the Jira and the ARCADE web tools, and the tasks are related to the same competency questions. To avoid learning effects, we used different datasets for Jira and ARCADE tasks.

We cared to leave open-ended comment boxes for all procedure steps, to allow participants to provide complementary information if they wished. All participants have made use of the comment boxes, thus providing qualitative information for our analysis.

External validity.

This survey was designed to evaluate the ARCADE, and it is not meant to generalise claims over the target population. However, it serves as a preliminary evaluation of the solution in the eyes of a highly skilled and experienced group of practitioners.

As the instrument was designed to be self-administered, its replication is feasible. In fact,

this study is still ongoing at the time of this thesis writing as we continue to accept new participants.

Internal validity.

The selection of participants and how to treat their answers may affect the internal validity of a survey. To avoid responses from people outside the expected profile, we clearly explained the desired profile on the landing page website. We also included personal profile questions towards the experience with the many aspects involved in our solution: SCS development, ACD, ontology specification, description logics and issue trackers. The participants' general profile comprises professionals with considerable expertise in SCS development and AC development.

7.4 CHAPTER SUMMARY

In this chapter, we present the evaluation of the ARCADE framework in three ways. We show how ARCADE can be used alongside a popular ITS to support the integrated management of assurance-related and regular project information. We describe two scenarios of adaptation of public datasets that illustrate how ARCADE can support consistency/completeness analysis and generation of AC fragments from project data stored in the ITS.

Finally, we present an expert survey to evaluate the usefulness and ease of use of the ARCADE framework. Our sample is very small, so we could not perform any statistical analysis. Our qualitative assessment based on the preliminary quantitative data and the qualitative opinions is that the ARCADE is perceived as useful and easy to use.

8 CONCLUSIONS

Assurance Case (AC) is a trending approach to communicate and argue about the safety of a system, offering more flexibility to justify requirements and design decisions (BLOOMFIELD; BISHOP, 2010). The development of ACs as part of the certification procedures of a Safety-Critical System (SCS) is increasingly being mandated or recommended by regulations (RINEHART; KNIGHT; ROWANHILL, 2017).

The development of ACs involves the identification and organisation of many types of information output from the development process. It also comprises reviewing the quality of the information items and their traceability (KELLY, 2018; RUSHBY, 2015). Challenges faced by practitioners are deciding which project information is relevant to the argumentation, how to assess the consistency and completeness of traceability, and how to structure the arguments in the AC notations (LANGARI; MAIBAUM, 2013; NAIR et al., 2014; DOSS; KELLY, 2016). Lack of integration between safety engineering and software engineering is also a difficulty reported by practitioners as affecting the ACD activities (LANGARI; MAIBAUM, 2013; MYKLEBUST; HANSEN; LYNGBY, 2017).

The way Assurance Case Development (ACD) is coordinated with the Software Development Life Cycle (SDLC) also impacts the quality of the ACs. The postponement of AC building may lead to loss of design rationale, late discovery of safety requirements or late rework due to impossibility to build a compelling argumentation (HAWKINS et al., 2013; CHENG et al., 2018; KELLY, 2018). Practitioners could better approach ACD as an evolutionary activity (DOSS; KELLY, 2016; KELLY, 2018). However, early AC development is yet to be a consolidated practice (MYKLEBUST; HANSEN; LYNGBY, 2017; ALMENDRA et al., 2022).

Besides, Requirements Engineering (RE) is a crucial process for the production of quality software-intensive SCSs (MARTINS; GORSCHKE, 2016; VILELA et al., 2017a). ACD and RE are disciplines altogether bound, as RE activities deal with key information used in AC arguments, and the disciplines share similarities regarding their lifecycles. In this thesis, we aim to contribute to the intersection of RE and ACD research areas by answering four RQs.

In this chapter, we present the conclusions of this research. In Section 8.1, we summarise RQs answers. In Section 8.2, we express contributions. In Section 8.3, we discuss the limitations of the research. In Section 8.4, we suggest an agenda of future works.

8.1 ANSWERING THE RESEARCH QUESTIONS

RQ1—How does assurance case development interplay with software development? We answered this RQ by conducting a mapping study (Chapter 3) that aimed to understand how the ACD can occur in an integrated way with SDLC. We identified 16 peer-reviewed studies and synthesised them regarding integration approach, roles involved, and expected outcomes. In this study, we observed a lack of works on the integration between RE and ACD.

RQ2—How does assurance case development interplay with requirements engineering? We answered this RQ by conducting the study with practitioners (Chapter 4) that aimed to identify the state of practice of ACD, the existing integration between requirements engineering and ACD, and the practitioners' opinion on this integration. We surveyed 31 practitioners working in 20 different countries through a self-administered online questionnaire. In this study, we observed that the interplay of RE and ACD occurs in practice, although not too frequently and that there is a lack of proper tools to leverage the cross-collaboration.

RQ3—How can assurance information be managed together with project information? We answered this RQ by providing the ARCADE Framework, in particular the Safety Assurance Ontology (Chapter 5.2), to support teams in customising its project management tools to incorporate assurance-related information and associated traceability. We evaluate the ontology through two illustrative scenarios (Sections 7.1 and Section 7.2) in which we customised one popular issue tracker (Jira) and adapted two public datasets.

RQ4—How is the integrated work on project and assurance information supported? We answered this RQ by providing a reference implementation of the ARCADE Framework (Chapter 6) to support teams in managing assurance information integrated with regular project management tools, and to benefit from automated consistency and completeness analysis and AC fragment generation. We evaluated the framework through a survey (Section 7.3) with four practitioner experts.

8.2 CONTRIBUTIONS

The early and continuous development of AC implies the integration of ACD into the SDLC. ACD should permeate all phases of the SDLC, and we were interested in the interplay with early RE activities. In this context, this thesis supports the following contributions:

Understanding the characteristics of incremental ACD approaches: We identified 16 papers dating from 1999 to 2018. We found approaches that integrate ACD into six different SDLC models: Agile, V-model, Incremental, Spiral, Unified and Waterfall. We devised a classification for Task Integration (Parallel and Blended) and a classification for Argument Development (Argument-driven, Scope-driven and Phased). We found studies combining *Agile*, *Blended* and *Scope-driven* characteristics. This combination is a promising approach, fitting ACD into the modern iterative development culture. We also identified developers playing an active role in AC construction (10 out of 16 studies) and also in AC assessment (five studies). This is an interesting finding, as such involvement can help close the gap between software and safety areas.

Understanding the state of the practice and opinions of practitioners towards the integration of RE and ACD: Our survey attracted 31 participants working in 20 different countries, who represent a variety of profiles in terms of the job roles, the roles of organisations, and years of experience.

We observed that the development of ACs is being perceived as beneficial beyond the simple need to pass a certification procedure, as practitioners adopt ACD in no obligatory regulations. Also, our results indicate that ACD is being integrated into SDLC. The mixed observation of parallel and incremental integration approaches suggests that research on processes and tools should seek to support both paradigms combined.

We found evidence of cross-collaboration between requirements and safety teams; for example, AC artefacts are often consulted during RE activities. The RE activities participants indicated as more suitable for ACD integration are Specification and Change request analysis. Participants perceived the integration of RE and ACD as more about promoting discussion towards safety assurance rather than focusing on the final quality of certification artefacts. Most of the participants are confident this integration could be mitigated the problems related to late discovery of safety requirements, late rework due to impossibility to build a compelling AC, and lack of integration between safety engineering and software development teams. There are, however, challenges regarding the level of automation and integrated use of tools to support such integration.

Safety Assurance Ontology: We developed an ontology to support the representation, assessment and classification of assurance and regular project information. The ontology comprises an information model, constraints, rules of formations, and competency questions. The main purpose is to support teams to continuously evaluate the quality of project information

and its traceability.

ARCADE Framework: We designed and developed a reference implementation of the ARCADE Framework. It is implemented as a web system that provides a user interface to demonstrate its features. It can retrieve data automatically from an issue tracker or load data from files using an interchange format. It automates the data retrieval, loading, reasoning, querying of competency questions, presentation of the project data, and the generation of assurance cases fragments in multiple argumentation patterns. The framework aims to support teams to continuously evaluate the quality of project information and its traceability. Also, the ontology provides a basis to allow teams to work together in the same tools, rather than in separate, specialised tools (e.g., requirements, hazard analysis, etc.).

ARCADE Framework evaluation: The framework was evaluated through an illustrative scenario with public datasets, an illustrative adaptation of a project management tools to use the ARCADE information model, and an expert survey evaluation with four practitioners. Regarding the practitioner's evaluation, we could gather evidence that i) our ontology provides a lean information model and set of rules that can be applied to early stages of safety requirements development, ii) our approach to integrated assurance and project information into common management tools is useful and easy to use, and iii) the ARCADE implementation is useful and easy to use.

The evaluation of the ARCADE may serve industry and academy communities in different ways. Practitioners considering the adoption of ARCADE can analyse its positive and negative aspects based on the evaluation of the experts. Academics may address potential improvements in future research. Finally, researchers seeking to evaluate similar solutions may reuse our evaluation procedure.

8.3 LIMITATIONS

We took several measures to mitigate threats to validity in all steps of this thesis; however, limitations still exist.

Regarding the systematic mapping study, the search was conducted in January 2020. Therefore, an update of the study is needed. Additionally, we consider the topic of incremental ACD needs a study with practitioners (interview or survey) to better produce evidence on the state of the practice. This kind of study will confirm or explain evidence collected in the literature studies.

Regarding the practitioners' survey, the sample was low (31) and a replication of the questionnaire with a larger sample would provide more confidence in the results. Also, the questionnaire could be followed by an interview study with practitioners willing to discuss with more depth the state of integration between RE and ACD.

Regarding the ARCADE framework and its components, we understand that real case studies would provide richer and insightful information for evaluating our framework. As the framework is designed to be lean, we can only evaluate the extensibility of its information model, ontology specification, competency questions and web tool when used in a real-world setting. The evaluation of approaches that aim to integrate different roles in the team and permeate different development activities is challenging. We suggest that its evaluation be broken down in short, focused studies, as a way to gather preliminary data before large, overall case studies.

Regarding the integration of assurance-related information within project management tools, the adaptation of other tools different from issue trackers will better provide evidence of the feasibility of our approach.

Regarding the use of semantic web technologies, the OWL reasoning may be a bottleneck in large datasets, so scalability testing should be addressed in future research.

8.4 FUTURE RESEARCH

In the broader topic of ACD and SDLC, we propose further investigation on how practitioners are integrating ACD into the SDLC, and which factors drive their choices of SDLC models, Task Integration and Argument Development approaches. In addition, we see a need for more empirical evidence to map out the adequate profile characteristics of such professionals, as well as their required capabilities and previous education. It would be interesting to investigate the factors that drive the non-mandatory adoption of ACD by organisations. Finally, we consider as a promising research topic the safety accountability of the various roles involved in safety-critical development and certification.

In the topic of RE and ACD, we aim to replicate the self-administered questionnaire and conduct interviews to strengthen the generalisation of the findings. The use of interviews would bring richer data for the open-ended questions. We also seek to investigate different characteristics of the interplay of RE and ACD at the project level, which could leverage how the requirements and assurance practices are used together. In particular, investigating the

challenges and pitfalls of such interplay is an exciting topic to explore. Finally, the expectation towards integrating ACD into requirement specification activities exceeds the current observed practice, indicating a promising research gap to be addressed.

In the wake of the next generation of AI-based approaches that rely on machine learning and large language models, we see the lack of public datasets and limited size of the available ones as challenging constraints to apply AI methods to generate or assess assurance cases from a dataset comprising safety requirements, hazard analysis and system design/implementation.

Practitioners developing safety-critical system and assurance cases lack proper tools to manage and review assurance information, and to map them to assurance case arguments. We implemented the ARCADE framework to support teams with a collaborative place where the interrelations between assurance and regular project information can be explicitly managed together early and continuously throughout the system development lifecycle. It leverages early and continuously the quality assessment of assurance information recorded in the project management tools, as well as the automated generation of assurance cases.

We also consider it important to extend the information model to encompass more sub-types for classes such as Solution (Evidence), Context, Strategy and Requirements. Such extension brings more readiness to adopt the framework and favours the understanding of the information model concepts and relationships.

In the topic of requirements and safety-related integrated evidence management, a promising future research is to investigate how to connect the output of the assurance case fragment generator with input formats of commercial AC tools, to explore the composition of tools around the ARCADE framework. We see a promising research direction to use ODE meta-model as exchange format to leverage the connection of ARCADE to other project tools.

Finally, future works include evaluating the use of ARCADE in industry settings, to assess its adaptability and efficacy to different regulations, projects and teams.

REFERENCES

- ADELARD. *Claims, Arguments, Evidence (CAE) framework*. 2023. Available at: <<https://claimsargumentsevidence.org>>. Accessed on: May 1, 2023.
- AGRAWAL, A.; KHOSHMANESH, S.; VIERHAUSER, M.; RAHIMI, M.; CLELAND-HUANG, J.; LUTZ, R. Leveraging artifact trees to evolve and reuse safety cases. In: IEEE PRESS. *Proc. of the 41st Intl. Conf. on Software Eng.* [S.l.], 2019. p. 1222–1233.
- ALEXANDER, R.; KELLY, T.; GORRY, B. Safety lifecycle activities for autonomous systems development. In: *4th SEAS DTC Technical Conf.* Edinburgh: [s.n.], 2010. p. 1323–1330.
- ALMENDRA, C.; SILVA, C. Managing assurance information: A solution based on issue tracking systems. In: *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*. New York, NY, USA: Association for Computing Machinery, 2020. (SBES '20), p. 580–585. ISBN 9781450387538. Available at: <<https://doi.org/10.1145/3422392.3422454>>.
- ALMENDRA, C.; SILVA, C. ARCADE: A framework for integrated management of safety assurance information. In: *2023 IEEE 31st International Requirements Engineering Conference (RE)*. [S.l.: s.n.], 2023.
- ALMENDRA, C.; SILVA, C.; MARTINS, L. E. G.; MARQUES, J. How assurance case development and requirements engineering interplay: A study with practitioners. *Requirements Engineering*, Springer-Verlag, Berlin, Heidelberg, v. 27, n. 2, p. 273–292, jun 2022. ISSN 0947-3602.
- ALMENDRA, C.; SILVA, C.; VILELA, J. Incremental Development of Safety Cases: a Mapping Study. In: *Proceedings of the 34th Brazilian Symposium on Software Engineering*. New York, NY, USA: ACM, 2020. p. 538–547. ISBN 9781450387538. Available at: <<https://dl.acm.org/doi/10.1145/3422392.3422398>>.
- ALMENDRA, C. C.; BARROS, F.; SILVA, C. Using assurance cases in requirements engineering for safety-critical systems. In: *Anais Estendidos da X Conferência Brasileira de Software: Teoria e Prática*. Porto Alegre, RS, Brasil: SBC, 2019. p. 47–55. ISSN 2177-9384.
- AMASS Consortium. *AMASS User guidance and Methodological Framework D2.5*. [S.l.], 2018. Available at: <<https://www.amass-ecsel.eu/content/deliverables>>.
- AMASS Consortium. *AMASS Assurance and Certification CPS*. 2023. Available at: <<https://www.amass-ecsel.eu/>>. Accessed on: May 1, 2023.
- Apache Software Foundation. *Apache Jena - A free and open source Java framework for building Semantic Web and Linked Data applications*. 2023. Available at: <<https://jena.apache.org/>>. Accessed on: May 1, 2023.
- ARMENGAUD, E. Automated safety case compilation for product-based argumentation. In: *Embedded Real Time Software and Systems (ERTS2014)*. [S.l.: s.n.], 2014.
- ATLASSIAN. *Jira issue tracking system*. 2023. Available at: <<https://www.atlassian.com/software/jira>>. Accessed on: May 1, 2023.
- BAIL, S. *Common reasons for ontology inconsistency*. 2013. Available at: <<https://ontogenesis.knowledgeblog.org/1343/>>. Accessed on: May 1, 2023.

- BARCELLOS, M. P. Towards a framework for continuous software engineering. In: *Proc. of the 34th Brazilian Symposium on Software Engineering*. [S.l.: s.n.], 2020. p. 626–631.
- BECKETT, D.; BERNERS-LEE, T.; PRUD'HOMMEAUX, E.; CAROTHERS, G. *RDF 1.1 Turtle - Terse RDF Triple Language*. [S.l.], 2014. Available at: <<https://www.w3.org/TR/2014/REC-turtle-20140225/>>.
- BISHOP, P.; BLOOMFIELD, R. A Methodology for Safety Case Development. *Safety and Reliability*, v. 20, n. 1, p. 34–42, 2000.
- BLOOMFIELD, R.; BISHOP, P. Safety and assurance cases: Past, present and possible future – an Adelard perspective. In: DALE, C.; ANDERSON, T. (Ed.). *Making Systems Safer*. London: Springer, 2010. p. 51–67. ISBN 978-1-84996-086-1.
- BRICKLEY, D.; GUHA, R. *RDF Schema 1.1*. [S.l.], 2014. Available at: <<https://www.w3.org/TR/rdf11-schema/>>. Accessed on: May 1, 2023.
- BURGE, J. E.; CARROLL, J. M.; MCCALL, R.; MISTRIK, I. Rationale and requirements engineering. In: _____. *Rationale-Based Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 139–153. ISBN 978-3-540-77583-6.
- CENELEC EN 50129 – Railway applications – Communication, signalling and processing systems – Safety related electronic systems for signalling. Belgium, 2018.
- CHENG, J.; GOODRUM, M.; METOYER, R.; CLELAND-HUANG, J. 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*. New York, NY, USA: ACM, 2018. p. 57–60.
- CICOTTI, G. An evidence-based risk-oriented V-model methodology to develop ambient intelligent medical software. *Journal of Reliable Intelligent Environments*, Springer, v. 3, n. 1, p. 41–53, 2017. ISSN 2199-4668.
- CLELAND-HUANG, J.; GOTEL, O. C. Z.; HAYES, J. H.; MÄDER, P.; ZISMAN, A. Software traceability: Trends and future directions. In: *Proceedings of the on Future of Software Engineering*. New York, NY, USA: ACM, 2014. (FOSE 2014), p. 55–69. ISBN 978-1-4503-2865-4.
- CLELAND-HUANG, J.; VIERHAUSER, M. Discovering, Analyzing, and Managing Safety Stories in Agile Projects. *IEEE 26th Intl. Requirements Engineering Conf.*, p. 262–273, 2018.
- CLELAND-HUANG, J.; VIERHAUSER, M.; BAYLEY, S. Dronology: An incubator for cyber-physical systems research. In: *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results*. New York, NY, USA: Association for Computing Machinery, 2018. (ICSE-NIER '18), p. 109–112. ISBN 9781450356626. Available at: <<https://doi.org/10.1145/3183399.3183408>>.
- COUNCIL, N. R. *Software for Dependable Systems: Sufficient Evidence?* Washington, DC: The National Academies Press, 2007. ISBN 978-0-309-10394-7. Available at: <<https://nap.nationalacademies.org/catalog/11923/software-for-dependable-systems-sufficient-evidence>>.
- CÂRLAN, C.; BEYENE, T. A.; RUESS, H. Integrated formal methods for constructing assurance cases. In: *2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. [S.l.: s.n.], 2016. p. 221–228.

Data and Knowledge Group. *HermiT OWL Reasoner*. 2023. Available at: <<http://www.hermit-reasoner.com/>>. Accessed on: May 1, 2023.

DAVIS, F. D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, JSTOR, p. 319–340, 1989.

DEFENCE Standard 00-056 – Safety Management Requirements for Defence Systems – Part 2 – Guidance on establishing a means of complying with part 1. UK, 2017. Issue 5.

DENNEY, E.; PAI, G. A lightweight methodology for safety case assembly. In: SPRINGER. *Computer Safety, Reliability, and Security: 31st International Conference, SAFECOMP 2012, Magdeburg, Germany, September 25-28, 2012. Proceedings 31*. [S.l.], 2012. p. 1–12.

DERMEVAL, D.; VILELA, J.; BITTENCOURT, I.; CASTRO, J.; ISOTANI, S.; BRITO, P.; SILVA, A. Applications of ontologies in requirements engineering: a systematic review of the literature. *Requirements Engineering*, v. 21, n. 4, p. 405–437, Nov 2016. ISSN 1432-010X.

DICK, A.; WILLS, S. Evidence-based development - applying safety engineering techniques to the progressive assurance and certification of complex systems. In: *2008 3rd IET Intl. Conf. on System Safety*. [S.l.: s.n.], 2008. p. 1–6.

DIEBOLD, P.; THEOBALD, S. How is agile development currently being used in regulated embedded domains? *Journal of Software: Evolution and Process*, v. 30, n. November 2017, 2018.

DIGITAL.AI. *16th Annual State of Agile Report*. [S.l.], 2022. Accessed: May 1, 2023. Available at: <<https://digital.ai/resource-center/analyst-reports/state-of-agile-report/>>.

DIMITRIJEVIĆ, S.; JOVANOVIĆ, J.; DEVEDŽIĆ, V. A comparative study of software tools for user story management. *Information and Softw. Technology*, v. 57, p. 352–368, 2015. ISSN 0950-5849.

DOSS, O.; KELLY, T. Challenges and opportunities in agile development in safety critical systems: A survey. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 41, n. 2, p. 30–31, May 2016. ISSN 0163-5948.

EASTERBROOK, S.; SINGER, J.; STOREY, M.; DAMIAN, D. Selecting empirical methods for software engineering research. In: _____. *Guide to Advanced Empirical Software Engineering*. London: Springer, 2008. p. 285–311. ISBN 978-1-84800-044-5.

EBERT, C. Implementing functional safety. *IEEE Software*, v. 32, n. 5, p. 84–89, Sept 2015. ISSN 0740-7459.

Eclipse Foundation. *Eclipse OpenCert*. 2023. Available at: <<https://www.polarsys.org/opencert/>>. Accessed on: May 1, 2023.

FDA — Infusion Pumps Total Product Life Cycle – Guidance for Industry and FDA Staff. USA, 2014.

FERNÁNDEZ, D. M.; WAGNER, S.; KALINOWSKI, M.; FELDERER, M.; MAFRA, P.; VETRÒ, A.; CONTE, T.; CHRISTIANSSON, M.-T.; GREER, D.; LASSENIUS, C.; MÄNNISTÖ, T.; NAYABI, M.; OIVO, M.; PENZENSTADLER, B.; PFAHL, D.; PRIKLADNICKI, R.; RUHE, G.; SCHEKELMANN, A.; SEN, S.; SPINOLA, R.; TUZCU, A.;

- VARA, J. L. D. L.; WIERINGA, R. Naming the pain in requirements engineering. *Empirical Software Engineering*, v. 22, n. 5, p. 2298–2338, oct 2017. ISSN 1382-3256. Available at: <<http://link.springer.com/10.1007/s10664-016-9451-7>>.
- FERNÁNDEZ-LÓPEZ, M.; GÓMEZ-PÉREZ, A.; JURISTO, N. Methontology: from ontological art towards ontological engineering. In: *Proc. AAAI Spring Symposium*. [S.l.]: American Association for Artificial Intelligence, 1997. p. 33–40.
- GADE, D.; DESHPANDE, S. A literature review on assurance driven software design. *Intl. Journal of Advanced Research in Computer and Communication Engineering*, v. 4, n. 9, p. 82–87, 2015.
- GE, X.; PAIGE, R.; MCDERMID, J. An iterative approach for development of safety-critical software and safety arguments. In: *Agile Conf.* [S.l.: s.n.], 2010. p. 35–43. ISBN 9780769541259.
- Generic Infusion Pump Research Project. *The Generic Infusion Pump (GIP) - A workbench for improving safety, security and usability of medical systems*. 2023. Available at: <<https://rtg.cis.upenn.edu/gip/>>. Accessed on: May 1, 2023.
- GLASS, R. L. A structure-based critique of contemporary computing research. *Journal of Systems and Software*, Elsevier, v. 28, n. 1, p. 3–7, 1995.
- GOAL Structuring Notation Community Standard (Version 3). [S.l.], 2021. Available at: <<https://scsc.uk/r141C>>.
- GÓMEZ-PÉREZ, A.; BENJAMINS, R. Overview of knowledge sharing and reuse components: Ontologies and problem-solving methods. In: *IJCAI AND THE SCANDINAVIAN AI SOCIETIES. CEUR WORKSHOP PROCEEDINGS*. [S.l.], 1999.
- GRAYDON, P.; KNIGHT, J.; STRUNK, E. Assurance based development of critical systems. In: *37th Annual IEEE/IFIP Intl. Conf. on Dependable Systems and Networks (DSN'07)*. [S.l.]: IEEE, 2007. p. 347–357.
- GRAYDON, P. J. Formal assurance arguments: A solution in search of a problem? In: *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. [S.l.: s.n.], 2015. p. 517–528.
- HALL, J.; RAPANOTTI, L. Assurance-Driven Design. In: *The Third Intl. Conf. on Software Engineering Advances*. [S.l.: s.n.], 2008. p. 379–388. ISBN 9780769533728.
- HANSSEN, G.; HAUGSET, B.; STÅLHANE, T.; MYKLEBUST, T.; KULBRANDSTAD, I. Quality Assurance in Scrum Applied to Safety Critical Software. In: SHARP, H.; HALL, T. (Ed.). *Agile Processes, in Software Engineering, and Extreme Programming*. Cham: Springer, 2016. p. 92–103. ISBN 978-3-319-33515-5.
- HANSSEN, G.; STÅLHANE, T.; MYKLEBUST, T. Safescrum® additional elements. In: _____. *SafeScrum® – Agile Development of Safety-Critical Software*. Cham: Springer, 2018. p. 109–134. ISBN 978-3-319-99334-8.
- HATCLIFF, J.; LARSON, B.; CARPENTER, T.; JONES, P.; ZHANG, Y.; JORGENS, J. The open pca pump project: an exemplar open source medical device as a community resource. In: *Proceedings of the 2018 Medical Cyber-Physical Systems (MedCPS) Workshop*. [S.l.: s.n.], 2018.

HATCLIFF, J.; WASSYNG, A.; KELLY, T.; COMAR, C.; JONES, P. Certifiably safe software-dependent systems: Challenges and directions. In: *Proc. of the on Future of Software Engineering*. India: ACM, 2014. (FOSE 2014), p. 182–200. ISBN 978-1-4503-2865-4.

HATLICFF, J. *Open PCA Pump Project*. 2019. Available at: <<http://openpcapump.santoslab.org/>>. Accessed on: May 1, 2023.

HAWKINS, R.; HABLI, I.; KELLY, T.; MCDERMID, J. Assurance cases and prescriptive software safety certification: A comparative study. *Safety Science*, v. 59, p. 55–71, 2013. ISSN 0925-7535.

HAWKINS, R.; HABLI, I.; KOLOVOS, D.; PAIGE, R.; KELLY, T. Weaving an assurance case from design: a model-based approach. In: IEEE. *2015 IEEE 16th International Symposium on High Assurance Systems Engineering*. [S.l.], 2015. p. 110–117.

HEEAGER, L. T.; NIELSEN, P. A. A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology*, v. 103, p. 22–39, 2018. ISSN 09505849.

HEIMDAHL, M. Safety and software intensive systems: Challenges old and new. In: IEEE COMPUTER SOCIETY. *2007 Future of Software Engineering*. [S.l.], 2007. p. 137–152.

HOLLOWAY, C. M. Safety case notations: alternatives for the non-graphically inclined. In: *IET Conference Publications*. [S.l.: s.n.], 2008. ISBN 9780863419706.

HORRIDGE, M.; SPERO, S. *OWL Explanation - An API and reference implementation for generating justifications for entailments in OWL ontologies*. 2023. Available at: <<https://github.com/matthewhorridge/owlExplanation/>>. Accessed on: May 1, 2023.

HULDT, T.; STENIUS, I. State-of-practice survey of model-based systems engineering. *Systems Engineering*, Wiley Online Library, v. 22, n. 2, p. 134–145, 2019.

ISO 26262 – Road vehicles – Functional safety – Part 10. Switzerland, 2018.

ISO, I. Systems and software engineering–vocabulary. *ISO/IEC/IEEE 24765: 2010 (E)*, ed, p. 1–418, 2010.

ISO/AWI TS 81001-2-1 – Health software and health IT systems safety, effectiveness and security – Part 2-1: Coordination – Guidance for the use of assurance cases for safety and security. [S.l.], 2021.

ISO/IEC/IEEE 15026-2 – Systems and software engineering – Systems and software assurance – Part 2: Assurance case. [S.l.], 2022.

JURISTO, N.; MORENO, A. M. *Basics of software engineering experimentation*. [S.l.]: Springer Science & Business Media, 2013.

KASUNIC, M. Designing An Effective Survey. *Software Engineering Institute*, 2005.

KELLY, T. *Arguing safety: a systematic approach to managing safety cases*. Phd Thesis (PhD Thesis) — University of York, 1999.

KELLY, T. Safety Cases. In: _____. *Handbook of Safety Principles*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2018. p. 361–385. Available at: <<http://doi.wiley.com/10.1002/9781119443070.ch16>>.

- KELLY, T.; MCDERMID, J. Safety case patterns-reusing successful arguments. In: *IEE Colloquium on Understanding Patterns and Their Application to Systems Engineering*. [S.l.: s.n.], 1998. p. 3/1–3/9.
- KITCHENHAM, B.; CHARTERS, S. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. [S.l.], 2007.
- KITCHENHAM, B. A.; PFLEEGER, S. L. Personal Opinion Surveys. In: *Guide to Advanced Empirical Software Engineering*. London: Springer London, 2008. p. 63–92. ISBN 9781848000438. Available at: <http://link.springer.com/10.1007/978-1-84800-044-5_3>.
- KNIGHT, J.; ROWANHILL, J.; AIELLO, M.; WASSON, K. A comprehensive safety lifecycle. In: KOORNNEEF, F.; GULIJK, C. van (Ed.). *Intl. Ws. on Assurance Cases for Software-Intensive Systems*. [S.l.]: Springer, 2015. p. 38–49. ISBN 978-3-319-24248-4.
- KOKALY, S.; SALAY, R.; CASSANO, V.; MAIBAUM, T.; CHECHIK, M. A model management approach for assurance case reuse due to system evolution. In: *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems*. [S.l.: s.n.], 2016. p. 196–206.
- KRÖTZSCH, M.; SIMANCIK, F.; HORROCKS, I. A description logic primer. *arXiv preprint arXiv:1201.4089*, 2012.
- KURAMITSU, K. Continuously revised assurance cases with stakeholders cross-validation : a DEOS experience. *PeerJ Computer Science*, v. 2, p. e101, 2016.
- LANGARI, Z.; MAIBAUM, T. Safety cases: A review of challenges. *2013 1st International Workshop on Assurance Cases for Software-Intensive Systems, ASSURE 2013 - Proceedings*, p. 1–6, 2013.
- LARSON, B. *Open PCA Pump Assurance Case*. [S.l.], 2014. Not available. Last accessed: 2019-08-07. Available at: <<http://santoslab.org/pub/open-pca-pump/artifacts/Open-PCA-Pump-Safety-Case.pdf>>.
- LARSON, B. R.; HATLICFF, J. *Open Patient-Controlled Analgesia Infusion Pump System Requirements 1.0.0*. Kansas State University, 2018. Accessed: 2019-08-07. Available at: <<https://web.archive.org/web/20160312145247/http://santoslab.org/pub/open-pca-pump/artifacts/Open-PCA-Pump-Requirements.pdf>>.
- LEMPIA, D. L.; MILLER, S. P. *Requirements Engineering Management Handbook*. [S.l.], 2009.
- LEVESON, N. G. *Engineering a safer world: Systems thinking applied to safety*. [S.l.]: The MIT Press, 2016.
- LIEBEL, G.; MARKO, N.; TICHY, M.; LEITNER, A.; HANSSON, J. Model-based engineering in the embedded systems domain: an industrial survey on the state-of-practice. *Software & Systems Modeling*, Springer, v. 17, p. 91–113, 2018.
- LIEBEL, G.; TICHY, M.; KNAUSS, E. Use, potential, and showstoppers of models in automotive requirements engineering. *Software & Systems Modeling*, Springer, v. 18, p. 2587–2607, 2019.

- LIN, C.; SHEN, W. Generation of assurance cases for medical devices. *Studies in Computational Intelligence*, Springer Verlag, v. 566, p. 127–140, 2015. ISSN 1860949X.
- LINÄKER, J.; SULAMAN, S. M.; MELLO, R. Maiani de; HÖST, M. *Guidelines for Conducting Surveys in Software Engineering*. [S.l.], 2015.
- ŁUKASIEWICZ, K.; GÓRSKI, J. Introducing agile practices into development processes of safety critical software. In: *Proc. of the 19th Intl. Conf. on Agile Software Development: Companion*. [S.l.]: ACM, 2018. (XP '18), p. 1–8. ISBN 9781450364225.
- LUTZ, R. R. Software engineering for safety: a roadmap. In: ACM. *Proceedings of the Conference on The Future of Software Engineering*. [S.l.], 2000. p. 213–226.
- MÄDER, P.; JONES, P.; ZHANG, Y.; CLELAND-HUANG, J. Strategic traceability for safety-critical projects. *IEEE Software*, v. 30, n. 3, p. 58–66, May 2013. ISSN 0740-7459.
- MAKSIMOV, M.; FUNG, N.; KOKALY, S.; CHECHIK, M. Two decades of assurance case tools: A survey. In: GALLINA, B.; SKAVHAUG, A.; SCHOITSCH, E.; BITSCH, F. (Ed.). *Intl. Works. on Assurance Cases for Software-Intensive Systems*. [S.l.]: Springer, 2018. p. 49–59.
- MARTINS, L. E. G.; GORSCHKE, T. Requirements engineering for safety-critical systems: A systematic literature review. *Information and Software Technology*, Elsevier, v. 75, p. 71–89, 2016.
- MARTINS, L. E. G.; GORSCHKE, T. Requirements Engineering for Safety-Critical Systems: An Interview Study with Industry Practitioners. *IEEE Transactions on Software Engineering*, v. 46, n. 4, p. 346–361, apr 2020. ISSN 0098-5589. Available at: <<https://ieeexplore.ieee.org/document/8409284/>>.
- MATSUNO, Y.; NAKAZAWA, J.; TAKEYAMA, M.; SUGAYA, M.; ISHIKAWA, Y. Towards a Language for Communication among Stakeholders. In: *2010 IEEE 16th Pacific Rim Intl. Symposium on Dependable Computing*. [S.l.: s.n.], 2010. p. 93–100. ISBN 9780769542898.
- MENG, B.; MOITRA, A.; CRAPO, A. W.; PAUL, S.; SIU, K.; DURLING, M.; PRINCE, D.; HERENCIA-ZAPANA, H. Towards developing formalized assurance cases. In: *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*. [S.l.: s.n.], 2020. p. 1–9.
- MERTEN, T.; MAGER, B.; HÜUBNER, P.; QUIRCHMAYR, T.; PAECH, B.; BÜRSNER, S. Requirements communication in issue tracking systems in four open-source projects. In: *REFSQ Works*. [S.l.: s.n.], 2015. v. 1342, p. 114–125. ISSN 16130073.
- MOLLÉRI, J. S.; PETERSEN, K.; MENDES, E. An empirically evaluated checklist for surveys in software engineering. *Information and Software Technology*, v. 119, p. 106240, mar 2020. ISSN 09505849. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S0950584919302575>>.
- MYKLEBUST, T.; HANSSEN, G.; LYNGBY, N. A survey of the software and safety case development practice in the railway signalling sector. In: *European Safety and Reliability Conf. (ESREL)*. [S.l.: s.n.], 2017. p. 3385–3391. ISBN 9781138629370.
- NAIR, S.; VARA, J. L. de la; SABETZADEH, M.; BRIAND, L. An extended systematic literature review on provision of evidence for safety certification. *Information and Software Technology*, Elsevier B.V., v. 56, n. 7, p. 689–717, jul 2014. ISSN 09505849. Available at:

<<http://dx.doi.org/10.1016/j.infsof.2014.03.001><https://linkinghub.elsevier.com/retrieve/pii/S0950584914000603>>.

NAIR, S.; VARA, J. L. de la; SABETZADEH, M.; FALESSI, D. Evidence management for compliance of critical systems with safety standards: A survey on the state of practice. *Information and Software Technology*, v. 60, p. 1–15, apr 2015. ISSN 09505849. Available at: <<http://www.sciencedirect.com/science/article/pii/S0950584914002560><https://linkinghub.elsevier.com/retrieve/pii/S0950584914002560>>.

NOY, N. F.; MCGUINNESS, D. L. et al. *Ontology development 101: A guide to creating your first ontology*. [S.l.]: Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, Stanford, CA, 2001.

O'HALLORAN, M.; HALL, J.; RAPANOTTI, L. Safety engineering with COTS components. *Reliability Engineering and System Safety*, Elsevier Ltd, v. 160, p. 54–66, 2017. ISSN 09518320.

OMG. *Structured Assurance Case Metamodel*. [S.l.], 2021. Available at: <<https://www.omg.org/spec/SACM/2.2>>.

ONE, C. V. *13th Annual State of Agile Report*. [S.l.], 2019. Accessed: 2019-08-07. Available at: <<https://explore.versionone.com/state-of-agile/13th-annual-state-of-agile-report>>.

PAECH, B.; DELATER, A.; HESSE, T.-M. Supporting project management through integrated management of system and project knowledge. In: *Software Project Management in a Changing World*. [S.l.]: Springer, 2014. p. 157–192. ISBN 978-3-642-55035-5.

PALMISANO, I. *OWL API main repository*. 2023. Available at: <<https://github.com/owlcs/owlapi>>. Accessed on: May 1, 2023.

PAPADOPOULOS, Y.; MCDERMID, J. The potential for a generic approach to certification of safety critical systems in the transportation sector. *Reliability Engineering and System Safety*, Elsevier Ltd, v. 63, n. 1, p. 47–66, 1999. ISSN 09518320.

POPPING, R. Analyzing Open-ended Questions by Means of Text Analysis Procedures. *Bulletin of Sociological Methodology/Bulletin de Méthodologie Sociologique*, SAGE Publications Sage UK: London, England, v. 128, n. 1, p. 23–39, oct 2015. ISSN 0759-1063. Available at: <<http://journals.sagepub.com/doi/10.1177/0759106315597389>>.

RAMESH, B.; JARKE, M. Toward reference models for requirements traceability. *IEEE Transactions on Software Engineering*, v. 27, n. 1, p. 58–93, Jan 2001. ISSN 0098-5589.

RAO, S. B. G. *A Comparative Analysis of Argumentation Languages in the Context of Safety Case Development*. Master's Thesis (Master's Thesis) — SCHOOL OF INNOVATION, DESIGN AND ENGINEERING, VASTERAS, SWEDEN, Jan. 2019.

RINEHART, D.; KNIGHT, J.; ROWANHILL, J. *Current Practices in Constructing and Evaluating Assurance Cases With Applications to Aviation*. [S.l.], 2015. 87 p.

RINEHART, D. J.; KNIGHT, J. C.; ROWANHILL, J. *Understanding What It Means for Assurance Cases to "Work"*. [S.l.], 2017.

RODRÍGUEZ-DAPENA, P. Software safety certification: a multidomain problem. *IEEE Software*, v. 16, n. 4, p. 31–38, 1999. ISSN 0740-7459.

RUIZ, A.; BARBOSA, P.; MEDEIROS, Y.; ESPINOZA, H. Safety Case Driven Development for Medical Devices. In: *Intl. Conf. on Computer Safety, Reliability, and Security (SAFECOMP)*. Cham: [s.n.], 2015. v. 9337, p. 183–196. ISBN 978-3-319-24254-5 978-3-319-24255-2.

RUPARELIA, N. Software Development Lifecycle Models. *SIGSOFT Softw. Eng. Notes*, v. 35, n. 3, p. 8–13, 2010.

RUSHBY, J. The interpretation and evaluation of assurance cases. *Comp. Science Laboratory, SRI Intl., Tech. Rep. SRI-CSL-15-01*, 2015.

SCHEKOTIHIN, K.; RODLER, P.; SCHMID, W.; HORRIDGE, M.; TUDORACHE, T. Test-driven ontology development in protégé. In: *ICBO*. [S.l.: s.n.], 2018.

SCHREIBER, G. Knowledge engineering. In: *Handbook of Knowledge Representation*. [S.l.]: Elsevier, 2008, (Foundations of Artificial Intelligence). chap. 25, p. 929–946.

SIKORA, E.; TENBERGEN, B.; POHL, K. Industry needs and research directions in requirements engineering for embedded systems. *Requirements Engineering*, Springer, v. 17, n. 1, p. 57–78, 2012.

SSZ et al. *ONT-API (OWL-API over Apache Jena)*. 2023. Available at: <<https://github.com/owlcs/ont-api>>. Accessed on: May 1, 2023.

STÅLHANE, T.; MYKLEBUST, T. The agile safety case. *Lecture Notes in Computer Science*, Springer Verlag, Trondheim, Norway, v. 9923 LNCS, p. 5–16, 2016. ISSN 03029743.

STEGHÖFER, J.; KNAUSS, E.; HORKOFF, J.; WOHLRAB, R. Challenges of scaled agile for safety-critical systems. In: *Product-Focused Software Process Improvement*. Cham: Springer, 2019. p. 350–366.

SUÁREZ-FIGUEROA, M. C.; GÓMEZ-PÉREZ, A.; VILLAZÓN-TERRAZAS, B. How to write and use the ontology requirements specification document. In: _____. *On the Move to Meaningful Internet Systems: OTM 2009: Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009, Vilamoura, Portugal, November 1-6, 2009, Proceedings, Part II*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 966–982. ISBN 978-3-642-05151-7.

SZCZYGIELSKA, M.; JARZĘBOWICZ, A. Assurance case patterns on-line catalogue. In: ZAMOJSKI, W.; MAZURKIEWICZ, J.; SUGIER, J.; WALKOWIAK, T.; KACPRZYK, J. (Ed.). *Advances in Dependability Engineering of Complex Systems*. Cham: Springer International Publishing, 2018. p. 407–417. ISBN 978-3-319-59415-6.

TIUSANEN, R. Qualitative risk analysis. In: _____. *Handbook of Safety Principles*. John Wiley & Sons, Inc., 2018. p. 463–492. Available at: <<http://doi.wiley.com/10.1002/9781119443070.ch21>>.

TIWARI, S.; GUPTA, A. A systematic literature review of use case specifications research. *Information and Software Technology*, v. 67, p. 128 – 158, 2015. ISSN 0950-5849.

TOKORO, M. *Open systems dependability: dependability engineering for ever-changing systems*. [S.l.]: CRC press, 2015.

VAGIAS, W. M. Likert-type scale response anchors. *Clemson International Institute for Tourism & Research Development, Department of Parks, Recreation and Tourism Management. Clemson University*, 2006.

VALASKI, J.; REINEHR, S.; MALUCELLI, A. Which roles ontologies play on software requirements engineering? a systematic review. In: THE STEERING COMMITTEE OF THE WORLD CONGRESS IN COMPUTER SCIENCE, COMPUTER ENGINEERING AND APPLIED COMPUTING (WORLDCOMP). *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*. Athens, 2016. p. 24–30.

VARA, J. D. L. Current and necessary insights into SACM: An analysis based on past publications. In: IEEE. *2014 IEEE 7th Intl. Works. on Requirements Engineering and Law*. [S.l.], 2014. p. 10–13.

VARA, J. D. L.; BORG, M.; WNUK, K.; MOONEN, L. An Industrial Survey of Safety Evidence Change Impact Analysis Practice. *IEEE Trans. on Softw. Eng.*, v. 42, n. 12, p. 1095–1117, 2016. ISSN 0098-5589.

VARA, J. D. L.; JIMÉNEZ, G.; MENDIETA, R.; PARRA, E. Assessment of the quality of safety cases: A research preview. In: SPRINGER. *International Working Conference on Requirements Engineering: Foundation for Software Quality*. [S.l.], 2019. p. 124–131.

VARA, J. D. L.; RUIZ, A.; ESPINOZA, H. Recent Advances towards the Industrial Application of Model-Driven Engineering for Assurance of Safety-Critical Systems. In: *Proc. of the 6th Intl. Conf. on Model-Driven Engineering and Software Development, {MODELSWARD} 2018*. [S.l.: s.n.], 2018. p. 632–641. ISBN 9789897582837.

VARA, J. L. D. L.; GARCÍA, A. S.; VALERO, J.; AYORA, C. Model-based assurance evidence management for safety-critical systems. *Software and Systems Modeling*, Springer, v. 21, n. 6, p. 2329–2365, 2022.

VARA, J. L. de la; BORG, M.; WNUK, K.; MOONEN, L. An industrial survey of safety evidence change impact analysis practice. *IEEE Transactions on Software Engineering*, IEEE, v. 42, n. 12, p. 1095–1117, 2016.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHKE, T. Integration between requirements engineering and safety analysis: A systematic literature review. *Journal of Systems and Software*, v. 125, p. 68 – 92, 2017. ISSN 0164-1212.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHKE, T.; SILVA, C. Specifying safety requirements with gore languages. In: *Proceedings of the 31st Brazilian Symposium on Software Engineering*. New York, NY, USA: ACM, 2017. (SBES'17), p. 154–163. ISBN 978-1-4503-5326-7.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHKE, T. Safe-RE: a safety requirements metamodel based on industry safety standards. In: ACM. *Proceedings of the XXXII Brazilian Symposium on Software Engineering*. [S.l.], 2018. p. 196–201.

W3C. *Resource Description Framework (RDF)*. 2023. Available at: <<https://www.w3.org/RDF/>>. Accessed on: May 1, 2023.

W3C. *Semantic Web Standards*. 2023. Available at: <https://www.w3.org/2001/sw/wiki/Main_Page>. Accessed on: May 1, 2023.

WAGNER, S.; FERNÁNDEZ, D. M.; FELDERER, M.; VETRÒ, A.; KALINOWSKI, M.; WIERINGA, R.; PFAHL, D.; CONTE, T.; CHRISTIANSSON, M.-T.; GREER, D.; LASSENIUS, C.; MÄNNISTÖ, T.; NAYEBI, M.; OIVO, M.; PENZENSTADLER, B.; PRIKLADNICKI, R.; RUHE, G.; SCHEKELMANN, A.; SEN, S.; SPÍNOLA, R.; TUZCU, A.; VARA, J. L. D. L.; WINKLER, D. Status Quo in Requirements Engineering: A Theory and a Global Family of Surveys. *ACM Transactions on Software Engineering and Methodology*, v. 28, n. 2, p. 1–48, apr 2019. ISSN 1049-331X. Available at: <<https://dl.acm.org/doi/10.1145/3306607>>.

WEAVER, R. *The Safety of Software – Constructing and Assuring Arguments*. Phd Thesis (PhD Thesis) — University of York, 2003.

WEINSTOCK, C.; GOODENOUGH, J. *Towards an Assurance Case Practice for Medical Devices*. [S.l.], 2009.

WIEGERS, K.; BEATTY, J. *Software requirements*. [S.l.]: Pearson Education, 2013.

WIERINGA, R.; MAIDEN, N.; MEAD, N.; ROLLAND, C. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering*, Springer-Verlag, v. 11, n. 1, p. 102–107, 2006. ISSN 0947-3602.

WILSON, S.; KELLY, T.; MCDERMID, J. Safety case development: Current practice, future prospects. In: SHAW, R. (Ed.). *Safety and Reliability of Software Based Systems*. [S.l.]: Springer London, 1997. p. 135–156. ISBN 978-3-540-76034-4.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: *Proc. of the 18th Intl. Conf. on Evaluation and Assessment in Software Engineering - EASE '14*. [S.l.: s.n.], 2014. p. 1–10. ISBN 9781450324762. ISSN 09505849.

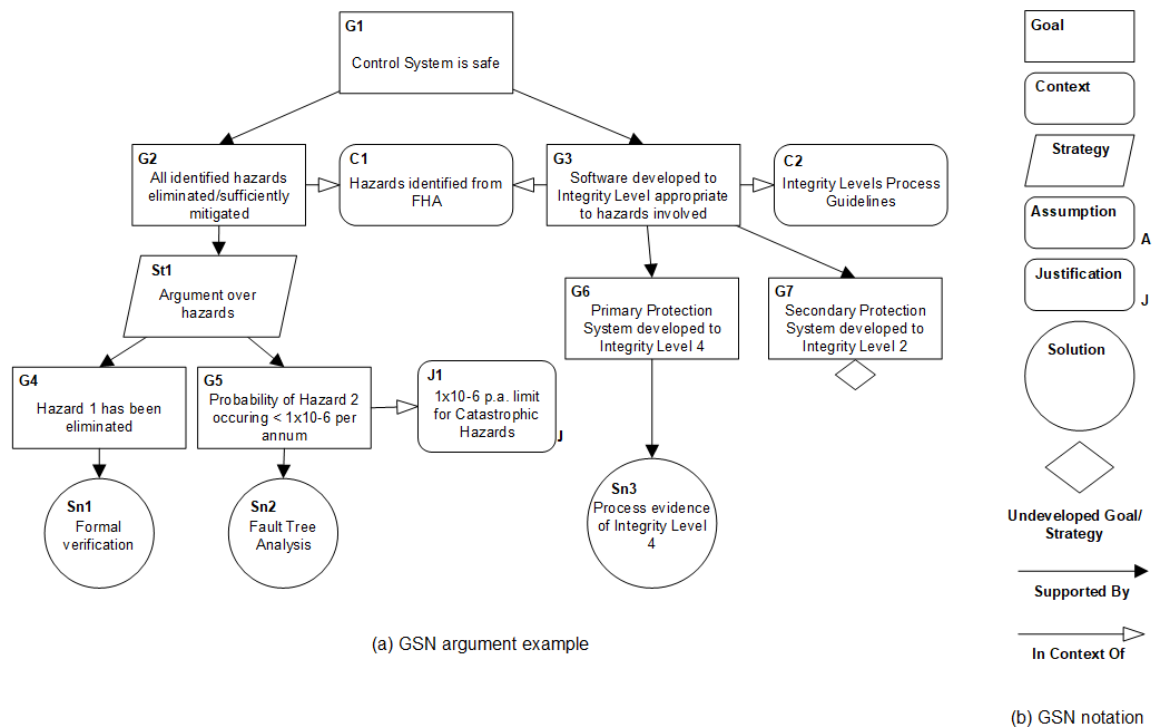
WOHLIN, C.; RUNESON, P.; HÖST, M.; OHLSSON, M. C.; REGNELL, B.; WESSLÉN, A. *Experimentation in software engineering*. [S.l.: s.n.], 2012. ISBN 9783642290442.

ZELLER, M.; SOROKOS, I.; REICH, J.; ADLER, R.; SCHNEIDER, D. Open dependability exchange metamodel: A format to exchange safety information. In: *2023 Annual Reliability and Maintainability Symposium (RAMS)*. [S.l.: s.n.], 2023. p. 1–7.

APPENDIX A – GOAL STRUCTURING NOTATION EXAMPLE

In this appendix, we illustrate the GSN notation through an assurance argument adapted from Weaver (2003). Figure 38 illustrates the assurance case example (a) and the GSN notation (b).

Figure 38 – GSN notation and a safety argument example



Source: adapted from WEAVER (2003)

There is a high-level Goal G1 and its decomposition results in the sub-goals G2 and G3. This decomposition means that the fulfilment of both sub-goals implies the fulfilment of the top goal. Both goals G2 and G3 are in the context of a Functional Hazard Analysis (FHA) (C1) that found a list of hazards associated with the use of the Control System.

Goal G3 is also in the context of a process guidelines specification (C2), meaning that its fulfilment has to comply with the guidelines. Strategy St1 addresses Goal G2, arguing over the elimination or acceptable mitigation of each hazard identified. So, the next level of decomposition is to create separate sub-goals (G4 and G5) that decompose Strategy St1 and address each one as a single hazard.

Goal G4 states the elimination of Hazard H1 from the system, supported by formal verification (Sn1). Goal G5 states the mitigation of Hazard H2, stating that its probability of occurrence is smaller than the specified limit. Justification J1 holds an explanation for this

limit. The evidence solution that supports Goal G5 is the output of a Fault Tree Analysis (Sn2).

Goal G3 decomposes into sub-goals G6 and G7, and the diamond below Goal G7 indicates that this goal is still undeveloped. Goal G6 is supported by process evidence appropriated for the required integrity level (Sn3). This example demonstrates a partially developed safety argument which already has concrete evidence traced to some goals, but yet to be defined how to accomplish others.

APPENDIX B – SURVEY SUPPLEMENTARY MATERIAL

This appendix contains the survey questionnaire presented to participants. It comprises a short introduction and 21 questions structured in four groups.

Survey Introduction

The target audience of this survey is researchers and professionals working on safety-critical systems development projects in which assurance (safety) cases are produced as part of the development process.

Group 1 – Professional Profile

1. How did you find this survey?

- ☐ Post on LinkedIn
- ☐ Invitation via LinkedIn
- ☐ Post on a mailing list
- ☐ Invitation via Email
- ☐ Post on Research Gate
- ☐ Invitation via Research Gate
- ☐ Other - please specify: [open text]

2. How long have you been working on development of safety-critical systems?

- ☐ Less than 1 year
- ☐ 1 to 2 years
- ☐ 3 to 5 years
- ☐ 6 to 10 years
- ☐ More than 10 years

3. What is your current main role in the organisation?

[open text]

4. How many projects that involved Assurance Cases for safety-critical systems have you participated in?

- ☐ Less than 5 projects
- ☐ 5 to 10 projects
- ☐ More than 10 projects

5. In which country(ies) have you principally worked upon Assurance Cases or Requirements Engineering for safety-critical systems?

[open text]

6. What is the main role of the organisation for which you have worked regarding the development of safety-critical systems?

- ☐ Certification authority
- ☐ Component supplier
- ☐ Consultant
- ☐ Developer/manufacturer of final systems
- ☐ Development tool vendor
- ☐ Independent safety assessor
- ☐ Regulation authority
- ☐ Research institution
- ☐ System user
- ☐ Other - please specify: [open text]

Group 2 – Assurance Case Development Practices

Please answer the questions below taking into account your experience in the past 5 years.

7. Is the development of Assurance Cases a mandatory, recommended or optional activity?

Please choose all that apply.

- ☐ Mandatory by regulations
- ☐ Mandatory by my organisation policy/process
- ☐ Recommended by regulations
- ☐ Recommended by my organisation policy/process
- ☐ Optionally performed by the project team
- ☐ Don't know

8. How often have you been involved in these following Assurance Case Development activities?

- Construction
- Review
- Assessment
- Rebuttal
- Approval

For each activity, the options are:

- ☐ Every project
- ☐ Most of the projects
- ☐ Some projects
- ☐ Few projects
- ☐ Never have been involved

9. Which of the following statements apply to how Assurance Cases (ACs) are developed in your projects? Please choose all that apply.

- ☐ ACs are constructed pre-development
- ☐ ACs are reviewed pre-development

- ☐ ACs are constructed during development in a parallel process/lifecycle
- ☐ ACs are reviewed during development in a parallel process/lifecycle
- ☐ ACs are constructed incrementally according to project iterations/sprints/milestones
- ☐ ACs are reviewed incrementally according to project iterations/sprints/milestones
- ☐ ACs are only constructed post-development
- ☐ Other practice - please specify: [open text]

10. Which is the main notation used to construct the Assurance Arguments?

- ☐ ARM
- ☐ ASCAD
- ☐ Bowtie
- ☐ CAE
- ☐ Claims table
- ☐ EUROCONTROL
- ☐ GSN
- ☐ KAOS
- ☐ MDD (MultiMarkdown doc.)
- ☐ Narrative
- ☐ SACM
- ☐ Tabular
- ☐ TRUST-IT Argument Representation
- ☐ WeFA
- ☐ Structured textual
- ☐ Textual (plain text)
- ☐ Other - please specify: [open text]

11. Which of the following scenarios best describe how Assurance Case Reports are organised considering only one system to be certified:

- ☐ One assurance case report containing only one comprehensive assurance argument.
- ☐ One assurance case report containing multiple assurance arguments.
- ☐ Multiple assurance case reports, each containing only one assurance argument each.
- ☐ Multiple assurance case reports, each containing more than one assurance arguments.
- ☐ Other - please specify: [open text]

12. Is there any kind of automation in the creation or maintenance of Assurance Arguments and Reports?

[open text]

Group 3 – Integrated Development Practices

Please answer the questions below taking into account your experience in the past 5 years.

13. Considering the management of traceability among requirements and safety analysis information, which of the following scenarios best describes the most used practice in your projects?

- ☐ A matrix or tool is used to manage only the trace links.
- ☐ A tool is used to manage both artefacts and trace links.
- ☐ Trace links are stated inside the requirements and safety analysis artefacts.
- ☐ Trace links are maintained directly in the certification documents (including assurance cases).
- ☐ Trace links are only recovered to produce final documentation.
- ☐ Other - please specify: [open text]

14. How often are Assurance Cases consulted during Requirements Engineering activities?

- Elicitation
- Analysis and negotiation
- Specification
- Customer/ stakeholder validation
- Change request analysis

For each activity, the options are:

- ☐ Always
- ☐ Very Often
- ☐ Sometimes
- ☐ Rarely
- ☐ Never
- ☐ Don't know

15. How often does your team consider the safety assurance when performing Requirements Engineering activities?

- ☐ Always
- ☐ Very Often
- ☐ Sometimes
- ☐ Rarely
- ☐ Never
- ☐ Don't know

16. Which of the following collaborations between requirements and safety specialists occurs and in which frequency?

- Requirements engineers participate in assurance case development activities.
- Requirements engineers participate in other safety analysis activities.
- Requirements engineers review safety engineering results/artefacts.
- Safety engineers participate in requirements engineering activities.
- Safety engineers review requirements engineering results/artefacts.

For each collaboration scenario, the options are:

- ☐ Every project
- ☐ Most of the projects
- ☐ Some projects
- ☐ Few projects
- ☐ Never
- ☐ Don't know

Group 4 – Personal Opinion on Integration

(Q17 appeared only for those who selected Few projects or above in any of the first three collaboration scenarios of Q16)

17. What are the benefits of Requirements Engineers participation in assurance case development or safety analysis development activities?

[open text]

(Q18 appeared only for those who selected Few projects or above in any of the last two collaboration scenarios of Q16)

18. What are the benefits of Safety Engineers participation in requirements engineering activities?

[open text]

19. How much do you agree with the development of Assurance Cases during Requirements Engineering activities?

- ☐ Strongly agree
- ☐ Agree
- ☐ Undecided
- ☐ Disagree
- ☐ Strongly disagree

20. To what extent do you believe the development of Assurance Cases during Requirements Engineering activities could mitigate the following problems in SCS development?

- Late rework due to impossibility to build a compelling assurance case.
- Poor quality (incompleteness or inconsistency) of assurance cases.
- Loss of requirements and design rationale.
- Late discovery of safety requirements or constraints.
- Lack of integration between safety engineering and software development.

For each problem, the options are:

- ☐ To A Great Extent
- ☐ Somewhat
- ☐ Very Little

- ☐ Not At All
- ☐ Undecided

21. Which Requirements Engineering activities do you think the development or sketching of Assurance Cases could be combined with? Mark all that apply. You may provide an explanation in the open text field.

- ☐ Elicitation: [open text]
- ☐ Analysis and negotiation: [open text]
- ☐ Specification: [open text]
- ☐ Customer/stakeholder validation: [open text]
- ☐ Change request analysis: [open text]

APPENDIX C – OPEN PATIENT-CONTROLLED ANALGESIC PUMP

This section presents an example of Safety-Critical System (SCS) documentation to serve as illustrative system throughout this thesis proposal. We present the Open Patient-Controlled Analgesic Pump (Open PCA), which is a project to provide open source design artefacts for a realistic PCA Pump (HATCLIFF et al., 2018). The project provides artefacts to foster research and standards development, and includes: high level requirements, safety requirements and constraints, system requirements, hazards and architecture design, and an assurance case report. All artefacts are available in the project website (HATLICFF, 2019).

The Open PCA project provides two main artefacts that will be discussed in this section: the Open PCA Infusion Pump System Requirements (LARSON; HATLICFF, 2018) and the Open PCA Pump Assurance Case (LARSON, 2014). The requirements document is the result of extensive work of specification and modelling.

This document is part of a broader set of artefacts meant to illustrate best practices in engineering safety-critical medical devices. Other open source artefacts being developed by KSU SAnToS for the PCA pump include detailed hardware/software architectural descriptions specified in AADL, use cases and requirements modelling with automated traceability to the AADL architecture, formal behavioural specifications in BLESS, and an assurance case for the PCA pump. (LARSON; HATLICFF, 2018, Sec. 1.1)

The assurance case report was specified in NOR-STA tool. There is no evidence in Open PCA project website and documents that the assurance case has been generated entirely or partially from the other project artefacts.

C.1 OVERVIEW OF OPEN PCA PUMP

Patient-controlled analgesia (PCA) is a means for the patient to self-administer analgesics (pain medications) intravenously by using a computerised pump, which introduces specific doses into an intravenous line. PCA uses a computerised pump, which is controlled by the patient through a hand-held button that is connected to the machine. The pump usually delivers medications in small regular doses, and it can be programmed to issue a large initial dose and then a steady, even flow (LARSON; HATLICFF, 2018).

The medical need is to provide medication to dull excruciating pain. Delivering medication as prescribed is what makes a PCA pump effective. Avoiding overdose, and all other harms to

patients, is what makes a PCA pump safe (LARSON; HATLICFF, 2018). Figure 39 depicts an example of a real PCA pump.

Figure 39 – A patient-controlled analgesia infusion pump.

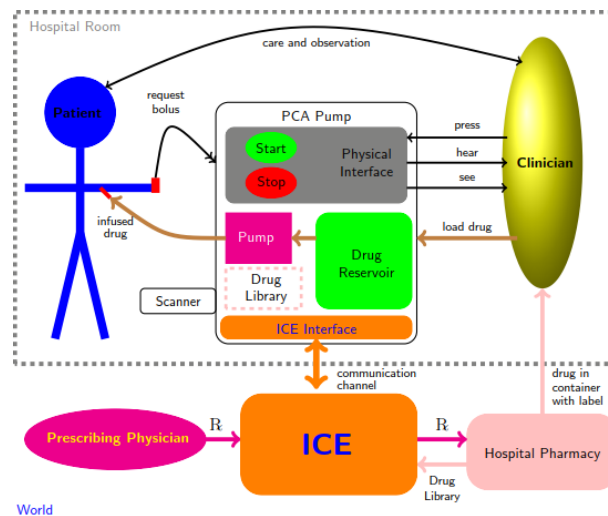


Source: Wikipedia ¹

Open PCA is designed to infuse narcotic, liquid pain-killer at a prescribed basal rate plus any bolus doses that the patient may request to alleviate their pain, or be commanded by an attending clinician (Figure 40). Open PCA is designed for use in an Integrated Clinical Environment (ICE). Thus, the device shall communicate with, and be controlled by an ICE application (LARSON; HATLICFF, 2018).

¹ Available at: <https://en.wikipedia.org/wiki/Patient-controlled_analgesia>. Accessed on: May 1, 2023

Figure 40 – Open PCA pump use



Source: Larson and Hatlicff (2018)

C.2 OPEN PCA SYSTEM REQUIREMENTS

The Open PCA System Requirements (LARSON; HATLICFF, 2018) artefact specifies a set of *System Goals*, *Use Cases*, *Exception Cases* and *Requirements*, following the guidelines in (LEMPIA; MILLER, 2009). There are various links among these concepts, although traceability artefacts are not available in the project. We highlight below the inter-relationships among them, to illustrate how the Open PCA project requirements and architecture information are organised.

C.2.1 Organisation of the Requirements

System Goals are high-level statements about the product. These systems goals must be satisfied for the product to be considered fit-for-use. Although not stated explicitly, seven of the system goals are safety-related, and one is security-related. Example:

- G1 The patient should receive enough drug to reduce his pain.
- G2 The patient should not receive so much drug that makes him unaware, or is harmful.

Use Cases are scenarios that describe how the system is used within the context of its operating environment. They comprise the fields: Title, Description, Related System Goals, Primary Actor, Precondition, Postcondition, Main Success Scenario, Alternate Success Scenario (optional), and Use Case Map. The scenarios are a list of steps representing action and

behaviour of actors and the system. The use case map as just visual representation of the scenario steps. In the step, there are textual references to system functions and architecture components (see Functional Architecture below), as well as linking to other use case or exception cases. Example:

Use Case: Patient-Requested Bolus (UC2)

This use case describes operation when the patient requests an extra dose of drug.

Related System Goals G1 and G2

Primary Actor Patient

Precondition

- Steps 1 to 14 of Normal Operation Use Case completed
- Basal rate being infused
- Prescribed minimum time between boluses has elapsed

Postcondition

- Resume basal rate infusion

Main Success Scenario

1. Patient presses bolus request button
2. Time since last bolus compared with prescribed minimum time between boluses (Exception Case: Bolus Request Too Soon)
3. If not too soon, begin infusing VTBI (Exception Case: Maximum Safe Dose)
4. After prescribed volume-to-be-infused (VTBI) has been infused, resume basal rate infusion

Exception Cases are used to describe exceptional situations or ways in which a use case can fail to meet its goal or postconditions. They have the same fields of the Use Case, except for the Exception Success Scenario. Example:

Exception Case: Bolus Request Too Soon (EC1)

A bolus is requested prior to prescribed minimum time elapsing between boluses.

Related System Goals G2 and G3

Primary Actor Patient or Clinician

Precondition Patient received recent bolus

Postcondition No bolus infused

Exception Success Scenario

1. Check of minimum time between boluses fails (Use Cases: Patient-Requested Bolus or Clinician-Requested Bolus)
2. Control Panel and ICE supervisor user interface (if connected) issue audible warning and display visual warning
3. Warning recorded in Fault Log

Requirements are derived from use and exception cases and stated separately. They are presented organised in four groups (PCA Pump Function, PCA Pump Interfaces, Safety Requirements and Security) and then further divided in subgroups. The requirements listed in

first two groups can be classified as safety-related requirement, since its malfunction could lead to patient harm. Example:

```

5 PCA Pump Function
5.1 Basal Flow Rate
5.2 Patient-Requested Bolus
5.3 Clinician-Requested Bolus
6 PCA Pump Interfaces
6.1 Sensors
6.2 Actuators
6.3 Device Parameters
6.4 Alarms
7 Safety Requirements
...
8 Security
...
```

The requirements comprises a unique identifier, a short title, a statement and references to use or exception cases. Example:

```

R5.2.0(2)(patient-requested bolus): A patient-requested bolus shall be delivered at its
prescribed rate,  $F_{\text{bolus}}$ , in addition to the prescribed basal flow rate,  $F_{\text{basal}}$ , but no more
than the maximum flow rate for the pump,  $F_{\text{max}}$ . (UC2.3)

R5.2.0(3)(minimum time between patient-requested bolus): Patient-requested bolus shall not
be delivered more often than a prescribed minimum time between patient-requested bolus,
 $\Delta_{\text{prb}}$ . (UC2.2)

R6.1.0(2)(detect downstream occlusion): The PCA pump shall detect downstream occlusion. (EC10)

R6.1.0(3)(detect upstream occlusion): The PCA pump shall detect upstream occlusion.
(EC11)
```

C.2.2 Safety Requirements

There are a specific group for “Safety requirements”, which are presented in same manner as of the other requirements. These requirements are mostly related to hardware/physical properties and to the hazard detection and alarming properties. They are also organised in sub-groups:

7 Safety Requirements

- 7.1 Safety Architecture
- 7.2 Anomaly Detection and Response
- 7.3 Power Supply
- 7.4 Diagnostics and Fail-Stop
- 7.5 Tamper-Resistant Door
- 7.6 Biocompatibility
- 7.7 Mechanical

Safety requirements are linked to use and exception, most often to exception cases. There are also some references for normative references (e.g., an IEC standard). The reference is to the standard itself, and not to a specific part or item of the standard. Example:

R7.2.0(4)(upstream occlusion alarm): An upstream occlusion alarm shall be triggered when the pump senses an upstream (drug reservoir side) occlusion exceeding $P_{uo} = 1$ psi. (EC11)

R7.2.0(5)(downstream occlusion alarm): A downstream occlusion alarm shall be triggered if the pump senses a downstream (patient side) occlusion exceeding $P_{do} = 10$ psi. (EC10)

R7.2.0(6)(occlusion alarm): When an occlusion alarm occurs, the pump shall be stopped immediately without completing the current pump stroke. (EC10, EC11)

C.2.3 Functional Architecture

The Open PCA project provides within the Requirements document an overview of the system architecture, components and subsystems. Example:

10 Functional Architecture

...

10.5 Operation Subsystem

...

10.5.9 Patient Bolus Checker

...

10.7 Fluid Subsystem

10.7.1 Pump

10.7.2 Upstream Monitor

10.7.3 Downstream Monitor

10.7.4 Drug Reservoir

For each architecture element, the document provides the allocation of requirements. In the example below, we illustrate 3 subsystems that reference some of the requirements depicted above:

10.5.9 Patient Bolus Checker

The patient bolus checker thread prevents patient-requests bolus delivery sooner than the minimum time between patient-requested bolus.

Allocated Requirements

R5.2.0(3) minimum time between patient-requested bolus

—

10.7.2 Upstream Monitor

The upstream monitor measures drug flow into the pump and detects upstream occlusion.

Allocated Requirements

R6.1.0(1) measure drug flow

R6.1.0(3) detect upstream occlusion

--

10.8.2 Alarm Thread

The alarm thread evaluates fault signals to determine whether the infusion rate should be changed, issues alarm and warning signals to be sounded and displayed by the control panel, and creates fault entries to be stored in the fault log.

Allocated Requirements

...

R7.2.0(4) upstream occlusion alarm

R7.2.0(5) downstream occlusion alarm

R7.2.0(6) occlusion alarm

C.2.4 Traceability Information

The project used automated traceability to manage requirements and architecture (LARSON; HATLICFF, 2018, Sec. 1.1), but it does not released any traceability artefacts for the public. Nonetheless, it is possible to recall some tracing information from the provided artefacts.

Table 40 shows some of tracing information that related requirements and architecture. This information will be helpful to understanding the assurance case report discussed in the next Section.

The hazards considered in the design of the Open PCA Pump follow the are one identified and listed in the US FDA Guidance on Infusion Pumps (FDA..., 2014, Sec. B). There is no

Table 40 – Tracing information among project items.

Items	SG	UC	EC	Re	SR	Co	Rg
System Goal (SG)							
Use Case (UC)	relates	calls	calls			refers	
Excep. Case (EC)	relates					refers	
Requirement (Re)		derives	derives				
Safety Req. (SR)			derives				refers
Component (Co)				allocates	allocates		
Regulation (Rg)							

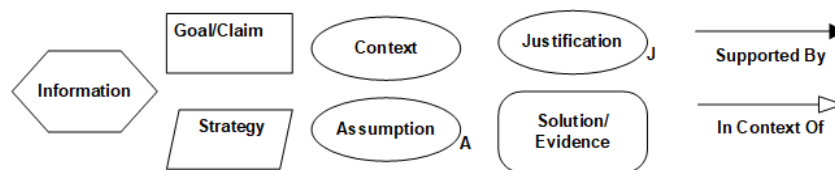
Source: The author (2023)

explicit information on Hazards and Cause of Hazards in the requirements document. However, they are explicitly cited in the Assurance Case Report.

C.3 OPEN PCA ASSURANCE CASE

The project provides an extensive Assurance Case (AC) report (LARSON, 2014). It is not a complete, ready-to-submit for certification boards, but is present a thorough argumentation over the safety properties of the system. The visual notation is the NOR-STA, which is based on GSN (GOAL..., 2021) but with some modifications. Figure 41 show the notation used in the AC report.

Figure 41 – NOR-STA visual notation for assurance arguments.



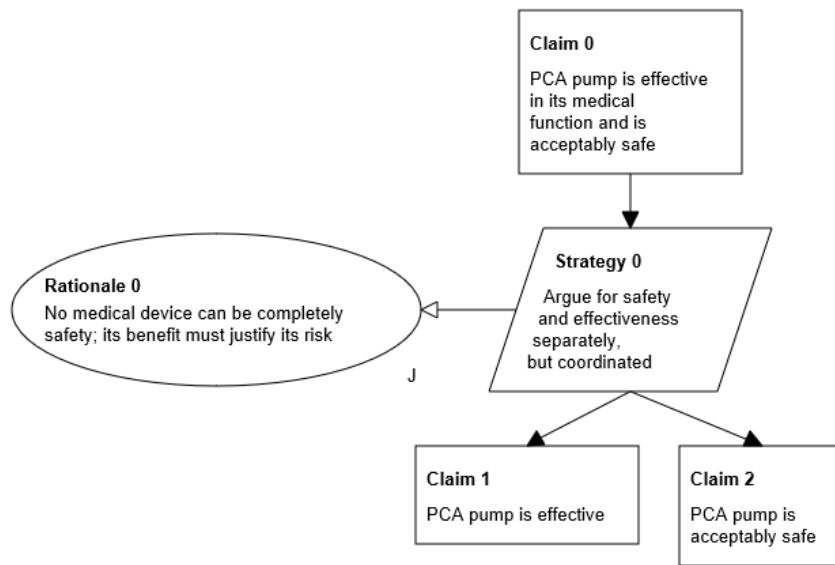
Source: Argevide ²

The main Claim that start the assurance argument hierarchy is depicted in Figure 42.

Claim 0 is satisfied through the *Strategy 0*, which divide the argumentation into two sub-claims: one that the system is effective (*Claim 1*), and other that is safe (*Claim 2*). This reflects the major separation that we see in the requirements documents, in which safety requirements are stated in a separate of other functional requirements.

² GSN assurance cases in NOR-STA. Available at: <<https://www.argevide.com/documents/gsn-assurance-cases-in-nor-sta.pdf>>. Accessed on: May 1, 2023

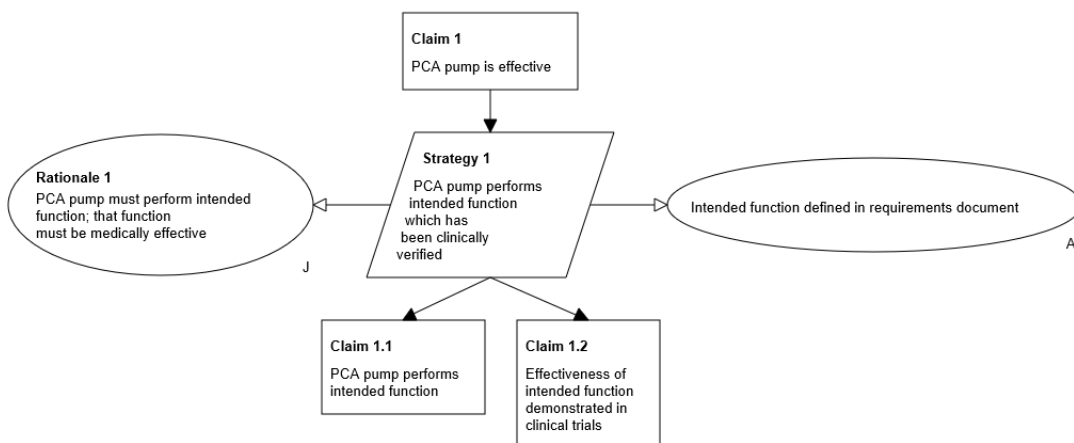
Figure 42 – GSN diagram for Claim 0.



Source: (LARSON, 2014)

The next step in arguing over *Claim 1* is to justify what is considered to be “effective”. The *Claim 1* is satisfied through *Strategy 1* (Figure 43).

Figure 43 – GSN diagram for Claim 1.



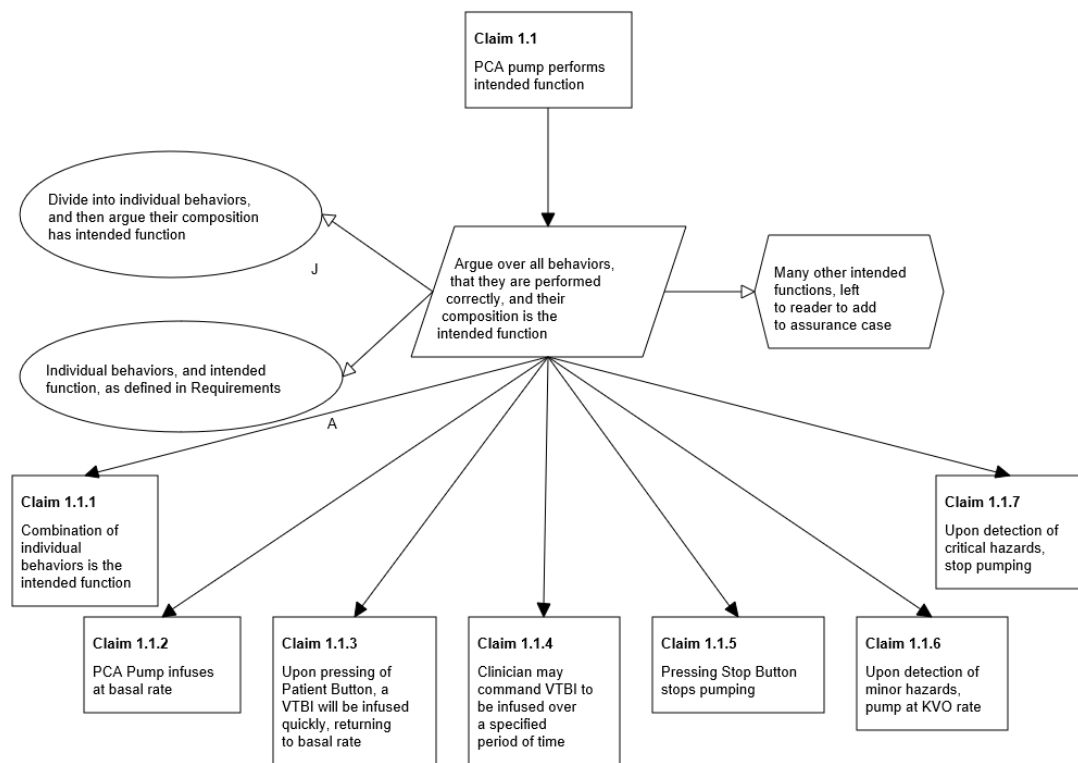
Source: (LARSON, 2014)

Finally, the *Claim 1.1* is decomposed into sub-claims, each one designated to a Requirement (except for the Safety Requirements). The argument is not complete, as we can see in the Information element in the right of the diagram (Figure 44).

The sub-claim *Claim 2* has its satisfaction demonstrated by arguing over all classes of hazards (Figure 45).

Finally, Figure 46 shows how a hazard has its argumentation linked to Requirements,

Figure 44 – GSN diagram for Claim 1.1.



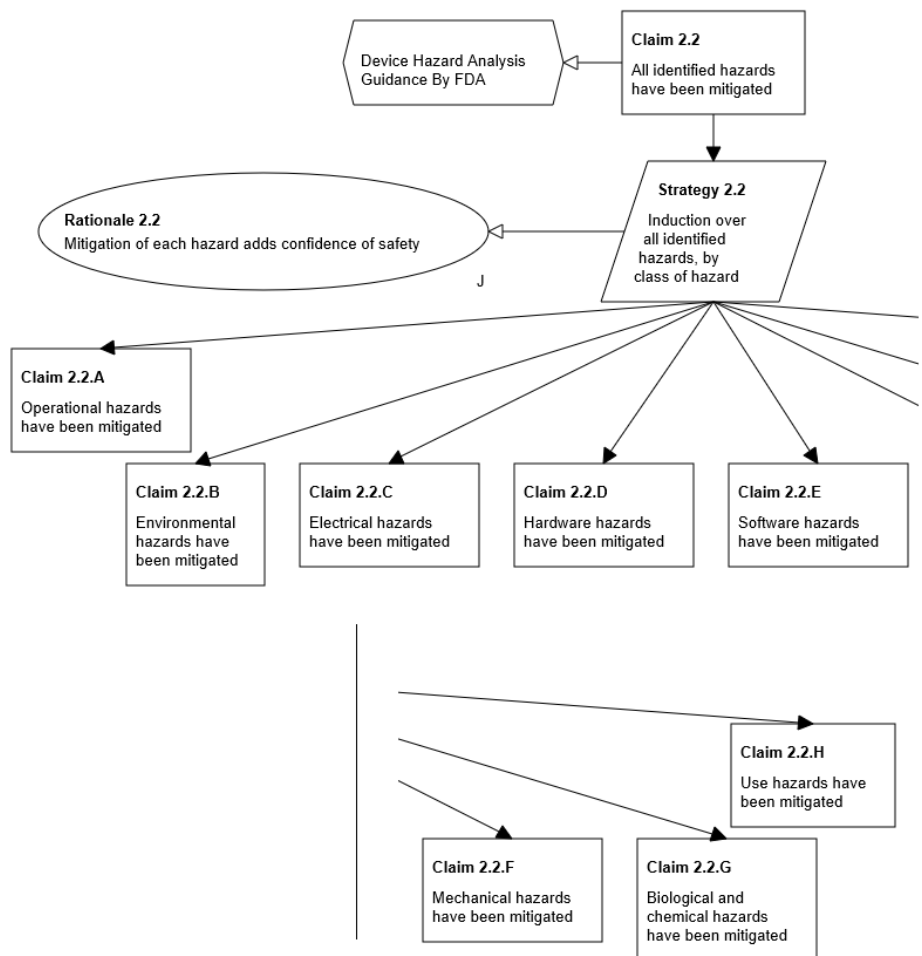
Source: (LARSON, 2014)

Architecture and Testing evidence.

C.3.1 Assurance Case Elements x Project Information

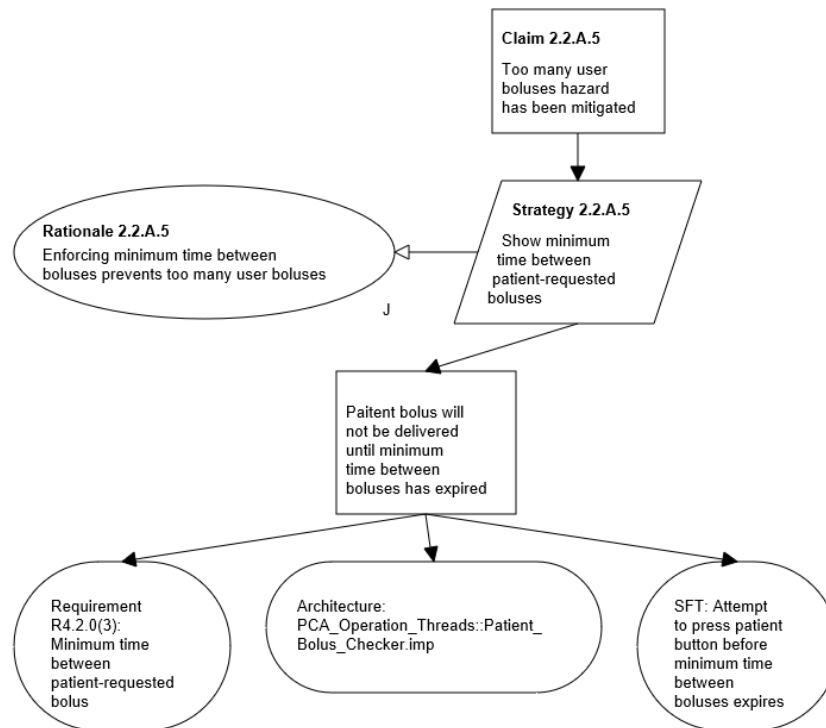
An AC is expected to refer to many types of project management information. We refer to the elements of an assurance case as Argument Element. An argument element may be a reference to a project item or it can be exclusive information not stated elsewhere. Table 41 shows the relationships between argument elements and project items, with examples. Table 42 shows examples of argument elements that brings exclusive information.

Figure 45 – GSN diagram for Claim 2.2.



Source: (LARSON, 2014)

Figure 46 – GSN diagram for Claim 2.2.A.5.



Source: (LARSON, 2014)

Table 41 – Examples of relationships between AC elements and project items.

Reference to Project Item	Example
Argument element: Claim/ Goal	
Requirement	Claim 1.1.2: PCA Pump infuses at basal rate
Mitigation of hazard	Claim 2.2.A.2: Occlusion hazard has been mitigated
Mitigation of class of hazards	Claim 2.2.A: Operational hazards have been mitigated
Argument element: Strategy	
Combination of two requirements to support claim	Strategy 2.2.A.2: Detect occlusion; stop pump
Re-statement of a claim (requirement)	Strategy 2.2.A.2.2: Pump stops when commanded to do so
Elimination of a hazard	Strategy 2.2.A.3: Show pump is incapable of free-flow
A hardware decision mitigates a hazard	Strategy 2.2.A.7: Argue drug leakage minimised by competent mechanical engineering
Argument element: Evidence/ Solution	
Entire requirements document	(Requirements) Draft 0.11
Requirement item	Requirement: R5.2.0(6) Alarm Stops Patient-Requested Bolus
Section of normative reference	Table 2 – Environmental Hazard Examples ³
Subsystem	PCA_Alarm
Component	Architecture: PCA_Operation_Threads:: Patient_Bolus_Checker.imp
Implementation	
STF	
Review?	

Source: The author (2023)

Table 42 – Examples of argument elements that bring new information.

Description	Example
Argument Element: Strategy	
Argumentation strategy for the case	Strategy 0: Argue for safety and effectiveness separately, but coordinated
Explain the rationale for an requirements decision	Strategy 2.2.A.1.1: Stopping pump prevents air in line from entering patient
Argument Element: Justification	
Explain the rationale for an requirements decision	Rationale 2.2.A.5: Enforcing minimum time between boluses prevents too many user boluses
Explain the rationale for an architecture decision	Rationale 2.2.D.1: Separate safety architecture detects and mitigates faults in operation
Explain a process decision	Rationale 2.2.E.5: FDA Quality System Regulation requires proper version control
Argument Element: Assumption	
Process assumption	Intended function defined in requirements document
Process assumption	The Requirements define intended function
Argument Element: Information (used as Context)	
Reference to relevant information in external document	External Infusion Pumps are FDA Class II Devices

Source: The author (2023)

C.4 DISCUSSION

What is actually traced into ACs. In the AC report there are references to Requirements, Safety Requirements, Hazards, Architecture, Components, Test results, Inspection results, Field tests, Risk analyses and Correctness proofs. There are, then, **product** and **process-related** project information that made into the AC.

Missing or incorrect information. The set of artefacts is provided to support research and educational purposes, so there is missing or not detailed information that could not be regarded as documentation errors. There are also some mistakes (e.g., wrong numbering of requirements) that need to be disregarded as fault from the producers of the artefacts.

Re-statements. It is worth noting that the AC report shows some project information re-stated (literally or with minor changes in words). The AC report was developed using the NORSTA tool, but it was a manual development. Even with the use of automated traceability and the presence of an extensive requirements specification and architecture design specification, the AC Report produced lacks some information and duplicate other.

APPENDIX D – SAFETY ASSURANCE ONTOLOGY OWL SPECIFICATION

Source code 5 – Safety Assurance Ontology OWL Specification

```

1 <?xml version="1.0"?>
2 <rdf:RDF xmlns="dare:arcade/sao"
    xml:base="dare:arcade/sao"
4    xmlns:sao="dare:arcade/sao#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
6    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xml="http://www.w3.org/XML/1998/namespace"
8    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
10 <owl:Ontology rdf:about="dare:arcade/sao">
    <owl:versionIRI rdf:resource="dare:0.1"/>
12    <rdfs:comment xml:lang="en">ARO (Assurance Rationale Ontology) is an
        operational ontology designed to support the handling of assurance
        information inside project management tools</rdfs:comment>
    <rdfs:label xml:lang="en">Assurance Rationale Ontology</rdfs:label>
14    <owl:versionInfo xml:lang="en">0.4</owl:versionInfo>
</owl:Ontology>
16 <!--
    //
18    // Object Properties
    //
20    -->
<!-- dare:arcade/sao#allocates -->
22 <owl:ObjectProperty rdf:about="dare:arcade/sao#allocates">
    <owl:inverseOf rdf:resource="dare:arcade/sao#isAllocatedIn"/>
24    <rdfs:domain rdf:resource="dare:arcade/sao#Component"/>
    <rdfs:range rdf:resource="dare:arcade/sao#Requirement"/>
26    <rdfs:comment xml:lang="en">It links a Component to Requirements that it
        allocates</rdfs:comment>
    <rdfs:label xml:lang="en">allocates</rdfs:label>
28 </owl:ObjectProperty>
<!-- dare:arcade/sao#causes -->
30 <owl:ObjectProperty rdf:about="dare:arcade/sao#causes">
    <owl:inverseOf rdf:resource="dare:arcade/sao#isCausedBy"/>
32    <rdfs:domain rdf:resource="dare:arcade/sao#Cause"/>
    <rdfs:range rdf:resource="dare:arcade/sao#Hazard"/>
34    <rdfs:comment xml:lang="en">It links any Object to the Hazards it causes.
        Any object can be a Cause.</rdfs:comment>
    <rdfs:label xml:lang="en">causes</rdfs:label>
36 </owl:ObjectProperty>
<!-- dare:arcade/sao#contextualizes -->
38 <owl:ObjectProperty rdf:about="dare:arcade/sao#contextualizes">
    <owl:inverseOf rdf:resource="dare:arcade/sao#isContextualizedBy"/>

```

```

40    <rdfs:domain rdf:resource="dare:arcade/sao#Context"/>
    <owl:propertyChainAxiom rdf:parseType="Collection">
42        <rdf:Description rdf:about="dare:arcade/sao#contextualizes"/>
        <rdf:Description rdf:about="dare:arcade/sao#isRefinedBy"/>
44    </owl:propertyChainAxiom>
    <rdfs:comment xml:lang="en">It links a Context to Objects it
        contextualizes.</rdfs:comment>
46    <rdfs:label xml:lang="en">contextualizes</rdfs:label>
</owl:ObjectProperty>
48 <owl:Axiom>
    <owl:annotatedSource rdf:resource="dare:arcade/sao#contextualizes"/>
50    <owl:annotatedProperty rdf:resource="http://www.w3.org/2002/07/owl#
        propertyChainAxiom"/>
    <owl:annotatedTarget rdf:parseType="Collection">
52        <rdf:Description rdf:about="dare:arcade/sao#contextualizes"/>
        <rdf:Description rdf:about="dare:arcade/sao#isRefinedBy"/>
54    </owl:annotatedTarget>
    <rdfs:comment xml:lang="en">If a Context is linked to a higher object,
        then also all the refinements of the Object will be</rdfs:comment>
56 </owl:Axiom>
<!-- dare:arcade/sao#documents -->
58 <owl:ObjectProperty rdf:about="dare:arcade/sao#documents">
    <owl:inverseOf rdf:resource="dare:arcade/sao#isDocumentedBy"/>
60    <rdfs:domain rdf:resource="dare:arcade/sao#Source"/>
    <rdfs:range rdf:resource="dare:arcade/sao#Object"/>
62    <rdfs:comment xml:lang="en">It links a Source to Objects that it
        documents.</rdfs:comment>
    <rdfs:label xml:lang="en">documents</rdfs:label>
64 </owl:ObjectProperty>
<!-- dare:arcade/sao#explains -->
66 <owl:ObjectProperty rdf:about="dare:arcade/sao#explains">
    <owl:inverseOf rdf:resource="dare:arcade/sao#isExplainedBy"/>
68    <rdfs:domain rdf:resource="dare:arcade/sao#Rationale"/>
    <rdfs:comment xml:lang="en">It links a Rationale to Objects it provide
        explanation.</rdfs:comment>
70    <rdfs:label xml:lang="en">explains</rdfs:label>
</owl:ObjectProperty>
72 <!-- dare:arcade/sao#implements -->
<owl:ObjectProperty rdf:about="dare:arcade/sao#implements">
74    <owl:inverseOf rdf:resource="dare:arcade/sao#isImplementedBy"/>
    <rdfs:domain rdf:resource="dare:arcade/sao#Component"/>
76    <rdfs:range rdf:resource="dare:arcade/sao#DesignDefinition"/>
    <rdfs:comment xml:lang="en">It links a Component to DesignDefinition that
        it implements.</rdfs:comment>
78    <rdfs:label xml:lang="en">implements</rdfs:label>
</owl:ObjectProperty>
80 <!-- dare:arcade/sao#isAllocatedIn -->

```

```

82   <owl:ObjectProperty rdf:about="dare:arcade/sao#isAllocatedIn">
      <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="dare:arcade/sao#refines"/>
84        <rdf:Description rdf:about="dare:arcade/sao#isAllocatedIn"/>
      </owl:propertyChainAxiom>
86      <rdfs:comment xml:lang="en">Inverse of allocates</rdfs:comment>
      <rdfs:label xml:lang="en">is allocated in</rdfs:label>
88    </owl:ObjectProperty>
    <!-- dare:arcade/sao#isCausedBy -->
90    <owl:ObjectProperty rdf:about="dare:arcade/sao#isCausedBy">
      <rdfs:comment xml:lang="en">Inverse of causes</rdfs:comment>
92      <rdfs:label xml:lang="en">is caused by</rdfs:label>
    </owl:ObjectProperty>
94    <!-- dare:arcade/sao#isContextualizedBy -->
    <owl:ObjectProperty rdf:about="dare:arcade/sao#isContextualizedBy">
96      <rdfs:comment xml:lang="en">Inverse of contextualizes</rdfs:comment>
      <rdfs:label xml:lang="en">is contextualized by</rdfs:label>
98    </owl:ObjectProperty>
    <!-- dare:arcade/sao#isDocumentedBy -->
100    <owl:ObjectProperty rdf:about="dare:arcade/sao#isDocumentedBy">
      <rdfs:domain rdf:resource="dare:arcade/sao#Object"/>
102      <rdfs:range rdf:resource="dare:arcade/sao#Source"/>
      <rdfs:comment xml:lang="en">Inverse of documents</rdfs:comment>
104      <rdfs:label xml:lang="en">is documented by</rdfs:label>
    </owl:ObjectProperty>
106    <!-- dare:arcade/sao#isExplainedBy -->
    <owl:ObjectProperty rdf:about="dare:arcade/sao#isExplainedBy">
108      <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="dare:arcade/sao#refines"/>
110        <rdf:Description rdf:about="dare:arcade/sao#isExplainedBy"/>
      </owl:propertyChainAxiom>
112      <rdfs:comment xml:lang="en">Inverse of explains</rdfs:comment>
      <rdfs:label xml:lang="en">is explained by</rdfs:label>
114    </owl:ObjectProperty>
    <!-- dare:arcade/sao#isImplementedBy -->
116    <owl:ObjectProperty rdf:about="dare:arcade/sao#isImplementedBy">
      <rdfs:comment xml:lang="en">Inverse of implements</rdfs:comment>
118      <rdfs:label xml:lang="en">is implemented by</rdfs:label>
    </owl:ObjectProperty>
120    <!-- dare:arcade/sao#isMitigatedBy -->
    <owl:ObjectProperty rdf:about="dare:arcade/sao#isMitigatedBy">
122      <owl:inverseOf rdf:resource="dare:arcade/sao#mitigates"/>
      <rdfs:comment xml:lang="en">Inverse of mitigates</rdfs:comment>
124      <rdfs:label xml:lang="en">is mitigated by</rdfs:label>
    </owl:ObjectProperty>
126    <!-- dare:arcade/sao#isRealizedBy -->
    <owl:ObjectProperty rdf:about="dare:arcade/sao#isRealizedBy">

```

```

128     <owl:inverseOf rdf:resource="dare:arcade/sao#realizes"/>
    <rdfs:comment xml:lang="en">Inverse of realizes</rdfs:comment>
130     <rdfs:label xml:lang="en">is realized by</rdfs:label>
  </owl:ObjectProperty>
132  <!-- dare:arcade/sao#isRefinedBy -->
  <owl:ObjectProperty rdf:about="dare:arcade/sao#isRefinedBy">
134     <owl:inverseOf rdf:resource="dare:arcade/sao#refines"/>
    <rdfs:comment xml:lang="en">Inverse of refines</rdfs:comment>
136     <rdfs:label xml:lang="en">is refined by</rdfs:label>
  </owl:ObjectProperty>
138  <!-- dare:arcade/sao#mitigates -->
  <owl:ObjectProperty rdf:about="dare:arcade/sao#mitigates">
140     <rdfs:domain rdf:resource="dare:arcade/sao#SafetyRequirement"/>
    <rdfs:range>
142     <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
144         <rdf:Description rdf:about="dare:arcade/sao#Cause"/>
        <rdf:Description rdf:about="dare:arcade/sao#Hazard"/>
146     </owl:unionOf>
    </owl:Class>
148  </rdfs:range>
  <owl:propertyChainAxiom rdf:parseType="Collection">
150     <rdf:Description rdf:about="dare:arcade/sao#mitigates"/>
    <rdf:Description rdf:about="dare:arcade/sao#causes"/>
152  </owl:propertyChainAxiom>
  <owl:propertyChainAxiom rdf:parseType="Collection">
154     <rdf:Description rdf:about="dare:arcade/sao#refines"/>
    <rdf:Description rdf:about="dare:arcade/sao#mitigates"/>
156  </owl:propertyChainAxiom>
    <rdfs:comment xml:lang="en">It links a SafetyRequirement to the Hazards
      that it addresses</rdfs:comment>
158     <rdfs:label xml:lang="en">mitigates</rdfs:label>
  </owl:ObjectProperty>
160  <!-- dare:arcade/sao#realizes -->
  <owl:ObjectProperty rdf:about="dare:arcade/sao#realizes">
162     <rdfs:domain rdf:resource="dare:arcade/sao#DesignDefinition"/>
    <rdfs:range rdf:resource="dare:arcade/sao#Requirement"/>
164     <rdfs:comment xml:lang="en">It links a DesignDefinition to the
      Requirements that are realized by it</rdfs:comment>
    <rdfs:label xml:lang="en">realizes</rdfs:label>
166  </owl:ObjectProperty>
  <!-- dare:arcade/sao#refines -->
168  <owl:ObjectProperty rdf:about="dare:arcade/sao#refines">
    <rdfs:comment xml:lang="en">It represents a refinement of Requirement or
      Hazard</rdfs:comment>
170     <rdfs:label xml:lang="en">refines</rdfs:label>
  </owl:ObjectProperty>

```

```

172 <!--
    //
174    // Data properties
    //
176    -->
    <!-- dare:arcade/sao#creationdate -->
178    <owl:DatatypeProperty rdf:about="dare:arcade/sao#creationdate">
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"
            />
180        <rdfs:domain>
            <owl:Class>
182                <owl:unionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="dare:arcade/sao#Object"/>
184                    <rdf:Description rdf:about="dare:arcade/sao#Source"/>
                </owl:unionOf>
186            </owl:Class>
        </rdfs:domain>
188        <rdfs:comment xml:lang="en">Creation date of the source or object</
            rdfs:comment>
        <rdfs:label xml:lang="en">creation date</rdfs:label>
190    </owl:DatatypeProperty>
    <!-- dare:arcade/sao#externaluri -->
192    <owl:DatatypeProperty rdf:about="dare:arcade/sao#externaluri">
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"
            />
194        <rdfs:domain>
            <owl:Class>
196                <owl:unionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="dare:arcade/sao#Object"/>
198                    <rdf:Description rdf:about="dare:arcade/sao#Source"/>
                </owl:unionOf>
200            </owl:Class>
        </rdfs:domain>
202        <rdfs:comment xml:lang="en">External link for the object or source</
            rdfs:comment>
        <rdfs:label xml:lang="en">external URI</rdfs:label>
204    </owl:DatatypeProperty>
    <!-- dare:arcade/sao#lastupdate -->
206    <owl:DatatypeProperty rdf:about="dare:arcade/sao#lastupdate">
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"
            />
208        <rdfs:domain>
            <owl:Class>
210                <owl:unionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="dare:arcade/sao#Object"/>
212                    <rdf:Description rdf:about="dare:arcade/sao#Source"/>
                </owl:unionOf>

```



```

214         </owl:Class>
        </rdfs:domain>
216     <rdfs:comment xml:lang="en">Last update of the object or source</
        rdfs:comment>
        <rdfs:label xml:lang="en">last update</rdfs:label>
218 </owl:DatatypeProperty>
<!-- dare:arcade/sao#status -->
220 <owl:DatatypeProperty rdf:about="dare:arcade/sao#status">
    <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"
        />
222    <rdfs:domain>
        <owl:Class>
224            <owl:unionOf rdf:parseType="Collection">
                <rdfs:Description rdf:about="dare:arcade/sao#Object"/>
226                <rdfs:Description rdf:about="dare:arcade/sao#Source"/>
            </owl:unionOf>
228        </owl:Class>
    </rdfs:domain>
230    <rdfs:comment xml:lang="en">Status of the object or source</rdfs:comment>
    <rdfs:label xml:lang="en">status</rdfs:label>
232 </owl:DatatypeProperty>
<!--
234 //
    // Classes
236 //
    -->
238 <!-- dare:arcade/sao#Assumption -->
<owl:Class rdf:about="dare:arcade/sao#Assumption">
240    <rdfs:subClassOf rdf:resource="dare:arcade/sao#Rationale"/>
    <rdfs:comment xml:lang="en">A statement that reflects a common situation
        in the domain so ingrained into stakeholders general beliefs, that it
        is unquestioned</rdfs:comment>
242    <rdfs:label xml:lang="en">Assumption</rdfs:label>
</owl:Class>
244 <!-- dare:arcade/sao#Cause -->
<owl:Class rdf:about="dare:arcade/sao#Cause">
246    <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
    <rdfs:comment xml:lang="en">An environmental, system condition,
        requirement or design definition that causes a hazard.</rdfs:comment>
248    <rdfs:label xml:lang="en">Cause</rdfs:label>
</owl:Class>
250 <!-- dare:arcade/sao#Component -->
<owl:Class rdf:about="dare:arcade/sao#Component">
252    <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
    <rdfs:comment xml:lang="en">A unit of architecture, design or code of the
        system</rdfs:comment>
254    <rdfs:label xml:lang="en">Component</rdfs:label>

```

```

256 </owl:Class>
<!-- dare:arcade/sao#Context -->
<owl:Class rdf:about="dare:arcade/sao#Context">
258   <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
   <rdfs:comment xml:lang="en">An environmental or operational condition
       that may occur, and it is relevant to safety</rdfs:comment>
260   <rdfs:label xml:lang="en">Context</rdfs:label>
</owl:Class>
262 <!-- dare:arcade/sao#DesignDefinition -->
<owl:Class rdf:about="dare:arcade/sao#DesignDefinition">
264   <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
   <rdfs:comment xml:lang="en">A decision that represents a choice in the
       design of the system</rdfs:comment>
266   <rdfs:label xml:lang="en">Design Definition</rdfs:label>
</owl:Class>
268 <!-- dare:arcade/sao#Hazard -->
<owl:Class rdf:about="dare:arcade/sao#Hazard">
270   <owl:equivalentClass>
     <owl:Class>
272       <owl:unionOf rdf:parseType="Collection">
         <owl:Restriction>
274           <owl:onProperty rdf:resource="dare:arcade/sao#isRefinedBy"
               />
           <owl:someValuesFrom rdf:resource="dare:arcade/sao#Hazard"
               />
276         </owl:Restriction>
         <owl:Restriction>
278           <owl:onProperty rdf:resource="dare:arcade/sao#refines"/>
           <owl:someValuesFrom rdf:resource="dare:arcade/sao#Hazard"
               />
280         </owl:Restriction>
       </owl:unionOf>
     </owl:Class>
282   </owl:equivalentClass>
   <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
   <rdfs:comment xml:lang="en">A single hazard or category of hazards</
       rdfs:comment>
286   <rdfs:label xml:lang="en">Hazard</rdfs:label>
</owl:Class>
288 <!-- dare:arcade/sao#Justification -->
<owl:Class rdf:about="dare:arcade/sao#Justification">
290   <rdfs:subClassOf rdf:resource="dare:arcade/sao#Rationale"/>
   <rdfs:comment xml:lang="en">A statement that provides an explanation for
       a project decision</rdfs:comment>
292   <rdfs:label xml:lang="en">Justification</rdfs:label>
</owl:Class>
294 <!-- dare:arcade/sao#Object -->

```

```

296   <owl:Class rdf:about="dare:arcade/sao#Object">
      <rdfs:comment>Input or output of the development process, in any level of
        granularity, that is managed in the project management tool.</
        rdfs:comment>
      <rdfs:label xml:lang="en">Object</rdfs:label>
298 </owl:Class>
    <!-- dare:arcade/sao#Rationale -->
300 <owl:Class rdf:about="dare:arcade/sao#Rationale">
      <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
302 <rdfs:comment xml:lang="en">An explanation that provides additional
      information for an Object</rdfs:comment>
      <rdfs:label xml:lang="en">Rationale</rdfs:label>
304 </owl:Class>
    <!-- dare:arcade/sao#Requirement -->
306 <owl:Class rdf:about="dare:arcade/sao#Requirement">
      <owl:equivalentClass>
308 <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
310 <owl:Restriction>
          <owl:onProperty rdf:resource="dare:arcade/sao#isRefinedBy
            "/>
312 <owl:someValuesFrom rdf:resource="dare:arcade/sao#
            Requirement"/>
          </owl:Restriction>
314 <owl:Restriction>
          <owl:onProperty rdf:resource="dare:arcade/sao#refines"/>
316 <owl:someValuesFrom rdf:resource="dare:arcade/sao#
            Requirement"/>
          </owl:Restriction>
318 </owl:unionOf>
        </owl:Class>
320 </owl:equivalentClass>
      <rdfs:subClassOf rdf:resource="dare:arcade/sao#Object"/>
322 <rdfs:comment xml:lang="en">A system requirement</rdfs:comment>
      <rdfs:label xml:lang="en">Requirement</rdfs:label>
324 </owl:Class>
    <!-- dare:arcade/sao#SafetyRequirement -->
326 <owl:Class rdf:about="dare:arcade/sao#SafetyRequirement">
      <owl:equivalentClass>
328 <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
330 <rdf:Description rdf:about="dare:arcade/sao#Requirement"/>
        <owl:Restriction>
332 <owl:onProperty rdf:resource="dare:arcade/sao#mitigates"/
          >
          <owl:someValuesFrom>
334 <owl:Class>

```

```

336         <owl:unionOf rdf:parseType="Collection">
            <rdf:Description rdf:about="dare:arcade/sao#
                Cause"/>
            <rdf:Description rdf:about="dare:arcade/sao#
                Hazard"/>
338         </owl:unionOf>
        </owl:Class>
340     </owl:someValuesFrom>
    </owl:Restriction>
342 </owl:intersectionOf>
    </owl:Class>
344 </owl:equivalentClass>
    <rdfs:subClassOf rdf:resource="dare:arcade/sao#Requirement"/>
346 <rdfs:comment xml:lang="en">A requirement related to system safety. It
        represents actions and constraints should or should not be performed
        to maintain the system in a safe state</rdfs:comment>
    <rdfs:label xml:lang="en">Safety Requirement</rdfs:label>
348 </owl:Class>
    <!-- dare:arcade/sao#Source -->
350 <owl:Class rdf:about="dare:arcade/sao#Source">
    <rdfs:comment xml:lang="en">An external artefact to the project
        management tool</rdfs:comment>
352 <rdfs:label xml:lang="en">Source</rdfs:label>
    </owl:Class>
354 <!-- dare:arcade/sao#Strategy -->
    <owl:Class rdf:about="dare:arcade/sao#Strategy">
356 <rdfs:subClassOf rdf:resource="dare:arcade/sao#Rationale"/>
    <rdfs:comment xml:lang="en">A statement behind a project's decision. It
        can be a design decision, a requirements decision, or a mitigation
        strategy selected for a hazard or a cause.</rdfs:comment>
358 <rdfs:label xml:lang="en">Strategy</rdfs:label>
    </owl:Class>
360 <!--
    //
362 // General axioms
    //
364 -->
    <rdf:Description>
366 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#AllDisjointClasses
        "/>
        <owl:members rdf:parseType="Collection">
368 <rdf:Description rdf:about="dare:arcade/sao#Assumption"/>
            <rdf:Description rdf:about="dare:arcade/sao#Justification"/>
370 <rdf:Description rdf:about="dare:arcade/sao#Strategy"/>
        </owl:members>
372 </rdf:Description>
    <rdf:Description>

```

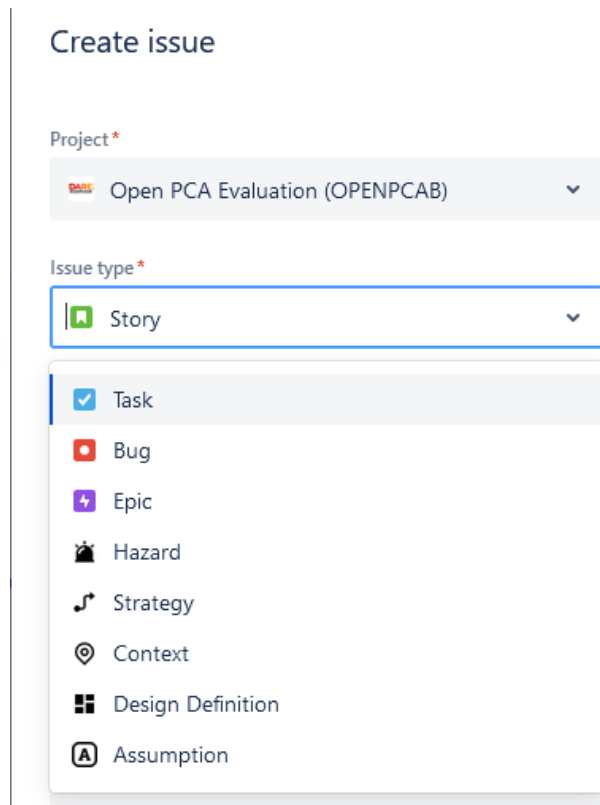
```
374     <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#AllDisjointClasses
        "/>
        <owl:members rdf:parseType="Collection">
376         <rdf:Description rdf:about="dare:arcade/sao#Component"/>
        <rdf:Description rdf:about="dare:arcade/sao#Context"/>
378         <rdf:Description rdf:about="dare:arcade/sao#DesignDefinition"/>
        <rdf:Description rdf:about="dare:arcade/sao#Hazard"/>
380         <rdf:Description rdf:about="dare:arcade/sao#Rationale"/>
        <rdf:Description rdf:about="dare:arcade/sao#Requirement"/>
382     </owl:members>
    </rdf:Description>
384 </rdf:RDF>
<!-- Generated by the OWL API (version 4.5.9.2019-02-01T07:24:44Z) https://github
.com/owlcs/owlapi -->
```

Source: The author (2023)

APPENDIX E – CUSTOMIZED JIRA INSTANCE

This appendix shows some screenshots from the customized Jira instance available at <https://arcade-dare.atlassian.net>.

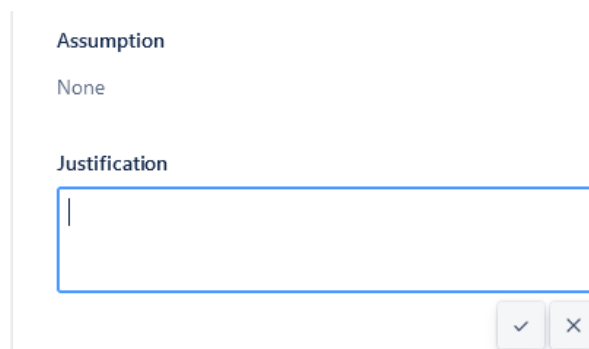
Figure 47 – Custom issue types



The screenshot shows the 'Create issue' form in Jira. The 'Project' dropdown is set to 'Open PCA Evaluation (OPENPCAB)'. The 'Issue type' dropdown is open, showing a list of custom issue types: Task (checked), Bug, Epic, Hazard, Strategy, Context, Design Definition, and Assumption. Each issue type has a corresponding icon.

Source: The author

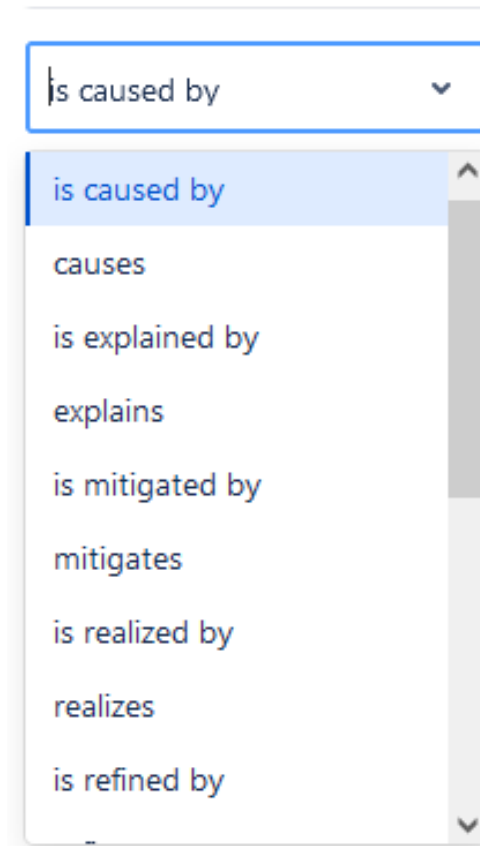
Figure 48 – Custom fields



The screenshot shows the 'Assumption' custom field in Jira. The field is currently set to 'None'. Below the field name is a 'Justification' text area with a blue border and a vertical cursor. At the bottom right of the text area are two small buttons: a checkmark and an 'X'.

Source: The author

Figure 49 – Custom issue link types



Source: The author

Figure 50 – View of all issues

Issues


Share Export issues Go to all issues LIST VIEW DETAIL VIEW

Search issues Project: Open PCA Evaluation Type Status Assignee More + Reset Save filter BASIC JQL




Type	Key	Summary	Labels	Components	Status	Updated	Create
	OPENPCAB-36	Patient-Requested Bolus			TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-37	Maximum VTBI	safety	Pump	TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-38	Minimum time between patient-request bolus	safety	Patient Bolus Checker	TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-39	Delivery of patient-requested bolus		Control Panel	TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-40	Basal infusion flow tolerance	safety	Pump	TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-41	Measure drug flow and alarm	safety		TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-42	Basal infusion flow range	safety	Pump	TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-43	Uneven delivery hazard			TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-44	Too Many User Boluses Hazard			TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-45	ASTM International F2761-09			TO DO	May 9, 2023	May 9, 2023
	OPENPCAB-46	IEC 60601-1-8			TO DO	May 9, 2023	May 9, 2023

Source: The author

Figure 51 – A safety requirement recorded as a Story issue with label “safety”

 OPENPCAB-38


Minimum time between patient-request bolus

 Create subtask  Link issue  


Description
Patient-requested bolus shall not be delivered more often than a prescribed minimum time between patient-requested bolus.

Assumption
None


Justification
Enforcing minimum time between boluses prevents too many user boluses, avoiding overinfusion.

Linked issues 

mitigates


 OPENPCAB-44 Too Many User Boluses Hazard   TO DO 

refines





 OPENPCAB-36 Patient-Requested Bolus   TO DO 

Source: The author

Figure 52 – A hazard recorded as a Hazard issue

 OPENPCAB-43


Uneven delivery hazard

 Create subtask  Link issue  





Description
The programmed bolus dose is delivered unevenly over its specified duration. Potential causes: Algorithmic errors, The pump motor does not operate as intended.

Assumption
Alarming upon uneven delivery stops flow and hails clinician.





Justification
None

Linked issues 






is caused by

 OPENPCAB-32 Clinician starts/stops pump   TO DO 

is explained by

 OPENPCAB-41 Measure drug flow and alarm   TO DO 

is mitigated by

 OPENPCAB-58 Basal over-infusion alarm   TO DO  

Source: The author

APPENDIX F – INFORMED CONSENT AND PERSONAL PROFILE QUESTIONNAIRES


ARCADE

FRAMEWORK FOR SAFETY ASSURANCE
INFORMATION MANAGEMENT

ARCADE-FRAMEWORK.GITHUB.IO

Informed Consent & Personal Profile



Please, fill your email address to receive the evaluation procedures.

cca3@cin.ufpe.br [Alternar conta](#) 

* Indica uma pergunta obrigatória

E-mail *

Seu e-mail



Dear participant,

Through some short questionnaires and tasks, we invite you to participate in the **evaluation of ARCADE (AssuRance CAsE DEvelopment) framework**.

This evaluation aims to collect **expert opinions** on the features of the ARCADE framework.

We want to emphasize that:

1. Your participation is entirely voluntary.
2. We are interested in investigating only your professional opinion regarding the area of study and the object of this research.
3. At any time, you can withdraw from participating and withdraw your consent.
4. We will disclose the data collected in this form in a way that the identification of the respondents is not possible.

Researchers involved:

- Camilo Almendra (Doctoral Student of the Postgraduate Program in Computer Science - CIn/UFPE) - cca3@cin.ufpe.br
- Prof. Dr. Carla Silva (CIn/UFPE) - ctlls@cin.ufpe.br

I agree to the terms: *

☐ Yes

Próxima

Limpar formulário

Nunca envie senhas pelo Formulários Google.

Este formulário foi criado em Centro de Informatica - UFPE. [Denunciar abuso](#)

Google Formulários




ARCADE

FRAMEWORK FOR SAFETY ASSURANCE
INFORMATION MANAGEMENT

ARCADE-FRAMEWORK.GITHUB.IO

Informed Consent & Personal Profile

cca3@cin.ufpe.br [Alternar conta](#) 

* Indica uma pergunta obrigatória

Personal profile

We thank you for sharing your expertise. These questions are focused on gathering your expertise on the safety and system development.

How long have you been working on development and/or research of safety-critical systems? *

☐ Less than 1 year

☐ 1 to 2 years


☐ 3 to 5 years


☐ 6 to 10 years

☐ More than 10 years

What is your current main role in the organization *

Sua resposta





What is your level of familiarity (knowledge or practice) of Safety Assurance Cases *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar

How many projects (concluded or in-progress) that involved Assurance Cases for safety-critical systems have you participated in? *

- ☐ None
- ☐ 1 to 4 projects
- ☐ 5 to 10 projects
- ☐ More than 10 projects

What is your level of familiarity (knowledge or practice) of the GSN (Goal Structuring Notation)? *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar



What is your level of familiarity (knowledge or practice) with ontology specification? *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar

What is your level of familiarity (knowledge or practice) with Description Logic reasoning or use of semantic reasoners? *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar

What is your level of familiarity (knowledge or practice) in **using** Product/Application Lifecycle Management and/or Issue Tracking tools? *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar



What is your level of familiarity (knowledge) in **configuring/administering** Product/Application Lifecycle Management and/or Issue Tracking tools? *

- ☐ Not at all familiar
- ☐ Slightly familiar
- ☐ Somewhat familiar
- ☐ Moderately familiar
- ☐ Extremely familiar

Uma cópia das suas respostas será enviada para o endereço de e-mail fornecido

Voltar

Enviar

Limpar formulário

Nunca envie senhas pelo Formulários Google.



reCAPTCHA
[Privacidade](#)[Termos](#)

Este formulário foi criado em Centro de Informatica - UFPE. [Denunciar abuso](#)

Google Formulários



APPENDIX G – EVALUATION PROCEDURES



UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ



ARCADE Framework Evaluation

Participant: NAME/EMAIL

Note: all response data will be handled without disclosing the participant identification.

Part 1 - Safety Assurance Information Model (SAIM)

(time estimate = 55 minutes, including videos)

1.1 Short training

Safety Assurance Information Model (9 min) - Portuguese audio, English subtitles

- <https://youtu.be/UiPa9IdaRU8>

1.2 Short training

Issue Tracking Systems (12 min) - Portuguese audio, English subtitles

- <https://youtu.be/LGmEh3bsH-w>

1.3 Short training

Safety Assurance Cases (9 min) - Portuguese audio, English subtitles

- <https://youtu.be/Urp-SzrTJSw>

1.4 Evaluation tasks using SAIM

(if any step could not be performed, contact us)

1.4.0 Instructions

To perform tasks below, you need to access an Atlassian Jira instance that is configured with a public dataset, and it is adapted to the Safety Assurance Information Model (SAIM) we discussed in the short training.

- Link: <https://arcade-dare.atlassian.net/browse/OPENPCAB>
- Credentials:
 - email =
 - password =
- Information on the dataset: <https://arcade-framework.github.io/openpca/>

1.4.1 Identify and list below which are the Hazards that are not mitigated by any Safety Requirement of the project. You can provide the summary or only the issue ids.

--

1.4.2 How difficult was it to retrieve the information requested in step 1.4.1?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

1.4.3 Identify and list below which are the Safety Requirements that are not explained by any Rationale (Assumption, Justification or Strategy) of the project. You can provide the summary or only the issue ids.

--

1.4.4 How difficult was it to retrieve the information requested in step 1.4.3?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

1.4.5 Regarding the issue <https://arcade-dare.atlassian.net/browse/OPENPCAB-40>, which are the project items that contextualize and provide assurance evidence to it?

--

1.4.6 How difficult was it to retrieve the information requested in step 1.4.5?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

1.4.7 Regarding the issue <https://arcade-dare.atlassian.net/browse/OPENPCAB-70>, which are the project items that contextualize and provide assurance evidence to it?

--

1.4.8 How difficult was it to retrieve the information requested in step 1.4.7?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

1.5 Opinion questionnaire (Part 1)

1.5.1 To what extent do you support the approach to manage safety assurance information together with regular project information inside project management tools?

Strongly oppose	Somewhat oppose	Neutral	Somewhat favor	Strongly Favor	Undecided
Comment if any:					

1.5.2 For each concept of the Safety Assurance Information Model, please indicate if it applies to the projects you participated in the last 5 years. Mark with an "X".

Concept	Every Project	Most of the Projects	Some Projects	Few Projects	Never applied
Requirement					
Safety Requirement					
Hazard					
Cause (of Hazard)					
Design Definition					
Component					
Context					
Assumption					
Justification					
Strategy					
Object					
Source					

(Please indicate an answer for each line, leave it blank for "don't know")

1.5.3 From your experience, please indicate any important concept which is not covered in the SAIM model.

1.5.4 Suppose the following scenario:

- You have to review safety assurance information of a project
- You have to produce an Assurance Case for the project
- Traceability data is available in Atlassian Jira adapted to the SAIM model.

A. In your opinion, which are the **positive aspects** of using the Jira adapted to the SAIM to accomplish the tasks?

B. In your opinion, which are the **negative aspects** of using the Jira adapted to the SAIM to accomplish the tasks?

1.5.5 Do you have any additional comments regarding the use of SAIM?

Part 2 - ARCADE Framework

(time estimate = 35 minutes, including video)

TO AVOID BIAS, WE ASK TO ONLY PROCEED TO PART 2
AFTER COMPLETING THE TASKS OF PART 1

2.1 Short training

ARCADE Framework (22 min) - Portuguese audio, English subtitles

- <https://youtu.be/MXC9xnU-QIs>

2.2 Evaluation tasks using ARCADE

(if any step could not be performed, contact us)

2.2.0 Instructions

To perform the tasks below, you need to access the ARCADE Framework we discussed in the short training.

- Link: <https://arcade.cin.ufpe.br>
- Credentials
 - login =
 - password =

Optionally, you may access the Atlassian Jira to explore the DRONE project

- Link: <https://arcade-dare.atlassian.net>
- Credentials (*different from the step 1.4.0*)
 - email =
 - password =
- Information on the dataset: <https://arcade-framework.github.io/drone>

2.2.1 Load the DRONE project in the ARCADE.

- Go to ARCADE tool
- Go to Home page
- Go to section “Project data items and links”, sub-section “Load project data”
- Click on “Load DRONE project from Jira”
- Check after few seconds if a message “Project items successfully imported from “PROJECT DRONE” is show
- Go to section “View the project items..” and check if 155 individuals were loaded

2.2.2 Identify the Hazards that are not mitigated by any Safety Requirement of the project.

- There is a competency question that matches this task
- List the project items that answer the question

2.2.3 Identify the Safety Requirements that are not explained by any Rationale (Assumption, Justification or Strategy) of the project.

- There is a competency question that matches this task
- List the project items that answer the question

2.2.4 How difficult was it to retrieve and explore the information requested in steps 2.2.2 and 2.2.3?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

2.2.5 Regarding the project item DRONE-33 <https://arcade.cin.ufpe.br/#DRONE-33>, which are the project items that contextualize and provide assurance evidence to it?

2.2.6 Regarding the project item DRONE-136 <https://arcade.cin.ufpe.br/#DRONE-136>, which are the project items that contextualize and provide assurance evidence to it?

2.2.7 How difficult was it to retrieve and explore the information requested in steps 2.2.5 and 2.2.6?

Very difficult	Difficult	Neutral	Easy	Very easy
Comment if any:				

2.2.8 Explore the assurance case fragments

- Go to “Generate assurance case arguments”
- Go to “Functional Decomposition pattern” and click to expand (it opens new tab)
 - The fragment image is in SVG format and all text is searchable
 - Find the GSN element that contains the item “DRONE-33”
 - Explore the GSN elements adjacent to it
- Go to “Hazard Avoidance pattern” and click to expand
 - Find the GSN element that contains the item “DRONE-33”
 - Explore the GSN elements adjacent to it

2.2.9 How difficult was it to retrieve and explore the information requested in step 2.2.8?

Very difficult	Difficult	Neutral	Easy	Very easy

Comment if any:				

2.3 Opinion questionnaire (Part 2) – almost there, quick final questions

2.3.1 To what extent do you find relevant the ARCADE feature that organizes all project items by their types/classifications (the “Basic” competency questions)?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.2 To what extent do you find relevant the ARCADE feature that reveals gaps in traceability (the “Traceability Gaps” competency questions)?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.3 To what extent do you find relevant the ARCADE feature that checks consistency and generates explanations in case of inconsistent project data?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.4 To what extent do you find relevant the ARCADE feature that generates assurance cases fragment in multiple perspectives using the same project data?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.5 To what extent do you believe the adoption of ARCADE Framework could foster the cross-collaboration between safety and software teams towards shared management of safety assurance information?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.6 To what extent do you believe the adoption of ARCADE Framework could reduce the effort to review safety traceability information?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.7 To what extent do you believe the adoption of ARCADE Framework could reduce effort to develop safety assurance cases?

To A Great Extent	Somewhat	Very Little	Not At All	Undecided
Comment if any:				

2.3.8 Do you have any additional comments on ARCADE Framework?

--

Final Part - Submission Instructions

Submission

THANK YOU! We appreciate the time you put on in this evaluation. To submit your answers, you just need to notify us, and leave the document as is. You may save a copy for you.

Referral

If you wish to refer other researchers and practitioners to participate in this evaluation, please leave their contacts below:

--