Establishing Confidence in

Safety Assessment Evidence

Linling Sun

Submitted for the degree of Doctor of Philosophy

University of York

Department of Computer Science

July 2012

To my parents,

my son,

and my husband,

for their love, support and understanding.

Abstract

With the increased complexity and higher safety commitment of modern safety–critical systems, safety assessment models of these systems are increasingly complicated and obscure. In practice, however, there is insufficient guidance on how to improve the understanding and evaluation of these models, while they are often used as important items of evidence in safety cases. This significantly threatens the confidence we can have in the soundness of safety cases.

In this thesis, a coherent, structured approach to establishing confidence in safety assessment evidence is developed. Firstly, a means for the structured documentation of the core data elements of safety assessment models is defined, to support the development of both primary safety arguments and confidence arguments. Secondly, a model of evidence is developed to support the interfacing of safety assessment evidence with safety arguments. Thirdly, a structured cross-model inconsistency analysis method is proposed as a means of scrutinizing potentially inadequate models. Finally, an expanded argument construction process is established to add rigour to safety case development, and a number of argument patterns are designed to guide and inspire structured justification of the adequacy of safety assessment models as evidence for safety critical systems.

The evaluation of the approach is carried out primarily through examples and cases studies. It is demonstrated that the approach is feasible and the confidence issue in safety assessment evidence is addressed more explicitly and more rigorously by using the approach.

Contents

Abstract 3

Contents 4

List of Figures 11

List of Tables 14

Acknowledgements 15

Author's Declaration 16

1 Introduction 19

1.1 The Crater Model of Space Shuttle Columbia 19

1.2 The Loss of Nimrod XV230 21

1.3 Lessons Learnt 23

1.4 Safety Assessment Models 24

1.5 Existing Practice 26

1.6 Research Problem 27

1.7 Thesis Proposition 28

1.8 Thesis Structure 28

2 Literature Survey 31

2.1 Introduction 31

2.2 Models in General 32

2.2.1 Definition 32

2.2.2 Model Building 33

2.3 Model Validation 34

2.3.1 Viewpoints on Validation 34

2.3.2 Means of Validation 35

2.4 Safety Assessment Models 37

2.4.1 Traditional Techniques 37

2.4.2 Safety Analysis Based on Formal Languages 42

2.5 Safety Assessment Processes 44

2.5.1 SAE ARP 4754A Safety Process 44

2.5.2 Other Safety Processes 46

2.5.3 Process-Related Issues 46

2.6 Evaluation of Safety Assessment Models 48

2.6.1 Comparison of Safety Assessment Techniques 48

2.6.2 Existing Evaluation of Safety Assessment Models 49

2.7 Safety and Assurance Cases 52

2.7.1 Argumentation 53

2.7.2 Safety Case Overview 54

2.7.3 Structured Argument 55

2.7.4 Safety Evidence 59

2.7.5 Confidence Associated with Safety Cases 62

2.7.6 Models and Safety Cases 64

2.8 Summary 65

3 Safety Assessment Meta-Modelling 66

3.1 Introduction 66

3.2 Meta-Modelling 67

3.3 Safety Assessment Modelling 70

3.3.1 Domain Context of Safety Assessment Modelling 70

3.3.2 Contributing Factors to 'Wrong' Models 74

3.4 Safety Assessment Core Data Meta-Model 75

3.4.1 Core Data Meta-Model (CoreDMM) Overview 76

3.4.2 Representation of Typical Safety Assessment Models 80

3.4.3 Relations with Other Safety Assessment Meta-Models 85

3.4.4 Relations with System Assurance Meta-Models 87

3.5 Application of CoreDMM 89

3.6 Summary 90

4 A Model of Argument-Evidence Interface 91

4.1 Introduction 91

4.2 The Concept of Evidence 92

4.2.1 Evidence in Other Domains 93

4.2.2 Evidence in Safety Domain 95

4.2.3 Common Basis 96

4.3 Classification of Evidence in Safety Cases 97

4.4 Relationship between Evidence and Argument 99

4.4.1 Toulmin's Argument Structure 100

4.4.2 A Simplified View of Evidence 101

4.4.3 Evidence Representation in Notations 103

4.5 A Model of Evidence (EviM) 105

4.5.1 Three Perspectives 105

4.5.2 EviM Overview 107

4.5.3 Relations with ARM 110

4.5.4 Evidence Assertion 112

4.5.5 Evidence Result Assertion 114

4.5.6 Evidence Descriptive Assertion 118

4.5.7 Eliciting Evidence Assertions 120

4.5.8 Trustworthiness and Appropriateness 122

4.6 Relationship between Evidence and Confidence 125

4.6.1 Confidence in Safety Cases 125

4.6.2 Importance of Evidence towards the Establishment of Confidence 127

4.7 Summary 128

5 Managing Safety Model Inconsistency 129

5.1 Introduction 129

5.2 The Territory of Consistency 130

5.2.1 Range of Meaning 130

5.2.2 Typology of Safety Model Consistency 131

5.2.3 Factual Consistency 133

5.2.4 Intra-Model Consistency 134

5.2.5 Cross-Model Consistency 136

5.3 Consistency Relationships 137

5.3.1 Common Features 137

5.3.2 Types of Consistency Relationships 139

5.3.3 Consistency Related to Conditions 140
5.3.4 Consistency Related to MCSs 141
5.4 Cross-Model Inconsistency Analysis 142
5.4.1 Method Overview 143
5.4.2 Exemplar Models for a Running Example 145
5.4.3 Phase 1: Selecting Models 149
5.4.4 Phase 2: Elaborating Consistency Relationships 151
5.4.5 Phase 3: Implementing Model Comparison 154
5.4.6 Phase 4: Interpreting Comparison Results 161
5.4.7 Phase 5: Evaluating the Inconsistency Analysis 163
5.4.8 Phase 6: Citing and Justifying Analysis Results 163
5.4.9 Summary of the Inconsistency Analysis Method 164
5.5 Practicality and Capability Analysis 164
5.6 Summary 166
6 Using and Justifying Safety Assessment Evidence 167
6.1 Introduction 167
6.2 Role of Evidence in Safety Cases 168
6.3 Consideration of Counter Evidence 170
6.4 Presentation of Counter Evidence 173
6.5 Safety Case Processes 174
6.5.1 Existing Processes 174
6.5.2 Six-Step GSN Method 175
6.6 Expanded Structured Argument Construction 177
6.6.1 Overall Expanded Process 177
6.6.2 Step E1: Justify Basis of Goals 180
6.6.3 Step E2: Identify Alternative Basis of Goals Stated 181
6.6.4 Step E3: Justify Unsuitability of Alternative Basis of Goals 181
6.6.5 Step E4: Justify Basis of Strategy 182
6.6.6 Step E5: Identify Alternative Basis of Strategy Stated 183
6.6.7 Step E6: Justify Unsuitability of Alternative Basis of Strategy 184
6.6.8 Step E7: Justify Adequacy of Solutions 184

6.6.9 Step E8: Identify Counter Evidence 185

6.6.10 Step E9: Justify Discharge of Counter Evidence 186

6.6.11 Rethinking of Strategy 187

6.6.12 Practicality Analysis 189

6.6.13 Relations with Confidence Argument 190

6.6.14 Relationship to Assurance Claim Points 193

6.6.15 Confirmation Bias in Arguments 195

6.7 Argument Pattern Essentials 195

6.7.1 Pattern Overview 195

6.7.2 Pattern Documentation and Generation 196

6.8 Two-Sidedness Argument Pattern 198

6.9 Model Adequacy Argument Pattern 202

6.9.1 Synopsis View 202

6.9.2 Main View - Post Modelling Justification 204

6.9.3 Branch View - Model Consistency Justification 206

6.9.4 Contextual Module of Pre-Modelling Justification 208

6.9.5 Pattern Features 209

6.9.6 Role of Evidence Assertion 210

6.9.7 Undeveloped Goals 211

6.10 Cross-Model Inconsistency Analysis Adequacy Pattern 212

6.11 Special Concerns 214

6.12 Summary 214

7 Evaluation 216

7.1 Introduction 216

7.2 Means of Evaluation 217

7.2.1 Running Examples 217

7.2.2 Comparison with Existing Work 217

7.2.3 Formalisation with Tool Support 218

7.2.4 Case Studies 218

7.2.5 Peer Review 220

7.3 Evaluation of Research 220

7.3.1 Evaluation of Elaborated Concepts 220
7.3.2 Evaluation of Information Models 221
7.3.3 Evaluation of Structured Processes 222
7.3.4 Evaluation of Argument Patterns 231
7.4 Conducting Further Evaluation 231
7.5 Summary 232
8 Conclusions and Future Work 233
8.1 Thesis Contributions 233
8.1.1 Conclusions on Definition of Meta-Models 233
8.1.2 Conclusions on Conceptual Framework of Confidence 234
8.1.3 Conclusions on Structured Inconsistency Analysis Method 235
8.1.4 Conclusions on Expanded Safety Argument Construction Process 236
8.1.5 Conclusions on Argument Patterns 236
8.2 Areas of Future work 238
8.2.1 Structured Expression of Evidence Assertions 238
8.2.2 Sophisticated Correspondence Specification 238
8.2.3 Mechanised Tool Support for Inconsistency Analysis 239
8.2.4 Analysis and Justification of Quantitative Models 239
8.2.5 Quantification of Confidence Structures 240
8.3 Finale 240
Appendix A Argument Patterns 242
A.1 Two-Sidedness Argument Pattern 242
A.2 Safety Assessment Model Adequacy Argument Pattern 247
A.3 Cross-Model Inconsistency Analysis Adequacy Argument Pattern 259
Appendix B Structured Models in EMF 263
B.1 CoreDMM in EMF 264
B.2 EviM in EMF 267
B.3 Inconsistency Information Model in EMF 269
Appendix C Case Study: Braking System Control Unit 271
C.1 System Description 271
C.2 Model I Description 273

C.3 Model II Description 277

C.4 Inconsistency Analysis - Model I & Model II 282

C.4.1 Phase 1 282

C.4.2 Phase 2 282

C.4.3 Phase 3 283

C.4.4 Phase 4 291

C.4.5 Phase 5 292

C.4.6 Phase 6 293

C.5 Model Adequacy Argument Construction 293

C.5.1 Primary Safety Argument Case Study 293

C.5.2 Model Adequacy Argument Case Study 295

Glossary 300

References 302

List of Figures

Figure 1 Interaction of System Development and Safety Analysis..25

Figure 2 A FHA Example..38

Figure 3 An Example of a Fault Tree 41

Figure 4 ARP 4754A Safety Assessment Process Model 45

Figure 5 Power Hierarchy among Some Safety Analysis Models..49

Figure 6 An Example Argument Presented in GSN..56

Figure 7 Argumentation Meta-Model...58

Figure 8 Key Parts of Software Assurance Evidence Meta-Model...61

Figure 9 Formal Assertion Class in SAEM 61

Figure 10 An Example Use of ACPs...63

Figure 11 OMG Meta-Modelling Infrastructure and Safety Meta-Models 67

Figure 12 A Domain Model of Safety Assessment Modelling 70

Figure 13 Safety Assessment Core Data Meta-Model (CoreDMM)...77

Figure 14 CoreDMM Instantiation of a Hypothetical Analysis...81

Figure 15 CoreDMM Instantiation of a Record in FMEA 82

Figure 16 Part of a Fault Tree 82

Figure 17 CoreDMM Instantiation of a Part of a Fault Tree...83

Figure 18 An Example of AltaRica Data Flow Declaration...84

Figure 19 CoreDMM Instantiation of a Valve Defined in AltaRica...84

Figure 20 Relationship of Safety Assessment Artefact and Assurance Meta-Models.......89

Figure 21 Toulmin's Argument Model...100

Figure 22 Simplified View of Argument-Evidence Interface 102

Figure 23 Representation of Argument-Evidence Relationships in GSN...........................104

Figure 24 Representation of Argument-Evidence Relationships in CAE...........................105

Figure 25 A Model of Evidence in Safety Cases (EviM)..107

Figure 26 Asserted Relationships in ARM...110

Figure 27 Evidence Assertion – a Subtype of Claim 111

Figure 28 Argument Links between Evidence and Argument 112

Figure 29 Role of Evidence Result Assertion Presented in GSN 118

Figure 30 A Typology of Evidence Assertions 121

Figure 31 Framework of Confidence in Safety Cases 125

Figure 32 Typology of Safety Model Consistency 131

Figure 33 Consistency of Safety Assessment Models 132

Figure 34 Information Model of Inconsistency Analysis 144

Figure 35 Six Phases of Cross-Model Inconsistency Analysis 145

Figure 36 A Hypothetical System S_x 146

Figure 37 A Fault Tree Model of S_x 146

Figure 38 OCAS AltaRica Model and MCS Outputs 147

Figure 39 AltaRica Definition of Node 'comA' in OCAS 148

Figure 40 Equivalent Fault Tree of OCAS MCS Outputs 148

Figure 41 Phase One Flowchart 150

Figure 42 Phase Two Flowchart 151

Figure 43 Phase Three Flowchart 156

Figure 44 Phase Four Flowchart 162

Figure 45 Illustration of Roles of Evidence 169

Figure 46 The Safety Argument Fallacy Taxonomy 171

Figure 47 Counter Consideration Representation 172

Figure 48 Representation of Counter Evidence 173

Figure 49 Representation of Counter Evidence Relationship 174

Figure 50 Six Steps of Argument Construction in GSN 176

Figure 51 Overall Expanded Process of Structured Argument Construction in GSN 179

Figure 52 Example of Basis of a Goal 181

Figure 53 Example of Basis of a Strategy 183

Figure 54 An Example of Parallel Strategies 188

Figure 55 Refined Product-Branch of the Framework of Confidence in Safety Cases 191

Figure 56 Overall Framework of Confidence in Safety Cases 192

Figure 57 Counter Evidence for Argument-Evidence Interface 194

Figure 58 Expansion of ACP Concept 194

Figure 59 A Generic Argument with Absence of Counter Evidence 198

Figure 60 An Argument with Presence of Counter Evidence 199

Figure 61 Two-Sidedness Argument Pattern 200

Figure 62 Use of Model SAM$_x$ as Supporting-Evidence 203

Figure 63 Synopsis View of Model Adequacy Argument Pattern 203

Figure 64 Main View - Model Adequacy Argument Pattern 205

Figure 65 Branch View - Model Consistency Argument Pattern 207

Figure 66 Argument Pattern of Pre-Modelling Justification 209

Figure 67 Cross-Model Inconsistency Analysis Adequacy Argument Pattern 213

Figure 68 Example System Architecture 219

Figure 69 OCAS.4761.WBS in OCAS Cecilia 223

Figure 70 FT.4761.IB in OpenFTA 224

Figure 71 Example Primary Safety Argument in the Case Study 226

Figure 72 Example Model Adequacy Argument in ASCE 227

Figure 73 Safety Assessment Core Data Meta-Model in EMF 264

Figure 74 Model of Evidence in EMF 267

Figure 75 Information Model of Inconsistency Analysis in EMF 269

Figure 76 WBS and BSCU Architecture 272

Figure 77 Main View of the WBS Model (OCAS.4761.WBS) 273

Figure 78 Independent BSCUs and the Components of a BSCU 274

Figure 79 Excerpt of MCS Results of FC-WBS2 in OCAS.4761.WBS 276

Figure 80 Graphical View of BSCU Fault Tree Structure (FT.4761.IB) 278

Figure 81 BSCU Fault Tree (FT.4761.IB) in OpenFTA and MCS Result 281

Figure 81 MCSs in OCAS.4761.WBS for Comparison 288

Figure 82 MCSs of Simplified FT.4761.IB 288

Figure 83 Primary Argument with BSCU Argument Elements 294

Figure 84 Example OCAS.4761.WBS Model Adequacy Argument 297

Figure 85 Branch of Model Component Representation Justification 298

Figure 86 Branch of Model Consistency Justification 299

List of Tables

Table 1 An Example of a FMEA Worksheet 39

Table 2 Fault Tree Symbols 40

Table 3 An FMEA Record 81

Table 4 Examples of Evidence Represented in Notations 103

Table 5 Examples of Evidence Result Assertion 116

Table 6 Examples of Evidence Descriptive Assertion 120

Table 7 Phase One Output in the Running Example 151

Table 8 Examples of Consistency Relationships 152

Table 9 Elicited Consistency Relationships of the Running Example 154

Table 10 Example Correspondences between 'Condition's in the Running Example 157

Table 11 Example Algorithm for Checking ExCR2 158

Table 12 Example Correspondences between 'MCS's in the Running Example 159

Table 13 Example Algorithm for Checking ExCR3 160

Table 14 Consistency Relationships of the Case Study 224

Table 15 Failure Events of Components in OCAS.4761.WBS 274

Table 16 WBS Failure Condition Definitions in OCAS.4761.WBS 275

Table 17-1 Event Description of FT.4761.IB 279

Table 18 Phase One Output of the Case Study 282

Table 19 Consistency Relationships Defined in the Case Study 282

Table 18 Initial Relations between Conditions of Two Models 284

Table 21 Refined Relations between Conditions of Two Models 286

Table 20 FT.4761.IB MCS Transformation 289

Table 21 OCAS.4761.WBS MCS Transformation 290

Table 22 Explanation of Identified Violations 292

Acknowledgements

First of all, I would like to thank my supervisor, Dr Tim Kelly, for his patience, constant encouragement, invaluable guidance and support. It is my pleasure and has been a valuable experience to have worked with such a kind, dedicated and persistent person.

This study has been funded by EC FP7 MISSA (More Integrated Systems Safety Assessment) project and the Department of Computer Science. I'm grateful to the project partners who had helped me extensively in terms of understanding the state-of-art of safety assessment practice in industry and their concerns about the validity of safety assessments. In particular, I would like to thank Jean-Pierre Heckman, Chris Papadopoulos and Christel Seguin for their time and discussions.

I would like to thank friends and colleagues from the Department of Computer Science, in particular, Rob Alexander, Katrina Attwood, Kester Clegg, George Despotou, Xiaocheng Ge, Ibrahim Habli, Richard Hawkins, Dimitris Kolovos, Oleg Lisagor, John McDermid, Mark Nicholson, Ru Niu, David Pumfrey, Andrew Rae, Louis Rose, Kun Wei. They have enriched me with valuable ideas and resources.

Finally, I am indebted to my family. Without their understanding and full support, it is impossible to launch and toil through this solitary journey of study, from which I have gained more than I have offered.

Author's Declaration

Some of the materials presented within this thesis have previously appeared in the following publications:

- L. Sun, T. Kelly, "Safety Argument in Aircraft Certification", in Proceedings of the 4th IET International System Safety Conference, October 2009, London, UK

- L. Sun, T. Kelly, Appendix B of MISSA Project Deliverable D6.20 Development Description Report Issue B, 2011

- L. Sun, T. Kelly, "Evaluating Safety Analysis - a Meta-Model Perspective", in Proceedings of the 29th International System Safety Conference (ISSC), August 2011, USA

- L. Sun, O. Lisagor, T. Kelly, "Justifying the Validity of Safety Assessment Models with Safety Case Patterns", in Proceedings of the 6th IET System Safety Conference. September 2011, Birmingham, UK

- L. Sun, T. Kelly, "Managing Inconsistency in Safety Analysis: an Initial Exploration", in Proceedings of the European Safety and Reliability Conference (ESREL2011), September 2011, Troyes, France

- L. Sun, C. Papadopoulos, K. Mehta, T. Kelly, J. Heckmann, D. Mulloy, A. Larkham, "On the Synthesis and Validation of Safety Assessment Models", in the Proceedings of the SAE 2011 AeroTech Congress, October 2011, Toulouse, France

- L. Sun, W. Zhang, T. Kelly, "Do Safety Cases Have a Role in Aerospace Certification?", in Proceedings of the 2nd International Symposium on Aircraft Airworthiness, October 2011, Beijing, China.

Except where stated, all of the work contained within this thesis represents the original contribution of the author.

A little learning is a dangerous thing.

Alexander Pope (1688 - 1744)

An Essay on Criticism, 1709

Intentionally blank

1 Introduction

Models are widely used for problem solving. They provide us with a means of describing the real-world problems that we observe or study, of recording our understanding of these problems, of making predictions of these problems, and sometimes of helping us to manage or control the problems being modelled.

However, models could be ‘wrong’ if incompletely or incorrectly constructed, insufficiently validated, or improperly used. The following accident examples illustrate the existence and potentially harmful impact of inadequate models and/or the inadequate use of models, and indicate a significant need for a better understanding of various types of models and their use as evidence in safety cases.

1.1 The Crater Model of Space Shuttle Columbia

On 1 February, 2003, Space Shuttle Columbia broke up into debris on re-entry into the atmosphere on its return to Earth, with a complete loss of the shuttle and the seven crew members. It was Columbia’s 28th mission, coded as STS-107. According to the Columbia Accident Investigation Board (CAIB), the immediate cause of the accident was a breach in the thermal protection layer on the left wing [13]. During the launch, a foam insulation tile fell down and hit the front edge of the left wing. The strike led to a damaged area that permitted hot energy to penetrate and destroy the shuttle rapidly in the re-entry.

A ‘Crater’ model had been used for the assessment of the damage of thermal protection after the strike by a foam tile during Columbia’s launch. Crater is a model developed for the prediction of possible penetration and damage by external objects, such as foam, ice, and metal debris. It runs with an algorithm specifically developed during the Apollo program and has been modified and calibrated with several test results for wider application scenarios [13]. From those test results, engineers inferred that Crater was ‘conservative’ - i.e. that the model tends to predict more damage than there is in reality. However, there were still a large set of conditions under which the accuracy of Crater remained to be validated. “When used within its validated limits, Crater provides conservative predictions. When used outside its validated limits, Crater’s precision is unknown” [13], as stated in the Columbia Accident investigation report presented by CAIB. In the STS-107 mission, the estimated size of the foam tile (according to video and photo image analysis) was at maximum 640 times larger (about 400 times larger estimated by the CAIB later) in volume than the validated input domain of the Crater model. The use of Crater in this situation was absolutely uncertain and the interpretation of the prediction results could not be convincingly grounded.

For Columbia STS-107, Crater predicted a degree of damage which was deeper than the thickness of the actual protection tile. But it was inconsistent with the results of some calibration tests with small projectiles which showed a less deep penetration. Additionally, Crater did not take into account the increased density of the lower tile layer model. Therefore, engineers misjudged that the actual damage from a foam collision in Columbia STS-107 would not be severe enough to cause failure of the thermal protection system. By contrast, the final test of a series of foam impact tests after the accident (that simulated collisions with similar left wing structures and similar foam projectiles) showed that the collision could produce a hole on the skin as big as 41cm by 42.5cm [121]. The test result comprises overwhelming evidence to disprove the impression that the potential breach damage of the front wing structure, resulted by a foam strike, was non-threatening.

The inappropriate use of the Crater model in STS-107 is addressed specifically by the findings in the accident investigation report presented by CAIB.

> *F3.8-6 NASA's current tools, including the Crater model, are inadequate to evaluate Orbiter Thermal Protection System damage from debris impacts during pre-launch, on-orbit, and post-launch activity*. [13]
>
> *F6.3-11 Crater initially predicted tile damage deeper than the actual tile depth, but engineers used their judgment to conclude that damage would not penetrate the densified layer of tile. Similarly, RCC damage conclusions were based primarily on judgment and experience rather than analysis.* [13]

Crater, as a model, illustrates the importance of the valid usage of models and simulations in the engineering domain. As a response to the CAIB report, NASA initiated an effort to formulate a standard for the development, documentation and operation of models and simulations [38]. In 2008, NASA-STD-7009 Standard for Models and Simulations [151] was released as guidance for engineering requirements for models and simulations (M&S) in a wide range of applications. NASA-STD-7009 is not a prescriptive standard with specifications on how to satisfy the requirements through M&S processes; rather, it gives objectives on what should be done or achieved by M&S being used in decision-making. In the standard, some key requirements of M&S aimed at reducing the risks associated with critical decisions based on M&S are highlighted. These include requirements concerning verification, validation, uncertainty quantification, training, credibility assessment, documentation, and configuration management. It is worth noting that the standard seriously recommends the use of credibility assessment when using results from M&S. *Verification* and *Validation* during M&S development are considered for the credibility of M&S in a wide intended application scope; *Input Pedigree, Results Uncertainty* and *Result Robustness* during M&S operations are considered for the credibility of M&S in a particular application; *Use History*, *M&S Management*, and *People Qualification* are considered as cross-cutting supporting evidence for the overall credibility of M&S. The explicit recommendations made by the standard concerning these credibility factors are a significant improvement towards assuring the proper use of M&S results.

1.2 The Loss of Nimrod XV230

On 2 September 2006, Nimrod XV230, a RAF aircraft, crashed during a mission over Afghanistan, with a total loss of the aircraft and the 14 crew on board. This catastrophic accident was initiated by fuel either leaking from joint positions or overflowing during an air-to-air refuelling before the aircraft headed to its operational area [91]. Some of the leaked fuel accumulated in a congested Tank Bay and was in contact with one of the areas with exposed high-temperature cross-feed ducting. The fuel in the hot section first auto-ignited and then ignited fuel in other areas. Within minutes, the midair fire spread out and the aircraft exploded and crashed without any chance of recovery.

The Nimrod aircraft had been approved for operation by means of a traditional safety assessment and safety case acceptance process. On December 4 2007, an independent review on this disaster was announced (chaired by Charles Haddon-Cave QC) in order to examine the safety assessment activities related to Nimrod, to assess the overall process of safety case construction, to draw out lessons to be learned and to make practical recommendations. The review report was released on 28 October 2009. The loss of the Nimrod was deemed as the result of “a failure of leadership, culture and priorities” [91]. The report covers three main areas: the physical causes, the safety case, and the organizational causes. The safety case of Nimrod was described as “seriously defective” [91] with poor planning, poor management, and poor execution. Some of the safety analysis results, which had been adopted as evidence to justify the safety of the aircraft in the Nimrod safety case, were fragmentary and flawed. The lack of proper attention and effort on the quality of safety analysis, unfortunately, combined with other factors, resulted in the failure of the Nimrod safety case. It is stated in the Nimrod review report [91] that:

> *There was a big hole in its analysis: BAE Systems had left 40% of the hazards "Open" and 30% "Unclassified". The work was, in any event, riddled with errors of fact, analysis and risk categorisation. The critical catastrophic fire hazard relating to the Cross-Feed/SCP duct (***Hazard H73****) had not been properly assessed and, in fact, was one of those left "Open" and "Unclassified".* [91]

As described, the aforementioned Hazard Analysis by BAE Systems does not appear to be 'fit for purpose' as evidence to demonstrate 'safety', because it demonstrates only an incomplete and undeveloped understanding of the potential hazards that are left unexamined, with no effective control measures planned and verified. Unfortunately, this unfinished and inadequate hazard analysis was not properly questioned and reviewed in the development of the Nimrod safety case. A good opportunity to detect and control the catastrophic *Hazard H73* was missed, and the hazard survived as a flaw in the design modification and operation of Nimrod. However, safety cases are not a remedy for eliminating all residual problems left behind by poor analyses, inadequate outputs of safety assessment activities and unsatisfactory safety audits. It is dangerous to use safety cases as the final barrier or a single barrier to examine 'whether the system is safe or not'. Although the safety case was criticised, Mr Haddon-Cave agreed that:

> *The Safety Case concept has a useful role to play as an Airworthiness management tool for MOD military platforms. It provides a useful vehicle and reference point for risk management and, properly applied, should "encourage people to think as actively as they can to reduce risks".* [91]

Safety cases, therefore, as a rationale and an approach to aid thinking, should not be treated merely as documentation, or as a substitute for comprehensive safety assessment and rigorous and independent reviews of the results of this safety assessment. The role of safety cases is to synthesize various forms of evidence and to clarify the reasoning gaps between safety claims and evidence items, in particular, with critical and active thinking. The validity of safety cases depends on both the validity of the argument structures and the validity of the safety evidence referenced in the argument structures. The basis of any safety case is the factual existence and soundness of the safety evidence presented. Any flawed and/or incomplete safety analysis results will fatally undermine the validity of the safety case. Accordingly, the quality of safety evidence must be addressed and justified; otherwise there can be no confidence in the achievement of safety objectives (concerning the risk posed by hazards in systems in development and operation) demonstrated by safety cases.

1.3 Lessons Learnt

Safety modelling and analysis are common and crucial in the engineering of modern complex systems. However, it is not easy to determine whether safety modelling and analysis activities have been properly carried out and therefore have resulted in trustworthy results on which we can base our decisions. The two inquiry reports, the Columbia accident investigation report [13] and the Nimrod review [91], provide a comprehensive view of the unsatisfactory status of much of the current practice on safety oversight, safety guidance, the management of safety requirements, and safety analysis. There are many lessons that can be learnt from the two accidents, from the ethical perspective, the organizational perspective, the technological perspective, and the evaluation perspective. The main lessons that we can learn from the findings related to models and analysis results used for critical decision-making are presented below.

- A model can be right or wrong, depending on its content, its validity envelope and its context of usage. It usually represents some degree of truth about the subject being modelled from a certain view point, but should not be trusted unconditionally.

- Trust in models (and/or their results) comes from in-depth understanding of those models and the application context. This should be obtained from purposefully-performed assessment activities, such as analyses, reviews and tests.

- Intuition, past knowledge and experience of models or a particular problem domain may engender a 'complacent' view of the capability and validity of previously-justified models. We should not be quick to dismiss any contradictory, inconsistent, or undesired results arising from our usage of models. Instead, we should be open and active to them and investigate them thoroughly.

- When employed as evidence, models are the grounds on which a safety argument is based. The validity of safety cases collapses if the quality of models cannot be assured or if they are not fit for the role of evidence for a specific branch of argument. Therefore, models must be sufficiently validated or assessed, as far as is practicable.

In the following section, we will introduce an important type of model used in the system lifecycle for safety-critical systems, namely safety assessment models – a common form of safety evidence in safety cases.

1.4 Safety Assessment Models

Safety assessment is a series of analytical and evaluative activities concerning system safety objectives that are carried out during system total life cycle. Safety assessment models are the outputs from these activities, and may be presented either in a tabular form, or in a graphical form, or in a numerical form. The development of safety assessment models is ideally integrated with system design, implementation and operation. Safety is a system property [129] that should be considered and designed into systems from the early development stages. In this thesis, we focus on typical qualitative safety assessment models at system development stages (not operational stages), depicting the safety characteristics of engineered systems (designed or real) in a given operational context, because "system safety emphasizes qualitative rather than quantitative approaches" [129] and models developed throughout this period are the most challenging ones regarding the problems of model validity and confidence in evidence. When safety assessment models are mentioned in the thesis, we do not mean methods or modelling techniques associated with safety analysis.

As a system design evolves, a multitude of safety assessment models will be produced and updated by different people, based on a variety of techniques. Figure 1 illustrates an example 'V' model of a safety analysis lifecycle. On the left-hand side, there is a top-down process, through which hierarchical safety requirements are identified and allocated according to various safety analyses at different levels, e.g. Hazard Identification. On the right-hand side, there is a bottom-up process, through which integration and requirement verification are carried out at different levels. The purpose of the safety process is to support companies in planning safety tasks and in showing compliance with system safety requirements. As illustrated in Figure 1, the overall safety analysis process in the outer 'V' interacts with the system development activities in the inner 'V' extensively.

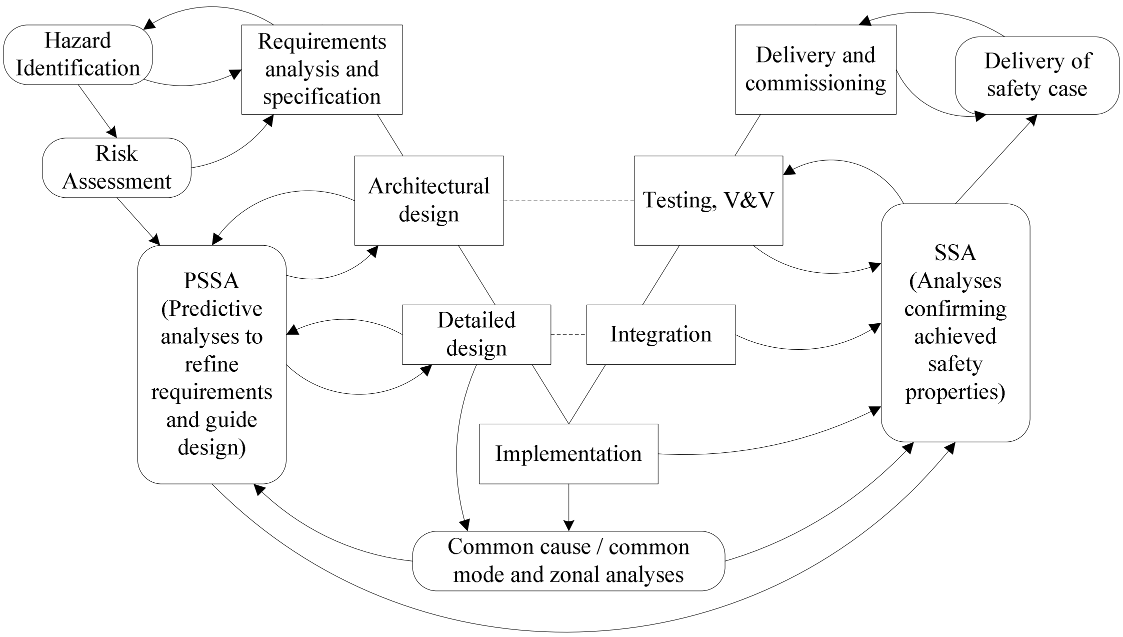

Hazard Identification
Requirements analysis and specification
Delivery and commissioning
Delivery of safety case
Risk Assessment
Architectural design
Testing, V&V
PSSA (Predictive analyses to refine requirements and guide design)
SSA (Analyses confirming achieved safety properties)
Detailed design
Integration
Implementation
Common cause / common mode and zonal analyses

Figure 1 Interaction of System Development and Safety Analysis (from [168])

Safety assessment models can be classified according to their own features, or the methods on which they are constructed, or the phases of a system life-cycle in which they are developed. The domain safety concerns and issues of a system are explored during various safety analyses with the following categories of duties (adapted from [211]).

- Hazard identification (e.g. a hazard log) — safety analyses of this type are usually undertaken at the early stage of system design and will be updated according to design changes. The aim of the construction of this category of models is to identify potentially hazardous conditions in a proposed system in order to formulate the safety goals or requirements for later system development stages.

- Causal analysis (e.g. a fault tree) — safety analyses of this type generally place emphasis on presenting the causal factors of hazardous or undesired conditions in the modelling process. A causal analysis starts with an unwanted event in a particular context and ends with the identification of potential contributing factors and their combinational relationships.

- Consequence analysis (e.g. a FMEA worksheet) — safety analyses of this type focus on presenting the potential consequences of identified hazards. A consequence analysis assumes the existence of failure modes or other hazardous conditions and then assesses their effects on functions, humans, properties, or missions. Consequence analysis represents thinking in the opposite direction to causal analysis. Models generated on the basis of both types of analyses have causal-effect relationships recorded.

- Risk assessment (e.g. a risk matrix) — safety analyses of this type present data on the likelihood and severity of identified risks, which are required for making decisions on design acceptance or determining risk control measures.

- Quantitative assessment (e.g. a Markov model) — safety analyses of this type have the feature of providing quantitative values of system safety parameters, usually the probabilities of undesired conditions. The quantitative results are generated on the basis of the structure of other qualitative analysis results such as a cause-consequence model or a state model. This type of analysis can be viewed as a particular case of risk assessment – with regard to the likelihood of risk in a numerical manner.

However, it is challenging to assure the quality of the different forms of safety assessment models, either individually or as a whole, which determines whether we can have confidence in our judgment of the level of safety achieved in certification. Whilst safety assessment models attempt to model potentially random failure events (i.e. model aleatoric uncertainty), there is also epistemic uncertainty in our understanding of the system under study and the associated safety problems. In addition, there can be uncertainty in the representation of our understanding and the collection and processing of safety data. We must be cautious while using our safety assessment models as evidence to justify system safety. There are many potential factors that can affect our confidence in safety assessment models. Were a safety model to be interpreted without considering its limitations and its application scenarios, intended or novel, there would be little confidence, but only uncertainty, in our decisions on the rejection or acceptance of results from the model.

1.5 Existing Practice

After more than sixty years of development, safety practitioners in various industrial sectors are equipped with both a large collection of available analysis techniques (e.g. there are more than seven hundred safety methods listed in the NLR safety methods database [14]) and practical process guidance on the systematic implementation of these techniques. However, to date, there is insufficient guidance or best-practice on the evaluation of safety assessment models.

In the early 1980s, Lloyd argued the need for assessment of the safety models [136] for aircraft safety assessment, although there were no standards in that area at that time. Leveson, in [129], explains the difficulty of validating safety analysis and declares that, in practice, few safety analyses are actually validated. Despite the pervasive and indispensible usage of safety assessment models nowadays, they are also “a significant source of error” in system safety analysis [52].

Regulatory bodies have also acknowledged the importance of the evaluation of safety assessment models. For example, AMC 25.1309, the aerospace standard that describes acceptable means for showing compliance with the requirements of CS 25.1309, states clearly that “any analysis is only as accurate as the assumptions, data, and analytical techniques it uses” [68]. Currently, however, most of the existing guidance on the evaluation or assessment of safety analysis is concerned with the human aspects of safety reviews. There are few recommendations on how to integrate and justify multiple safety assessment models in practice. SAE ARP 4754A requires that “the outcomes of these validation method activities and their appropriateness are reviewable and should be done with independence” [180]. Unfortunately, no further concrete detail is provided on *how* to perform such a review. Furthermore, multiple safety analyses, as a body of evidence should be subjected to sufficient scrutiny for their interrelationships; otherwise, there will not be sufficient confidence in the fulfilment of the overall safety objectives of a system.

1.6 Research Problem

From the preceding discussion on lessons from accidents and current practice in safety assessment, it is clear that there is an urgent need for better guidance on how to validate and review various types of safety assessment models. However, the validity of safety assessment is notoriously difficult to deal with, especially during the system development lifecycle. An important reason for this difficulty is that safety assessment models are concerned with unintended system conditions leading to undesired consequences, which should not (and generally do not) frequently occur. It is hard (and in many cases impossible) to generate enough data in real operation or in realistic test scenarios to perform comprehensive validation prior to a system’s acceptance into operation. In addition, relying upon *real* data in this way can involve placing humans at risk. Besides that, the difficulty is exacerbated with the following factors - diversity of analysis techniques, complexity of models and systems under study, informal description with natural language and divergent format of data from models.

But it is very valuable to have some reasonable feedback as early as possible about our understanding of the quality of these models and whether they are properly used for making decisions and judgments. Therefore, we change our view towards the problem with the following questions:

- How can we improve our understanding of safety assessment models and their role as evidence in safety cases?

- How can we rigorously examine safety assessment models, especially when the models are resistant to validation against the real world?

- How can we be more confident in the adequacy of safety assessment evidence in safety cases?

As a result, our research aim is not to guarantee the validity of safety assessment models, which is heavily based upon engineering judgments and applications, but to understand safety assessment models better at the development stage of a system lifecycle and establish increased confidence in the use of safety assessment models in a given context.

1.7 Thesis Proposition

Following from the research problems described in the previous sections, the hypothesis proposed in this thesis is that:

> *The use of a* ***structured*** *approach to the integration and justification of safety assessment evidence within safety cases facilitates the identification and potential resolution of issues which may otherwise reduce confidence in safety justification practice.*

The key term 'structured' used in the thesis hypothesis refers to the following thesis contributions:

- **structured** information – clearly defined concepts, a definite and highly organised form of model and metadata for safety assessment evidence;

- **structured** processes – decomposed but integrated steps with explicit inputs and outputs for model inconsistency analysis and safety argument construction;

- **structured** guidance – systematic thinking patterns that enhance explicitness, clarity and rigour in evidence justification.

1.8 Thesis Structure

The thesis is divided into the following chapters:

Chapter 2 Literature Survey

This chapter presents a review of literature on models in general, safety analysis techniques and safety case development. The essence of models and model validation is explored; a review of major widely-used safety analysis techniques and the latest trends in safety assessment are presented. Besides that, the general means of model validation and current research and practice in evaluating safety assessment models are provided and the progress on research work in the field of safety cases is reported. Through the review, the importance of and the need to establish confidence in safety assessment evidence used to support safety arguments are clearly grounded.

Chapter 3 Safety Assessment Meta-Modelling

This chapter presents a model of the domain of safety assessment modelling and a core data meta-model of safety assessment artefacts. The domain model organises the key factors involved in and associated with safety assessment modelling and used as the context of model evaluation. The core data meta-model addresses four types of core elements in common safety assessment models (metadata, validity contextual data, substance analysis results and conventional construction elements in safety assessment models) in support of evidence application and justification.

Chapter 4 A Model of the Argument-Evidence Interface

This chapter describes a model of evidence that depicts the connotation of evidence and interfaces safety assessment models with safety arguments. The model integrates views of evidence from the content perspective, the utilisation perspective and the evaluation perspective. Importantly, the nature and characteristics of safety assessment evidence and the existing view and representations of evidence and its relationship with arguments are analyzed in detail. The notion of the *evidence assertion* is elaborated for interfacing arguments and safety assessment evidence. The relationship between evidence and confidence in safety cases is discussed.

Chapter 5 Managing Safety Model Inconsistency

This chapter presents an inconsistency analysis method that deals with the potential consistency issue associated with multiple safety analyses. The method is established on the exploration and categorisation of the consistency problem amongst safety assessment models. The inconsistency analysis method comprises six phases and is supported by a structured information model. The key analysis phases are explained with detailed steps and demonstrated with reference to a running example.

Chapter 6 Using and Justifying Safety Assessment Evidence

This chapter introduces an expanded argument construction process to underpin the establishment of confidence during safety case development, through which counter evidence and the justification of evidence are explicitly considered. Furthermore, three generic argument patterns are developed in order to support more rigorous argument construction for confidence establishment, in particular - through argumentation from both positive and negative perspectives, through justification of the adequacy of safety assessment models, and through justification of the cross-model inconsistency analysis.

Chapter 7 Evaluation

In this chapter, the contributions of the research are evaluated against the thesis proposition. The soundness of the concepts elaborated is discussed; the efficacy and practicality of the approach presented is illustrated with a case study with models of a braking system control unit in an industrial application context.

Chapter 8 Conclusions and Future Work

This chapter concludes the thesis, with highlights on the contributions of the thesis. Potential areas for further work that can be undertaken to extend and strengthen the work are also presented.

Appendix A presents three argument patterns that are associated with the issue of confidence in using safety assessment evidence.

Appendix B defines a safety assessment core data meta-model, a model of evidence and an information model of inconsistency analysis in the Eclipse Modelling Framework (EMF) [189].

Appendix C presents a case study of cross-model inconsistency analysis and argument construction on the basis of the safety assessment of a Braking System Control Unit of an aircraft Wheel Braking System taken from ARP 4761 [181].

2 Literature Survey

2.1 Introduction

This chapter presents the findings of the literature review of the research in the fields of modelling, safety assessment and safety argumentation. It aims to provide the background to the research presented in this thesis. Knowledge of modelling in general is essential for understanding the fundamental common features of various models and modelling process, which are also exhibited by safety-related models. The complexity of safety assessment models is revealed with the review of traditional and modern safety analysis techniques, typical safety assessment processes and existing evaluation efforts on safety-related models. Then the regime of safety cases, a particular application context of safety assessment models, is surveyed for the further research centred on integration and evaluation of safety assessment evidence with safety arguments.

This chapter is divided into the following sections:

- Models in general – definition of a 'model' and some common features of models
- Model validation – concept, problems and means regarding the validation of models
- Safety assessment modelling techniques – traditional qualitative techniques and techniques integrated with formal languages
- Safety assessment processes – typical safety assessment processes and process-related issues
- Evaluation of safety assessment models – existing research and practice
- Safety and assurance cases
    - Essential argumentation elements and evaluation criteria
    - Existing requirements and research on assurance, evidence and argument
    - Confidence and models in safety cases.

2.2 Models in General

Models, as a very broad concept, are used explicitly and implicitly in all disciplines. Principles and methods concerning models and modelling from other fields can aid the understanding, evaluation and utilization of safety assessment models. So before the specific discussion of safety assessment models, it is useful to review some general literature concerning models.

2.2.1 Definition

There have been many definitions of what is meant by the term ‘model’:

- “A representation of something else, designed for a specific purpose” [47].

- “An abstract description of the real world; it is a simple presentation of more complex forms, processes, and functions of physical phenomena or ideas” [174].

- “A distinct domain that corresponds by analogy to the real world domain” [107].

- “Modelling in the broadest sense is the cost-effective use of something in place of something else for some cognitive purpose”. “A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. This allows us to deal with the world in a simplified manner, avoiding the complexity, danger and irreversibility of reality.” [172]

- “An analytical representation or quantification of a real system and the ways in which phenomena occur within that system, used to predict or assess the behaviour of the real system under specified (often hypothetical) conditions.” [103]

According to these definitions, some essential characteristics of a model are summarized below.

- **It is designed or used for a specific purpose** [47, 172]. A model cannot be good for all purposes. It should be evaluated according to its contribution to the purpose it was used for.

- **It is not real** [47, 172]. The object being modelled is different from the model we designed to represent the object. A model could not represent all aspects of the object being modelled. Simplification and abstraction are two most essential features of models. Models with different abstraction levels vary from the real object in different ways.

- **It is of something** [172]. The objects being modelled are diverse and generally they are complex [174]. The object could be a device, a phenomenon, a process, a function, a concept, an idea, etc.

- **It is based on building up relationships**. There should be some common or similar attributes between models and the objects being represented [107]. An analogy, metaphor or mapping exists between a model and a problem. Different models concentrate on different relationships.

- **It should be cost effective** [172]. A model allows us to deal with things or situations which are too costly to deal with directly by using something that is simpler, safer or cheaper.

Two main purposes of a model are 'to facilitate understanding and enhance prediction' [174]. In addition to the above two purposes, Levins [130] states another purpose as 'to modify nature'. This also has been addressed as 'control' [206] or 'prescription' [143]. This aim means that we can intervene in the system being modelled in some way to achieve a state we desire. This goal does not exist for every model or under every circumstance and it can only be achieved on the basis of the achievement of the former two goals.

It is necessary to distinguish between two types of representation of models. The first type of model is intended to represent the real world, or certain aspects of the real world. Then the real world problem can be safely explored and manipulated using the models, i.e. models for weather forecasts. In this case, we can get some feedback of the quality of our models from the real world. The second type of model is intended to represent the elements that should be present in an idealization of a system under research, which could be vague, ambiguous, and nonexistent at the beginning (for example, an aircraft design model). It is difficult to obtain operational feedback for the assessment of these models in advance of a system being allowed into operation.

2.2.2 Model Building

The construction of models is a creative and subjective exercise. The core of modelling is performing abstraction by aggregating elements that are strongly connected and separating groups of elements with less strong links [174]. Furthermore, Levins [130] points out that there are tradeoffs between different attributes of models during model construction. A model with maximized generality, realism, precision, manageability, and understandability is preferred. But these attributes cannot all achieve the maximum value at the same time.

Sacrificing one of the attributes might improve others. An adequate model comes through comprehensive thinking and considered tradeoffs - a painstaking process.

In a model building process, it is normal and necessary to have assumptions and simplifications to make the complexity of the real-world problems tractable [78, 130, 174]. The reasons for making assumptions and simplifications include: a) there might be too many features or parameters to model, or b) too many features in a model would exceed our current capability of problem-solving, or c) the complicated results of complex models became meaningless or less understandable and make it more difficult to explain the problem. While making simplification in the models, we must preserve the essential features of a problem. Understanding of a problem is not achieved by general models alone, but by a relation between the general and the particular [130].

In complex systems, there is usually a cluster of models due to the complexity of a system and the capability of our mind [130]. Starting with small models and using the divide-and-conquer technique will make the construction of a model a manageable task. It is also necessary to think about the relation of a model to other models during modelling, otherwise the usefulness of a model might be weakened within a larger system context.

As Shannon recommended, we can build a model according to four simplified steps by[185]: a) specification of the model's purpose; b) specification of the model's components; c) specification of the parameters and variables associated with the components; d) and specification of the relationships between the components, parameters and variables.

2.3 Model Validation

Since more and more models are used in a variety of disciplines, there are increasing concerns about how to evaluate a model, such as "How can we tell whether a model is a good one?", "How can we judge the strengths of different models?" [161]. This section presents a survey of different views on model validation and validation methods.

2.3.1 Viewpoints on Validation

Oreskes points out that 'calling a model validated does not make it valid' [160]. She also argues that 'it is impossible to demonstrate the predictive reliability of any model of a complex natural system in advance of its actual use'. This is because usually the system being modelled is not a closed system, and the model of a system is always not unique [162]. All models embed some uncertainties [160] when the system being modelled is open. The uncertainties could be theoretical, empirical, parametrical, or temporal. These uncertainties will have influences on the reliability of model prediction. In addition, she states that models may be conceptually flawed [160]. Thus she suggests using the terms ‘model evaluation’ or ‘model assessment’ instead of ‘model validation’[1].

McCarl, in [143], also comments that ‘models can never be validated, only invalidated’. He argues that the possible results from a model validation process are: a) the model is proved to be invalid; or b) the model is confirmed with an increased degree of confidence. Hodges [101] holds the same point of view that the validity of models is not binary but accrues continuously between ‘not valid’ and ‘valid’ when they pass more validation tests.

If a model can be validated and has passed its validation process, we often call or label it with the term ‘a valid model’ even if it is not a valid model. Numerous papers on model validation use the terminology ‘validate/validation’ in this way and it is widely accepted by most modellers and model-users. Hence it is unnecessary to avoid using the concept of ‘model validation’ as long as we understand that current research on model validation is a form of *confidence building* [28] – we may have increased level of confidence in the so-called ‘valid’ models because they have passed specific validation or evaluation processes.

Miser and Quade [147] define model validation as ‘the process by which the analyst assures himself and others that a model is a representation of the phenomena being modelled that is adequate for the purposes of the study of which it is a part’. Many researchers [101, 182, 190] agree that models should be validated according to their fitness for their intended purpose. In addition, models validated at a given time are not always valid, and continuous validation in operation should be conducted [143].

2.3.2 Means of Validation

Despite the consensus on model validation according to its adequacy for purpose, most research on models has treated the validation of a model as the agreement of the model with ‘reality’. There is only limited guidance on how to evaluate whether the abstract level and the precision level of models are appropriate to their anticipated use.

Rykiel suggests three kinds of information should be clarified before model validation: “the purpose of the model”; “the criteria the model must meet to be declared acceptable for use”; “the context in which the model is intended to operate” [178]. However, such information is

[1] Some people use the term ‘model accreditation’, ‘model appraisal’.

not always easy to define and is, in practice, either scarcely covered in validation or only vaguely described.

Much existing research concerns the validation of numerical models. There is not much discussion of techniques for the validation of qualitative models. More efforts on model validation in general are needed to increase our confidence in models and their outputs. Shannon suggests that establishing the possible validity of simulation models can be achieved by the following procedures [184]:

- “Using common sense and logic throughout the study.
- Taking maximum advantage of the knowledge and insight of those most familiar with the system under study.
- Empirically testing by the use of appropriate statistical techniques all of the assumptions, hypotheses, etc. that possibly can be tested.
- Paying close attention to details, checking and rechecking each step of the model building process.
- Assuring that the model performs the way it was intended by using test cases, etc. during the debugging phase.
- Comparing the input-output transformation of the model and the real world system (whenever possible).
- Running field tests or peripheral research where feasible.
- Performing sensitivity analysis on input variables, parameters, etc.
- Checking carefully the predictions of the model and actual results achieved with the real world system.” [184]

From the suggestion above, we can observe that the main approaches taken to evaluating models can be considered in two groups: a) by judgment from human knowledge and experience; b) by comparison of a model and its results with data from a statistical test, field use, or another model. The scope of our experience and knowledge and whether comparison data is available will determine the validation approach that is applicable in a given situation.

2.4 Safety Assessment Models

Models in the safety domain exhibit the fundamental attributes of models in general as outlined above: i.e. they are not real, they are simplified and they are abstracted. Usually they are dealing with engineering systems - man-made systems interacting with natural systems. These man-made systems are specified as we think they ought to be. The safety assessment models we use to analyse or demonstrate system safety are always based on: a) a system model or at least a conceptual mental model; b) and a kind of safety analysis technique.

As we have mentioned in Chapter 1, a large collection of methods have been developed to support system safety assessment, such as the ones presented in [14, 35]. Safety analysis techniques are broadly divided into two groups: qualitative techniques and quantitative techniques [173]. In this section, safety assessment models[2] are primarily reviewed from the perspective of qualitative safety analysis techniques on which the model instances are based. Safety analysis approaches which deal essentially with causal failure relationships have been chosen for this survey. Models for systematic failures of components related to specific failure mechanisms are not covered in this survey. Models of accidents, human factors, or socio-technical issues are also beyond the scope of this study.

2.4.1 Traditional Techniques

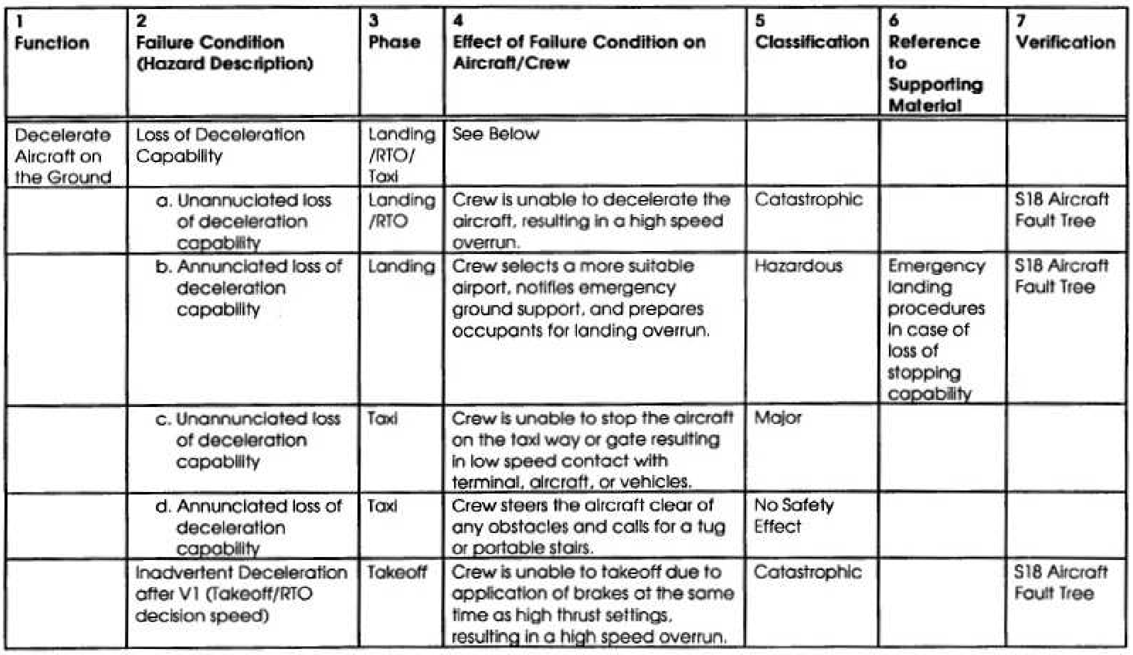
In this subsection, four common safety analysis techniques, namely, Functional Hazard Assessment (FHA), Failure Modes and Effects Analysis (FMEA), Fault Tree Analysis (FTA), Hazard and Operability Study (HAZOP), are briefly outlined and references made to published evaluations of them.

Functional Hazard Assessment (FHA)

Functional Hazard Assessment is recommended by SAE ARP 4754A [180] and is practiced at the beginning of the safety assessment process in the aviation industry. Through this approach, system functions are carefully examined and potential consequences caused by identified failure conditions are analyzed and classified according to their severity. In the aerospace domain, FHA is performed at both the aircraft level and the system level. Safety

[2] When ‘safety assessment model’ is mentioned, it could have two different meanings depending on the context of usage. The first meaning is a concrete model instance, which is the use of this term in this thesis. The other meaning is a specific type of safety analysis technique or method, which is addressed in this thesis as a safety analysis technique or a safety assessment meta-model.

objectives of both the aircraft level and the system level should be determined after FHA. An excerpt of an aircraft FHA table from SAE ARP 4761 [181] is shown in Figure 2.



1 Function	2 Failure Condition (Hazard Description)	3 Phase	4 Effect of Failure Condition on Aircraft/Crew	5 Classification	6 Reference to Supporting Material	7 Verification
Decelerate Aircraft on the Ground	Loss of Deceleration Capability	Landing /RTO/ Taxi	See Below			
	a. Unannuciated loss of deceleration capability	Landing /RTO	Crew is unable to decelerate the aircraft, resulting in a high speed overrun.	Catastrophic		S18 Aircraft Fault Tree
	b. Annunciated loss of deceleration capability	Landing	Crew selects a more suitable airport, notifies emergency ground support, and prepares occupants for landing overrun.	Hazardous	Emergency landing procedures in case of loss of stopping capability	S18 Aircraft Fault Tree
	c. Unannunciated loss of deceleration capability	Taxi	Crew is unable to stop the aircraft on the taxi way or gate resulting in low speed contact with terminal, aircraft, or vehicles.	Major		
	d. Annunciated loss of deceleration capability	Taxi	Crew steers the aircraft clear of any obstacles and calls for a tug or portable stairs.	No Safety Effect		
	Inadvertent Deceleration after V1 (Takeoff/RTO decision speed)	Takeoff	Crew is unable to takeoff due to application of brakes at the same time as high thrust settings, resulting in a high speed overrun.	Catastrophic		S18 Aircraft Fault Tree

Figure 2 A FHA Example (from [181])

FHA gives no support for the analysis of functional dependencies. The method does not generate any further information other than that input by analysts. It only presents our understanding about the loss of system functions and the associated effects. “Generation of the FHA at the highest appropriate level is dependent upon overall knowledge and experience and may require consultation of numerous specialists” [181].

There are some implementation issues with FHA, since it is presented in a purely textual format. It is hard to clearly define the functions and failure conditions with unambiguous and adequate words [210]. In practice, the completeness of failure conditions can also be difficult to achieve.

Failure Modes and Effects Analysis (FMEA)

Failure modes and effects analysis is one of the most commonly used and most effective reliability analysis methods [155]. It is an inductive analysis by which each of the potential failure modes in the system is analyzed to determine its resulting effects, and the resulting failure modes are classified by their severities and likelihood of occurrence. The findings of FMEA are usually documented textually in worksheets. A sample worksheet with typical column headings used by Electricité de France (excerpted from [201]) is shown in Table 1. Besides its use in reliability analysis, this method also provides useful information for maintainability and safety analysis.

FAILURE MODE AND EFFECTS ANALYSIS

PROJECT ___________

SYSTEM ____________ REFERENCE DOCUMENTS ________

Component identification (Code,Name, Type, Location)	Functions, States	Failure modes	Possible failure causes (internal, external causes)	Effects on the system	Effects on external systems	Means of detection	Inspection and test frequency	Comments

Table 1 An Example of a FMEA Worksheet (from [201])

Although it is easy to learn and use, this method has its limitations. It usually considers every single unit failure independently [129, 201], meaning that it is ineffective in dealing with multiple failures under diverse conditions in complex systems. The operating procedure of this method is not well-suited for the consideration of human factors [129]. In addition, it is difficult to trace the effects of low-level failures correctly through FMEA of complex systems [155].

FMEA is a method with limited analysis power. The effectiveness and quality of the analysis are heavily dependent on the knowledge and experience of the person performing the analysis. There is no support from this method to identify new failure modes [129]. “All the significant failure modes must be known in advance” [129]. Though it is clear that the purpose of FMEA is to examine possible failure modes and determine their impact on the product, the most significant problem that challenges the effectiveness of FMEA is the omission of failure modes [42]. The completeness of the analysis is always challenged by the ‘unknown unknowns’. Besides that, it does not provide means to determine failure effects or mitigation measures. The power of this method lies in the humans who perform the analysis.

Furthermore, FMEA is a task involving many hours of human effort. Practitioners always complain that doing FMEA is tedious and time-consuming [35, 201], especially for complex systems [129].

Fault Tree Analysis (FTA)

Fault tree analysis is one of the most popular safety analysis method used in safety engineering. Based on Boolean logic and with graphical representation, this method starts from an undesired system failure (top event) and continues by top-down analysis of possible causes of individual or combined part-level failures (basic events). The failure events and the logical connections between them are presented by standardized symbols. The traditional fault tree symbols are described in Table 2 (from [200]).

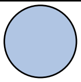
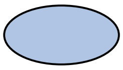
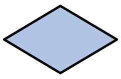
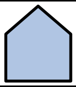
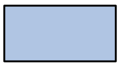
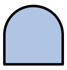
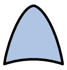
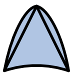
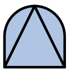
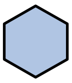
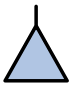
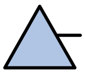
Primary Event Symbols	
	BASIC EVENT - A basic initiating fault requiring no further development
	CONDITIONING EVENT - Specific conditions or restrictions that apply to any logic gate
	UNDEVELOPED EVENT – An event which is not further developed either because it is of insufficient consequence or because information is unavailable
	EXTERNAL EVENT – An event which is normally expected to occur
Intermediate Event Symbols	
	INTERMEDIATE EVENT – A fault event that occurs because of one or more antecedent causes acting through logic gates
Gate Symbols	
	AND – Output fault occurs if all of the input faults occur
	OR – Output fault occurs if at least one of the input faults occurs
	EXCLUSIVE OR – Output fault occurs if exactly one of the input faults occurs
	PRIORITY AND – Output fault occurs if all of the input faults occur in a specific sequence
	INHIBIT – Output occurs if the (single) input fault occurs in the presence of an enabling condition
Transfer Symbols	
	TRANSFER IN – Indicates that the tree is developed further at the corresponding TRANSFER OUT
	TRANSFER OUT – Indicates that this portion of the tree must be attached at the corresponding TRANSFER IN

Table 2 Fault Tree Symbols (from [200])

An example of a simple fault tree from US Nuclear Regulatory Commission Fault Tree Handbook [200] is given in Figure 3.

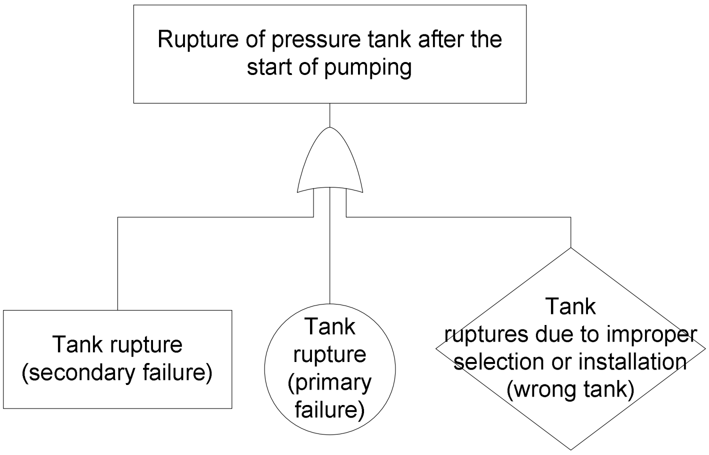

Rupture of pressure tank after the start of pumping
Tank rupture (secondary failure)
Tank rupture (primary failure)
Tank ruptures due to improper selection or installation (wrong tank)

Figure 3 An Example of a Fault Tree (from [200])

FTA is capable of considering multiple failures and diverse types of failure causes (e.g. human error, maintenance error). It is a deductive approach [200]. The qualitative analysis of FTA can generate Minimal Cut Sets (MCSs), which are specific minimum combinations of failures that lead to the top event. The quantitative analysis of FTA can provide the probability of the top event if data concerning all of the basic events is available. Moreover, there are methods for assessing the importance of events which is very useful for prioritizing measures for reducing safety risk.

However, a fault tree is only a simplified representation of complex system failure-cause relations. The fundamental assumption of its quantitative analysis is the independence of primary events in a tree [129, 201]. If this assumption is not valid for a real system, the FTA of that system is invalid. Although FTA is good for providing an intuitive and simple representation of failure-cause logical relations, the traditional logic gate symbols are not capable of displaying failures associated with time-ordering or time delay. FTA works well with binary states of a system (work or fail), but is weak at representing state transitions and partial failures in complex systems [129].

The depth of FTA is limited by the availability of information on system structure. The correctness of logical relations and the completeness of identified relations are to a large extent decided by human knowledge and experience [165]. For complex systems, constructing a fault tree manually becomes an infeasible and unmanageable task [201]. If there are many undesired system level failures, separate fault trees need to be built for each of these top events [181].

For the reasons above, many studies have been focused on the improvement of FTA for increased expressive power and easier means of construction. Some of the studies have focussed on introducing new gates for failure logic description, such as the ‘Priority-AND (PAND)’ gate [67], the ‘AND-THEN (TAND)’ gate [209], the ‘Priority-OR (POR)’gate [202], the Simultaneous-AND gate [202]. But these new gates are used much less frequently in practice because they are less familiar to people who are used to traditional FTA. Furthermore, these extended fault-tree-like models with new gates no longer have the qualitative analysis part of FTA. Minimal cut sets are not given because the concept of MCS does not work while considering timing and sequence scenarios in the analysis of trees with new gates. Only the probability of an undesired top event is provided as the final analysis result of trees with new gates. Similar to traditional fault trees, the probability provided by the quantitative analysis of trees models with new gates is also based on the structures of tree models.

Hazard and Operability Study (HAZOP)

HAZOP [124] was initially developed in the process industry. It is a structured and systematic examination of potential conditions that may endanger personnel, equipment, or task. By using a set of recommended *guidewords* (e.g. No, More, Less, As Well As), we can intentionally deviate the behaviour or states of system components and analyze its possible causes, consequences, and potential means of treatment. The HAZOP meeting process and participants from multi-disciplinary team are important factors influencing the effectiveness of the analysis results. Besides its usage in process industries, this method has also been extended for analyzing other types of system, such as software [168], human errors or operational procedures [124].

2.4.2 Safety Analysis Based on Formal Languages

More recently, researchers have explored the possibility of performing safety analysis based on models described in formal languages, which is also called ‘model’-based safety analysis [44, 111, 132, 163], and is largely driven by the motivation of achieving reusable and automated safety analysis. An important notion associated with these models is the idea of failure propagation and transformation.

Failure Propagation Transformation Notation (FPTN) [76] is a graphical notation used to describe the causal relationships between failures. This notation encapsulates a number of related failure relationships into a FPTN module. The relations in a module might vary from causal relations extracted from FMEAs to causal relations represented by FTAs. The failures contained in a module are not restricted to the range of failures of a single component, or failures of a single type, or failures of a same product level. FPTN modules can be connected together by matching failure-inputs and failure-outputs, rather than by the normal data flow, to illustrate the failure propagation and transformation with a ‘bigger’ module. This notation provides a more manageable way to trace failure effects back to failure modes or conditions between different levels. Failures in a FPTN module change and spread according to logical rules which are similar to the logic deployed in FTA.

In [44], system safety assessment is supported by FSAP/NuSMV-SA, a platform with a graphical interface (FSAP) and a specialised model-checker for safety assessment based on NuSMV 2 [49]. The safety assessment is based on formally-defined models - a formal design model of a system (the system model), a formal model of desired properties of the system, and a formal model of expected failure modes of the components of the design model. An extended system model is formulated by enriching the system model by injecting previously defined failure modes into it. Then the enriched model is checked with formalized safety requirements via exhaustive state-space analysis to identify all sets of basic events which can trigger a top level event. The NuSMV-SA model-checker will automatically extract all these collections of basic events to construct a fault tree.

One difference between this fault tree and a traditional one is that this tree is a two-layered fault tree constructed with basic events from detected minimal cut sets under ‘AND’ gates, and an ‘OR’ gate underneath the top event. By contrast, traditional FTA typically has more intermediate layers and the layout of gates is usually not so ‘tidy’. The top-down mode of thinking inherent in the process of developing a traditional FTA does not exist in those automatically-generated fault trees.

AltaRica is a high-level formal modelling language with an unambiguous semantics [23, 43]. The language is capable of describing both functional and dysfunctional logical features of the system being modelled [43]. Full AltaRica contains features (e.g. handling of state) that are difficult to formulate in simple Boolean formulae [43]. A sub-branch of AltaRica that excludes those features is called AltaRica data-flow language. The AltaRica Data-Flow modelling practice is well in line with the failure transformation and propagation description needed by safety analysis. An example of a component modelled in AltaRica is presented in Section 3.4.2 later. The modelling and analysis can be supported by industrial tools [1], such as OCAS Cecilia developed by Dassault Aviation, Simfia V2 by EADS APSYS and Safety Designer by Dassault Systèmes. AltaRica has been used for safety assessment of complex systems [31, 32] and multi-physical systems modelling [19].

However, physical system failure propagations could be difficult to model adequately (for instance, if a component failure is linked with bidirectional flows). AltaRica does not differentiate between transient and permanent faults. The AltaRica data model does not permit syntactical circular definition; time-delays need to be added to avoid loops. Furthermore there is no guarantee that all scenarios that cause one failure condition will be found [31]. Lisagor [132] suggested some ways to circumvent some of the limitations of AltaRica modelling, such as logical loops and reconfiguration handling, in the failure logic modelling (FLM) and analysis process. Adeline et al explored a validation process for AltaRica models [18].

It is worth noting that these safety assessment models based on formal languages are different from traditional fault trees in nature. Indeed, they are not top-down analysis, but bottom-up analysis. Even though they have MCSs as the modelling outputs; the model-based analysis should not substitute for traditional FTA in the safety assessment process. Ortmeier et al [163] suggests that it is beneficial to perform traditional fault tree analysis independently from formal safety analysis models. It would be helpful to enable comparison between traditional fault trees and model based safety analysis results in order to cross-check our understanding of the system behaviours to some degree.

2.5 Safety Assessment Processes

No single technique is powerful enough to deal with the representation and analysis requirements for safety assessment of all kinds of systems at different development or operational stages. Hence, safety assessment processes, which incorporate safety activities based on different safety analysis techniques, are usually recommended and are widely adopted in combination with other system engineering processes [180]. In this section, a typical safety assessment process is outlined and issues concerning the enactment of safety assessment processes are briefly discussed.

2.5.1 SAE ARP 4754A Safety Process

In the domain of civil aviation, ARP 4754A [180] provides overall guidance for the development of civil aircraft and systems. It defines the safety assessment process and deliverables required along with the aircraft/system development process and other integral processes, taking into account the overall aircraft operating environment and functions. Both requirements validation and implementation verification are stressed for certification and product assurance in ARP 4754A.

As described in Section 1.4, a variety of safety assessment activities span the system development process. Figure 4 shows the detailed interactions between safety assessment and aviation system development activities and demonstrates clearly how the outputs of safety assessment are interrelated.

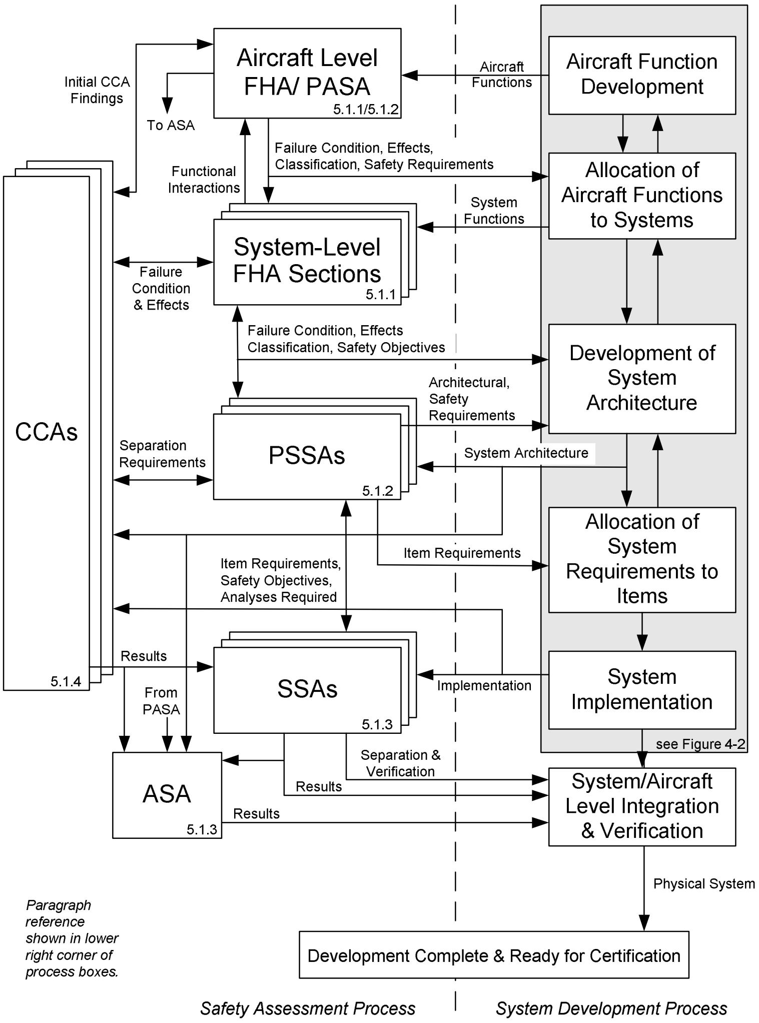

Aircraft Level FHA/ PASA
5.1.1/5.1.2
Initial CCA Findings
To ASA
Aircraft Functions
Aircraft Function Development
Functional Interactions
Failure Condition, Effects, Classification, Safety Requirements
Allocation of Aircraft Functions to Systems
System Functions
System-Level FHA Sections
5.1.1
Failure Condition & Effects
Failure Condition, Effects Classification, Safety Objectives
Development of System Architecture
Architectural, Safety Requirements
CCAs
Separation Requirements
PSSAs
5.1.2
System Architecture
Allocation of System Requirements to Items
Item Requirements
Item Requirements, Safety Objectives, Analyses Required
Results
5.1.4
From PASA
SSAs
5.1.3
Implementation
System Implementation
see Figure 4-2
ASA
5.1.3
Separation & Verification
Results
Results
System/Aircraft Level Integration & Verification
Physical System
Paragraph reference shown in lower right corner of process boxes.
Development Complete & Ready for Certification
Safety Assessment Process
System Development Process

Figure 4 ARP 4754A Safety Assessment Process Model (from [180])

The process starts with Aircraft-Level FHA. This step uses information from CCA (Common Cause Analysis), Aircraft Function Development, and System-Level FHA; it also establishes and supplies the derived safety requirements to Aircraft Function Allocation and System-Level FHA. In a similar style, the PSSA (Preliminary System Safety Assessment) and SSA (System Safety Assessment) steps of the safety assessment process use information from corresponding design and other analyses and derive more detailed requirements for a lower level, and provide justification for the achievability of the requirements of a higher level until the complexity of system implementation can be managed. Finally, ASA (Aircraft Safety Assessment) should integrate the analysis results of previous safety activities to demonstrate the overall aircraft/system safety in accordance with the requirements derived during Aircraft-Level FHA and PASA (Preliminary Aircraft Safety Assessment). ARP 4761 [181] presents detailed guidance on the corresponding safety assessment techniques employed at different stages of the ARP 4754A safety assessment process.

2.5.2 Other Safety Processes

There are some other standards and guidance that define a safety process or a safety lifecycle, which specifies the necessary activities involved in the specification, design, installation, operation and maintenance of a safety-critical system. These include MIL-STD-882E [65], IEC 61508 [5], IEC 61511 [6].

The safety processes defined in these standards vary in terms of their applicable domains and other details. MIL-STD-882E is developed especially for US military systems; IEC 61508 is generic for various safety-instrumented systems in a broad sense; whereas IEC 61511 is more specific and aimed at safety-instrumented systems in the process industry. However, the central objectives of these safety processes are the same - the identification and control of risk associated with undesired conditions at different levels of the system hierarchy during a certain time span, which is similar to the foundation of the ARP 4754A safety process.

Although details of specific safety/hazard analysis methods are not the primary concern of these standards, the documentation of hazard analysis methods and techniques are commonly stressed in the requirements associated with these safety processes. However, the assessment of the quality of the process outputs are not sufficiently addressed, which may undermine confidence in the actual achievement of these processes.

Safety assessment activities are, in nature, iterative and interactive [201]. The safety process adopted or customised in the development and operation of a specific system should take into account the existing best practice and the features of the specific system under study.

2.5.3 Process-Related Issues

Safety processes play an important role in system development and certification. With the recommended processes, which are usually verified and summarised from good practice, it is easy and clear for duty-holders to organize and plan activities and resources in the system development lifecycle.

A systematic and rigorous process (usually as recommended by standards and guidance) is one of the aspects on which we base our confidence in achieved system safety because it implies that we have the overall knowledge and capability necessary to control safety issues related to complex systems. Taking account of more lessons from practice and incorporating the latest progress in system development, we can update and improve recommended processes periodically, and can thereby establish a better foundation for confidence in safety. For example, from the safety assessment tasks depicted in [136], to the ARP 4754 safety assessment process [179] and the ARP 4754A safety assessment process [180], the generic safety assessment process adopted in the development of civil aviation systems is more and more systematic and practical. However, safety processes are not, and should not be, the single source of our confidence in system safety.

There are potential issues that need to be considered for in-depth and pragmatic understanding of these processes and for the effective implementation of these processes. Two major issues are discussed below, concerning the imperfection of a purely process-oriented safety culture. In fact, there has been lively critical discussion of the process-based and product-based standard and practice [28, 66, 118, 144].

First of all, the rationale underlying recommended safety processes as best practice are often implicit and exist only as hidden knowledge. Historically, the reasons for doing something in a certain way have been considered or been done implicitly. The underlying reasoning is not typically documented in a systematic form and is not included as a part of the certification document. Without understanding the rationale underpinning interrelationships between the safety activities, the contribution made by various safety analysis results towards achieving the system safety objectives is not sufficiently clear, and rigorous scrutiny and synthesis of the analysis results can be handicapped.

Secondly, the safety process/activities planned and the actual enactment of a planned process are different. The traditional recommended safety assessment process is derived from industrial best practice and has been adopted and practiced for a long time. It allows us to tailor practices as required, and provides us with flexibility over the choice of activities. However, the recommended process or planned activities do not guarantee the quality of results generated from the enactment of a specific process. The safety of a system needs to be justified with outputs from the ‘as-performed process’, not just from the promise of the ‘as-intended process’. This has been identified as a serious concern by some researchers [94, 110].

In summary, processes are not in themselves perfect, and confidence in the eventual system attributes of the delivered system cannot be derived from processes alone. The issues discussed in this section must be considered and properly handled by both regulators and practitioners in order to provide assurance that the systems are designed, maintained and operated in a safe manner. Both safety processes and safety evidence derived from those processes should be emphasized in safety management practice.

2.6 Evaluation of Safety Assessment Models

Safety assessment models are indispensable tools for us to identify, analyze, control, and evaluate system safety in many industries. Evaluation of safety assessment models has been a focus of concern for many years. However, some traditional safety analysis techniques (e.g. FTA) have been criticized for incompleteness and inaccuracy [80, 197]. Besides that, there could be various errors in a safety analysis process [51], e.g. over-simplification during modelling, omission of relevant failure modes, or inadvertent misapplication of data values. So the evaluation of safety assessment models, with regard to the methodology, the modelling process and the modelling results, is very important for a better understanding of the validity and reliability of safety assessment models and their outcomes.

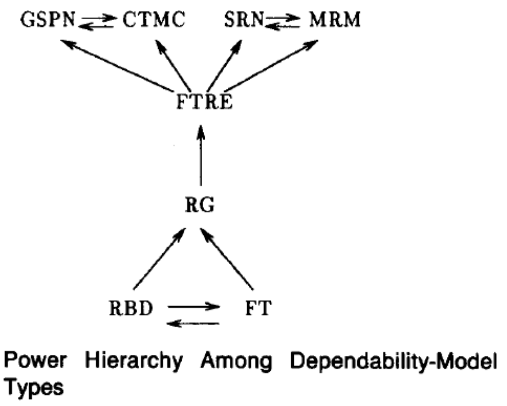
During the evolution of safety assessment techniques, much emphasis has been put on the description and comparison of the strengths and weakness of different modelling methodologies in order to help practitioners to decide which model will fulfil their needs best [14, 35]. There is also some detailed comparison of specific safety assessment models, which has tried to find the inherent transformation or the data correlations between safety assessment models. Section 2.6.1 presents some results from comparisons of the power relationships between different models.

Only a few papers are found on assessing the quality of a single safety assessment method or the validity of the modelling results of a single safety assessment model. Leveson [129] states that we should treat safety analysis results with appropriate scepticism since very little validation has been done on these analysis techniques. Section 2.6.2 outlines the limited scope of the existing work in this area and highlights the difficulties inherent in the assessment of safety assessment models.

2.6.1 Comparison of Safety Assessment Techniques

Rouvroye et. al [173] have compared several popular safety analysis techniques from three perspectives – the information needed for the analysis process, the actions performed during the analysis process, and the results obtained from analysis processes. They give the relation of some quantitative analysis techniques that is ranked according to the modelling power of these techniques and conclude that “the analysis complexity and effort to perform an analysis increases as the modelling power increases” [173].

Malhotra et al have studied the modelling power of different types of some commonly used safety assessment models [139]. They concentrate on the modelling power needed by the kind of dependencies within a system. Eight types of model are compared with the assumption that failure and repair time distributions of system components are distributed exponentially. As Figure 5 (from [139]) shows, some model forms can be transformed into other forms, which means they are equally powerful, such as from a reliability block diagram to a traditional fault tree or vice versa. But not all the conversion between models is bidirectional; some systems could be represented by a more powerful modelling approach, whereas there is no equivalent representation by a less powerful approach. For example, not every fault tree with repair events could be represented by a reliability graph.



GSPN- Generalized stochastic Petri net

CTMC- Continuous time Markov chain

SRN- Stochastic reward net

MRM- Markov reward model

FTRE- Fault tree with repeated events

RG- Reliability graph

RBD- Reliability block diagram

FT- Fault tree

Figure 5 Power Hierarchy among Some Safety Analysis Models (from [139])

From a different point of view, Wilson and McDermid examine the input and output data relations of a set of safety analyses and models [212]. They present an underpinning data model to support integration of those analyses and models. The consistency and completeness of analysis can be checked according to the data propagation.

Research on the comparison of various models gives us a better understanding of the capability of models and insights into how to make decisions on the selection of appropriate models. For large-scale complex systems, different safety assessment models should be used according to their power and the applicable situation and they should be integrated to generate safety assessment conclusions for the whole system. But the value of such research is limited to providing guidance for the application of the techniques, rather than assuring the validity of the combinational usage of these models.

2.6.2 Existing Evaluation of Safety Assessment Models

In terms of modelling approaches, many safety assessment techniques have been discussed in Section 2.4, and their advantages and disadvantages identified. Besides that, there is a body of research questioning and validating different aspects of safety assessment models (the approach itself, the modelling process, or the modelling results), though these articles are not as numerous as the efforts on the improvement and application of safety assessment models.

Some researchers question the methodology of FTA [80, 140, 177]. Manion is one of the significant ones. Manion criticizes the fault tree analysis method from an ethical point of view [140]. He questions the soundness of FTA epistemologically and methodologically in both theory and practice. Some of his major criticism of the weaknesses of FTA includes [140]:

- Incompleteness of system description and vagueness of trade-off principles for setting system analytical boundaries;
- Notable differences in fault trees generated for the same situation which indicate that there are no sound rules for branch construction;
- Foundation on engineering judgments which indicate the method is not a simple deductive reasoning process;
- Uncertain and inadequate data used to construct a fault tree undermine the 'correctness' of the fault tree.

The limitations of FTA discussed in Section 2.4.1 are also evaluated in Manion's paper. In short, he emphasizes that the inadequate understanding of the system and its failure logic and the unjustified assumptions in the FTA method make it far from an objective and scientific way of judging safety. Nevertheless, he has neglected to mention that FTA is not merely used in helping 'judge' safety at the time of certification, but also has a role in helping engineers to communicate understanding of the system and its failure mechanisms and in helping them to decide or select proper safety barriers in system design according to the causal relationships conveyed by the tree. Besides, he does not provide other better means of safety analysis, nor practical guidance or solutions to help safety analysts to change the current situation of unsound analysis.

The analysis results should also be judged by analysts themselves and by others on ethical grounds [140]. It would be helpful if we could retrace and review principles and decisions related to modelling when we evaluate our models. But unfortunately, few safety models are extensively evaluated on the basis of the principles and decisions on which they are built.

There are a very limited number of published materials on the validation of results of safety assessment modelling. Wong and Yeh notice that the model validation step is ‘never mentioned’ or in fact ‘ignored’ in most safety assessment activity due to insufficient relevant data and the purpose of predicting ‘rare’ events [213]. They propose a process for the validation of the fault tree structure which depends on data availability. But the validation of single-event or multi-event occurrences by comparing observational data with the fault tree model could be ineffective due to the possibility that the data in the consistent range simply arose because of a coincidence rather than on the basis of the correct structure of the fault tree model.

Adline [18] identifies two major techniques used for verifying formal safety analysis models. One is Step-by-Step Simulation; another is Model Checking. The simulation technique works on testing scenarios and human knowledge, which is widely used but not exhaustive. The model-checking technique is in fact verifying defined properties of a model: in some sense, checking the model’s conformity with the reality. However, the definition and formalization of the properties to be checked can be difficult. Even if the properties hold in a model, it means the model has desired features but not exactly that the model is a valid representation of the real system behaviour. He proposes a validation process for safety analysis based on dedicated formal models [18]. The process starts with a Specification of Failure propagation Model from informal documents, from which a series of test cases are generated with predefined criteria. After that, the outputs from the test cases running on the implementation of AltaRica Failure Propagation Model (including the nodes, units, and the overall model) are examined through expert judgments. This process is helpful in terms of checking the ‘validity’ of formal safety analysis models. Nevertheless, the method does not ensure the completeness of the specification and test cases and is still heavily dependent on human intervention and expert knowledge.

Suokas has carried out an evaluation of safety and risk analysis in the chemical industry in late 1980s. He proposes four main approaches in the evaluation [197]:

- “complete parallel analysis of the same object;
- parallel analysis on some parts of the same object;
- comparison of the analysis with descriptions of accident(s) occurring in corresponding systems and with personal experience;
- examination of the process behind the analysis”.

The first two approaches evaluate the reliability and coverage of the analysis results for the same object through: a) implementation of analysis by different people or teams and b) comparison with results from other different methods. However, in practice, it is too expensive to have many versions of analysis for one object. The third approach could reveal factors omitted or only occasionally identified from safety analysis. But it is late for feedback and it is inapplicable to systems that have little safety data generated in the operational phase. The fourth approach is indirect and was relatively new in the 1980s, but has been adopted widely in the past decade. The examination of analysis processes, however, is actually a means of establishing confidence in the degree of truth carried by the analysis results, rather than a means of demonstrating the validity of the analyses (as we have explained in Section 2.5.3).

As Shannon suggested, and we outlined in Section 2.3.2, there are many ways that we can perform model validation. However, except for those approaches that rely upon human knowledge and in-modelling checks, e.g. ‘using common sense and logic’ or ‘checking and rechecking each step of the model building process’, other means of validation suggested are almost inapplicable to the evaluation of safety assessment models. For example, the drawback of empirical tests or field tests is that the approach relies heavily upon an ability to test or to simulate the real world for safety issues and a capability of measuring the real-world data. Most of the time, we don’t have the opportunity to compare models with data from the real world. Given that a safety assessment model is often used in a predictive fashion, dealing with hypothesized system behaviours that we think the system ought or not to exhibit, the value of the model is to provide us with thoughts about and a view of the problem domain in the future tense, rather than in the past tense. This poses great challenges to the ‘validation’ of safety assessment models. In fact, we can only pursue better confidence in the quality of safety assessment models.

2.7 Safety and Assurance Cases

As “a reasoned attempt to justify a conclusion” [85], an argument can be used as a form of inquiry and communication for persuasion. For the demonstration of system safety with safety assessment models, it is beneficial to incorporate argument-based approaches to bolster our claims on system safety. Similarly, for the demonstration of our understanding of various safety assessment models, argumentation is also helpful to provide support to our claims of model adequacy. This section firstly introduces general argumentation concepts that underpin the development and evaluation of safety cases. Then a review is provided on the development of the safety case domain and of recent research work that reflects interest in the confidence issue in system assurance and the trend of a unified structure of arguments and evidence in assurance cases.

2.7.1 Argumentation

"An argument is a set of claims that a person puts forward in an attempt to show that some further claim is rationally acceptable" [85]. The most basic components in arguments are claims (expressed as propositions). The claims associated with evidence or reasons are the *premises* of an argument; and the claims on the view being defended is the *conclusion* of an argument [85]. The claims are interconnected and the conclusion is inferred if all relevant premises are satisfied. Toulmin identifies six elements in the structure of arguments – claims, data or grounds, warrants, backing, qualifiers, and rebuttals [198]. His work provides a good foundation for the analysis and construction of various kinds of arguments, regardless of the domain in which they are applied.

An argument is a form of communication used for persuasion by citing evidence or reasons to backup our claims. Trudy states that an argument is "to solve controversy in a natural, non-violent way" [85]. Fox argues that argumentation has far greater representational power than traditional mathematical formalism based on probability or other quantitative concepts, and it is more versatile and robust under conditions of lack of knowledge [82].

The process of putting forward an argument, including the whole activity of "making claims, challenging them, backing them up by producing reasons, criticising those reasons, rebutting those criticisms" [199], is argumentation.

General criteria of a good argument are identified in the 'ARG' conditions presented in [85]. These general criteria also demonstrate that the quality of the backing provided for a claim is crucial for a compelling argument. 'A' means acceptable – the data or evidence used in the argument must be accepted as a viable starting basis for an argument; 'R' means relevance – the data or evidence must be genuinely relevant to the concluding claim; 'G' means grounds – the data must provide sufficient grounds for the conclusion. The last two conditions deal with the proper connection from the data to the conclusion. Besides the logical aspects covered by the general criteria, arguments can also be evaluated according to another two criteria, dialectical and rhetorical criteria, for the effectiveness of their rational persuasion [55]. The dialectical criterion helps to investigate whether the objections and questions on arguments have been answered satisfactorily. The rhetorical criterion may take into account aesthetic consideration in the evaluation, which "examines an argument's effects on the audience" [55].

Graphical representation of argument and evidence was initiated by Toulmin [198] and Wigimore [208]. Toulmin's schematic concentrates on the layout of an argument, whereas Wigimore's chart focuses on the variety of evidence and the interrelated evidential relationships. The visualised representation of arguments has gained popularity in recent decades [123, 203] with tool support in different disciplines, such as education, philosophy, and artificial intelligence. Graphical notations for the presentation of structured safety arguments are useful for facilitating safety case development and reviews, due to the structure and clarity offered by the graphical views.

2.7.2 Safety Case Overview

The concept of presenting safety-related information and arguments in a formal report initially came from the nuclear industry, but the notion of 'safety cases' originated in major industrial accident control regulations introduced in the process sector in the UK in 1984 [4]. Lord Cullen, in his report on the Piper Alpha accident [58] in 1990, recommended the introduction of a safety case regime as part of the regulation of oil and gas facilities and operation. The philosophy of a safety case is to construct a clear, structured, compelling argument to demonstrate the safety of a system in a particular operational context.

The definition of a Safety Case from UK Defence Standard 00-56 [149] is:

> *"A structured argument, supported by a body of evidence that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given operating environment"*.

The core of a 'safety case' is the safety argument. A safety argument communicates how the overall objectives and claims of the safety case can be shown to be supported by the available evidence (such as the safety analysis results). A safety argument is generally composed of a hierarchy of safety claims and evidence, together with the inferential steps that are believed to connect the claims to the evidence. The explicit presentation of a safety argument is intended to encourage rigorous thinking and questioning which is potentially of great value when demonstrating the outputs of novel products and novel methods [56, 154].

Safety case development, review and acceptance has been adopted and practiced systematically in a wide number of industries, especially in Europe, for more than twenty years. Industries adopting the safety case method include railway, air traffic control, maritime, and defence. The mandatory requirements for safety cases in some industries (e.g. DS 00-56, EUROCONTROL's Safety Assessment Methodology) show that the importance of the role of safety arguments is acknowledged. The recent releases of international standards such as the ISO/IEC15026 (Part 1 and Part 2) [8, 9], ISO26262 [7], and a FDA guidance [74] also indicate increasing adoption and interest in the application of an argument-based approach for system assurance. The central theme of using arguments for justification of the achievement of system attributes has now been transferred and expanded beyond the area of safety engineering. There are security cases, reliability cases, dependability cases, trust cases, survivability cases, and assurance cases. Some aviation-related systems have adopted this approach for system safety assurance, for example, integrated modular avionics [109], air traffic control [48, 75] and aircraft operational hazard control [71].

In addition, there are three recent standardisation efforts by the Object Management Group (OMG) System Assurance Task Force [159], which aim to enforce knowledge exchange and integration of assurance cases. They are: the standardised Argumentation Metamodel (ARM) [156]; the Software Assurance Evidence Metamodel (SAEM) [158]; the Structured Assurance Case Metamodel (SACM) [159].

2.7.3 Structured Argument

To be ‘clear’ and ‘comprehensible’ is the fundamental requirement of a safety case. The way in which an argument is represented and organized will largely determine the effectiveness of communication between safety case providers and safety case reviewers. Despite the variety of approaches for argument representation, visualized arguments are a common method of presentation.

Currently the most commonly used graphical notations for safety arguments are the Goal Structuring Notation (GSN) [89, 117] and the Claims, Arguments and Evidence notation (CAE) [17]. For a comprehensive description of all GSN symbols, the reader is referred to the GSN Community Standard [89]. The construction and management of safety cases is also supported by commercial software tools, such as CertWare Safety Case Workbench by NASA [29], ASCE [16] by Adelard and GSN Modeler [24] by Atego. An example safety argument represented in GSN from [113] is shown in Figure 6. In the example, the top-level safety goal is ‘*Control System Logic contribution to system level hazards is sufficiently managed*’. The top safety goal is supported by lower-level sub-goals indirectly through two argument strategies. At the lowest level, the sub-goals need not be further decomposed and can be clearly supported by reference to items of safety evidence, such as a system analysis model or the results of system testing.

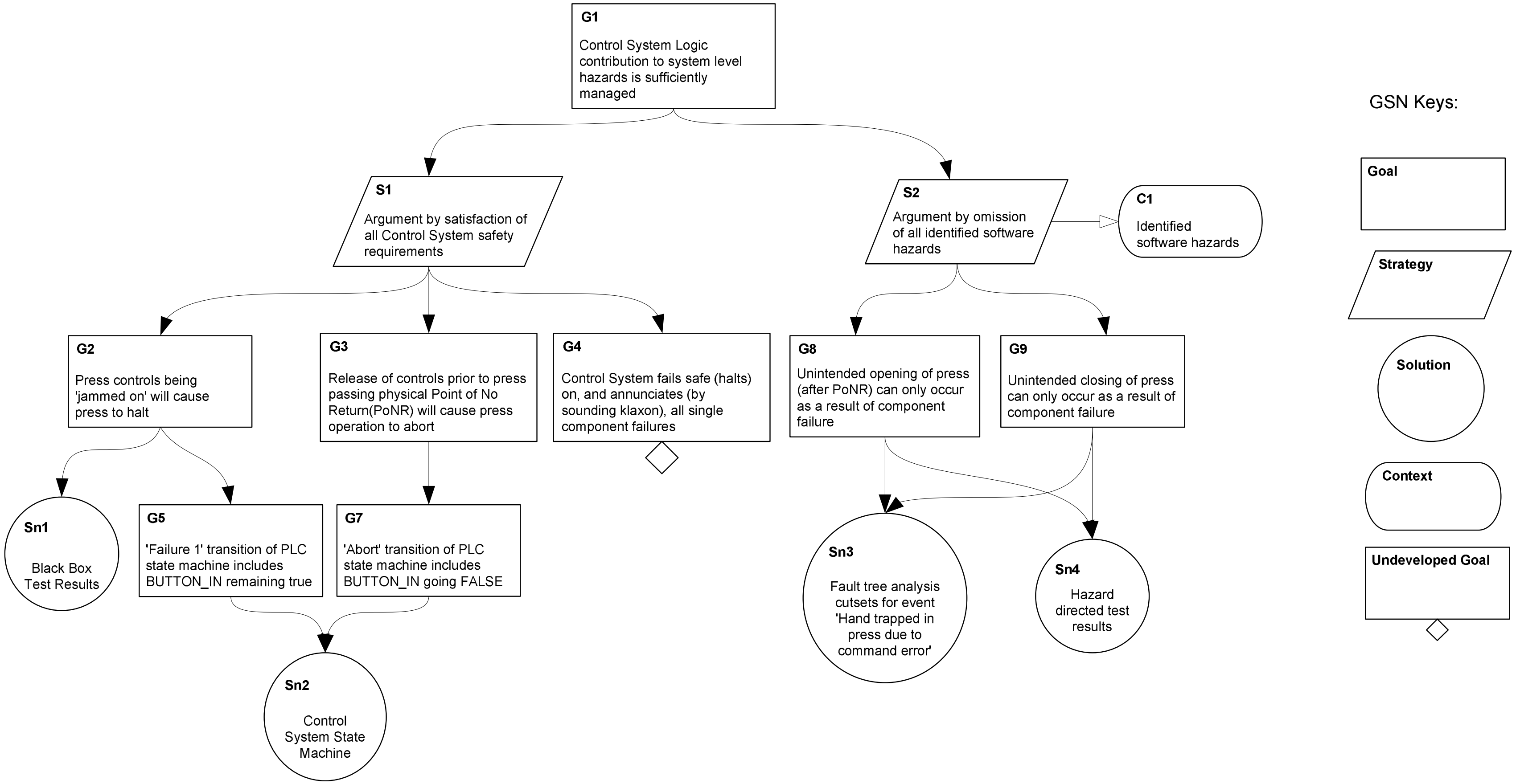

G1
Control System Logic contribution to system level hazards is sufficiently managed
S1
Argument by satisfaction of all Control System safety requirements
S2
Argument by omission of all identified software hazards
C1
Identified software hazards
G2
Press controls being 'jammed on' will cause press to halt
G3
Release of controls prior to press passing physical Point of No Return(PoNR) will cause press operation to abort
G4
Control System fails safe (halts) on, and annunciates (by sounding klaxon), all single component failures
G8
Unintended opening of press (after PoNR) can only occur as a result of component failure
G9
Unintended closing of press can only occur as a result of component failure
Sn1
Black Box Test Results
G5
'Failure 1' transition of PLC state machine includes BUTTON_IN remaining true
G7
'Abort' transition of PLC state machine includes BUTTON_IN going FALSE
Sn2
Control System State Machine
Sn3
Fault tree analysis cutsets for event 'Hand trapped in press due to command error'
Sn4
Hazard directed test results
GSN Keys:
Goal
Strategy
Solution
Context
Undeveloped Goal

Figure 6 An Example Argument Presented in GSN (adapted from [113])

Due to the complexity of the safety arguments required for non-trivial systems, the organization of separate pieces of safety arguments is in demand for a better overview of the whole argument and a manageable argument construction process. The concept of a compositional safety case is proposed by Kelly in [119]. Safety case modules, safety case architecture, safety case contracts are introduced to decompose a potentially complicated safety case into small blocks, but at the same time to keep a cohesive character for each block and to maintain clear and informative interfaces with other blocks as far as possible.

ARM [156], the argumentation meta-model, is intended to facilitate the communication of structured arguments between projects and the exchange of argument information between different tools. It provides the common structure and argumentation concept framework underlying several existing notations for assurance cases. The overview of ARM is shown in Figure 7.

There is an atomic component of argument represented by *ModelElement*. Two key subtypes of *ModelElement* are *ArgumentElement* and *ArgumentLink*. An *ArgumentLink* connects *ArgumentElement* or other sub-types of *ModelElement*. A branch of argument can be packaged up as an ‘Argument’ and used as a *ModelElement* in another argument. *Claim*s are the essential elements in an argument; they are the assertions with a True/False value stating our position or desired achievements. *ArgumentLinks* can be characterised as different sub-types according to the types of *ModelElements* being connected. For example, a *Claim* can be linked with another *Claim* via a link of *AssertedInference*; an *InformationElement* can be linked with a *Claim* via a link of *AssertedEvidence*.

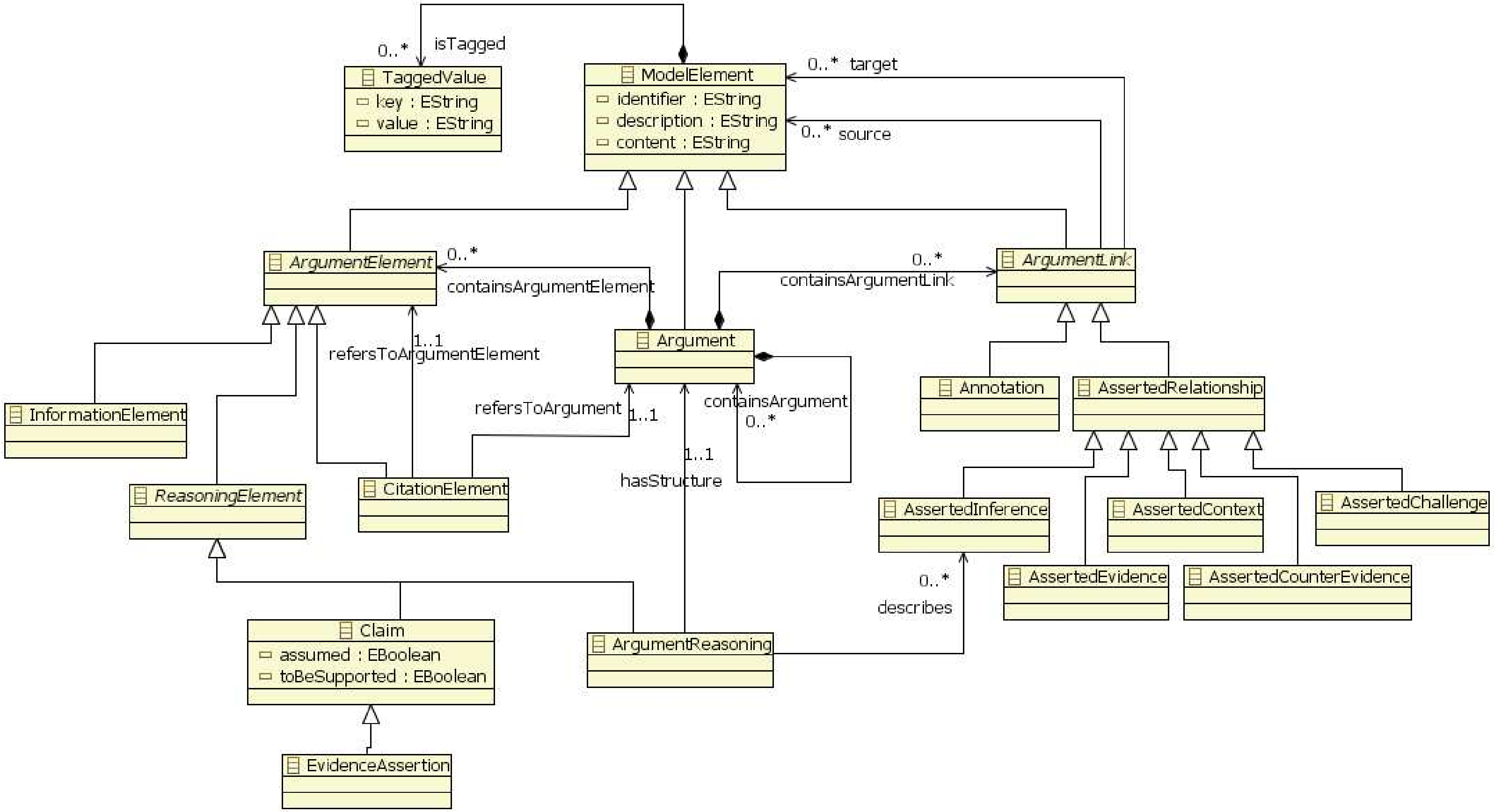


Figure 7 Argumentation Meta-Model (From [156])

2.7.4 Safety Evidence

A body of evidence is the grounds for the acceptance of a safety argument as compelling. DS 00-56 explicitly and particularly stresses the importance of safety evidence with meaningful and instructive requirements on evidence, which are broadly suitable for the safety assurance of various systems. In DS 00-56 Part 1 [149], the requirements on the provision of evidence include:

> *"9.5 The Safety Case shall contain a structured argument demonstrating that the evidence contained therein is sufficient to show that the system is safe. The argument shall be commensurate with the potential risk posed by the system, the complexity of the system and the unfamiliarity of the circumstances involved".* [149]

> *"11.3.1 The quantity and quality of the evidence shall be commensurate with the potential risk posed by the system, the complexity of the system and the unfamiliarity of the circumstances involved".* [149]

> *"11.3.2 The Contractor shall provide diverse evidence that safety requirements have been met, such that the overall safety argument is not compromised by errors or uncertainties in individual pieces of evidence".*[149]

In addition, counter-evidence and the rigour of the scrutiny of evidence and the evidence generation process are also further addressed in DS 00-56 Part 2 [148].

> *"9.5.6 Evidence that is discovered with the potential to undermine a previously accepted argument is referred to as* ***counter-evidence****. The process of searching for potential counter-evidence as well as the processes of recording, analysing and acting upon counter-evidence are an important part of a robust Safety Management System and should be documented in the Safety Case".*[148]

> *"17.4.3 The counter-evidence should be documented, analysed and referenced by relevant safety claims".*[148]

Similarly, safety experts [92] and regulators [73] are also concerned about the desired features of evidence, but there is no commonly accepted criteria for evidence yet in reality. Hamilton [92] identifies two types of evidence properties: objective properties and subjective properties. The first type, such as existence, completeness, correctness can, according to the author, be objectively audited. The second type, such as relevance, sufficiency, contextual validity in context, is judgment-dependent and should be evaluated in the context of a safety argument. However, most of the objective properties suggested cannot be 'objectively' evaluated and the suitability and means of evaluation of these properties are dependent on the nature of the evidence presented. Some researchers recommend managing safety evidence with qualitative requirements on the types of evidence required [36] or assessing the strength of evidence [48, 77] that we need of the evidence submitted for assuring the correct implementation of system safety requirements. Two typical examples are AELs (Assurance Evidence Levels) [48] and SEALs (Safety Evidence Assurance Levels) [77]. The levels in [48, 77] are a qualitative tag that defines the breadth and strength of evidence, or even the specific types of analysis or forms of artefacts required. Grouping evidence with such requirements is informative for planning safety activities and resources. But users still need to interpret and select an appropriate evidence assurance level according to the features of their products or service to be assured.

Another theme in the literature focuses on clarifying the level of assurance that the evidence can provide through evidence justification. In [96], a structured software evidence selection and justification process is proposed for evaluating software evidence in the context of arguments. ESALs (Evidence Safety Assurance Levels) are suggested in [170] as a means of setting requirements for the level of rigour of the justification of the soundness of evidence in the context of safety arguments.

Some existing research work [164, 216] concerns the extraction of items of required evidence and their relations with system design and recommended processes from standards in order to help practitioners to plan their safety activities and deliverables for certification, such as the evidence meta-models specifically based on IEC61508 [164]. They are primarily process-oriented and constrained by the forms of evidence required or recommended by specific standards and guidance. The role and nature of safety evidence, therefore, is not explored and elicited in these models.

The software assurance evidence meta-model (SAEM) [158] by the OMG, is used to describe the common vocabulary, features and attributes of evidence used in an assurance case. Figure 8 illustrates the core logical parts of assurance evidence. The logical parts of evidence, as presented in SAEM, spread from *Exhibits*, *Fact Model*, *Properties*, *Evidence Evaluation*, and *Administration*.

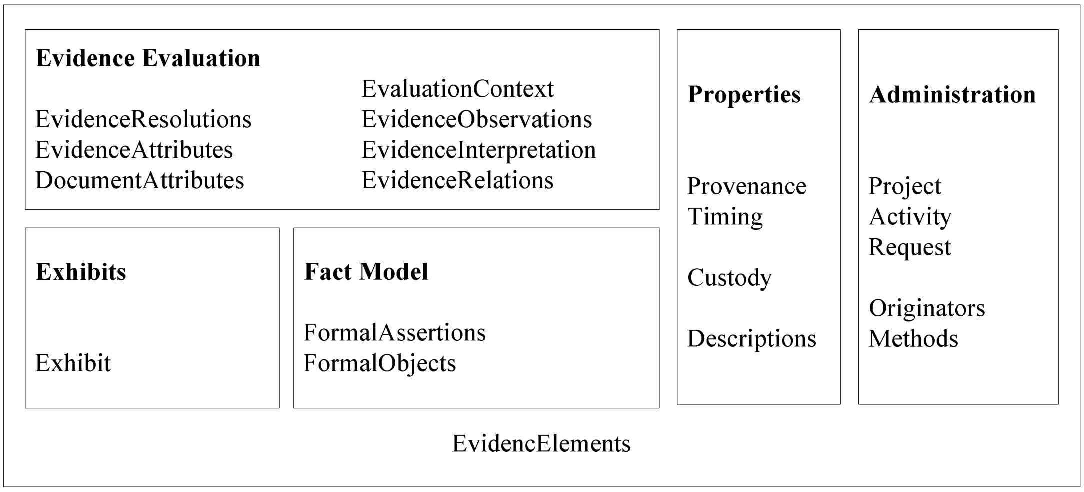


Figure 8 Key Parts of Software Assurance Evidence Meta-Model (from [158])

SAEM separates the meaning of evidence from the presentation or exhibition of evidence. It also integrates the evidence lifecycle data, the quality of evidence and the argumentation role of evidence into the meta-model. However, SAEM is complex, due to its motivation of supporting automation in the generation, exchange and management of assurance evidence. There is no clear interface between SAEM and ARM. Some overlapping data elements in ARM and SAEM should be rearranged to smooth the integration of evidence and argument within a coherent assurance case (or safety case). For example, in the fact model of SAEM, there is an assertion class which presents an argument's content. The definitions of *Assertion* in the two meta-models are not the same, which might cause confusion in the establishment of assurance and/or in information exchange. The reconciliation and interpretation of interface elements between SAEM and ARM is an interesting and challenging task.

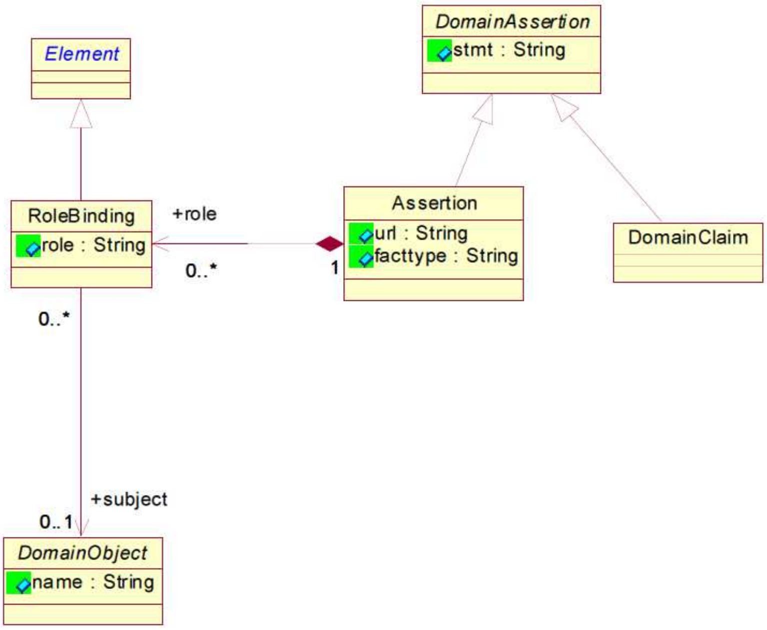


Figure 9 Formal Assertion Class in SAEM (from [158])

For example, in Figure 9, *DomainAssertion* has a property *stmt* which documents the statement of facts in natural language. *Assertion* in SAEM is a formally-represented relationship between domain objects through a reference to external vocabularies; whereas, in ARM, propositions in structured arguments are abstracted as *Claim*, which can be either formally or informally stated.

2.7.5 Confidence Associated with Safety Cases

Besides the importance of evidence, issues connected with uncertainty in safety arguments have also been addressed by some recent studies. Weaver examines the key sources of uncertainty which potentially affect the acceptance of safety cases [205]. Cyra and Gorski introduce a way of evaluating the strength of arguments as a whole in [59] by adopting Dempster-Shafer's evidence theory to deal with the uncertainty originated from the lack of system knowledge. However, the requirement on independence between evidence for D-S theory will bottleneck the use of Cyra and Gorski's method. Furthermore, to assign quantified data to argument elements is not a trivial task for complex systems.

The use of multiple diverse arguments has been proposed as a technique to limit and control uncertainty about claims in arguments. Bloomfield et al [40] study how diverse argument 'legs' work and how they benefit the degree of confidence we can have in safety claims. Littlewood et al in [134] evaluate increased confidence in safety claims based on a Bayesian belief network that reveals subtleties in interactions in arguments that might not be readily obvious. Similarly, Denney et al [63] identify and quantify the uncertainty in a safety argument and employ Bayesian networks to reason probabilistically about confidence.

In recent work [97], Hawkins et al propose the 'Assured Safety Argument' approach for explicit differentiation of the primary safety argument[3] and the confidence argument about the primary safety argument. The confidence argument part is also addressed as '*meta-case*' [39] or '*meta-argument*' [22, 204] linked with the main safety argument. The two types of arguments in an Assured Safety Argument are depicted as [97]:

- "A safety argument that documents the arguments and evidence used to establish direct claims of system safety".
- "A confidence argument that justifies the sufficiency of confidence in this safety argument".

[3] We use 'primary safety argument' to refer to the 'safety argument' depicted in [97] to avoid the confusion with the traditional use of 'safety argument' that covers both the primary safety part and the confidence part.

In this way, the core safety arguments related directly with domain safety objectives are separated from the argument and information that are adopted for establishing confidence in the core safety argument. *Assurance Claim Points* (ACPs), which can be represented using graphical identifiers in GSN diagrams, are introduced for three different types of asserted links in the primary safety argument to address aspects of the associated confidence argument. An example of the use of ACPs is presented in Figure 10.

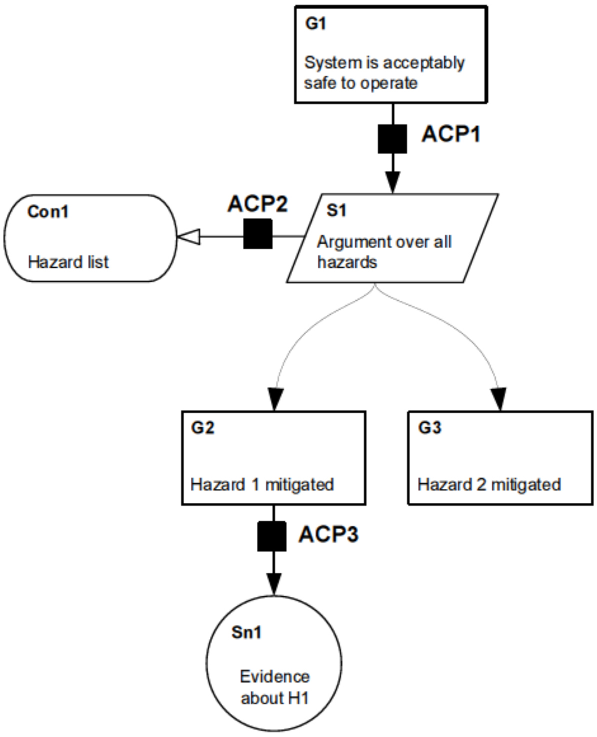


Figure 10 An Example Use of ACPs (from [97])

The confidence argument presented in [97] provides a way of establishing confidence qualitatively. The scope of ACPs to be considered should be carefully determined according to the level of risk associated, especially for large-scale safety arguments where it would be impractical to consider *all* ACPs.

Goodenough et al [83] introduce a framework of confidence with three types of defeater in argumentation – rebutting defeaters, undercutting defeaters, and undermining defeaters. They state that a confidence argument should be constructed on the basis of defeaters and confidence in a claim can be quantified with a Baconian Probability, which is an ordinal number that depicts the relationship between the number of defeaters against a claim and the number of eliminated defeaters identified. However, the values of Baconian Probability of different claims are not comparable and the approach assumes identified defeaters as a sufficient boundary of the confidence issues associated with a claim.

2.7.6 Models and Safety Cases

Safety assessment models play a central role in the construction of a safety case [45]. Although we agree that information on models is useful for constructing a safety case, it is said that "there is little guidance available as to how to incorporate information from fault models into the safety case" [138]. Alexander et al point out that different types of models can provide different types of evidence at different effectiveness levels [21]. But they do not analyze how a specific model supports particular kinds of safety claim. Lutz and Patterson-Hine formulate three questions to investigate how evidence from tool-supported fault modelling and analysis activities can be used to construct safety cases [138].

1. *"How readily can the fault models be used as evidence for a safety case"?*

2. *"How can we use or adjust the modelling process to ease the construction of safety-case arguments"?*

3. *"How well do steps to support construction of the safety case align with the developer's interests"?*

Their exploration in this area is based on an Advanced Diagnostic and Prognostics Testbed (ADAPT) under development at NASA Ames. The ADAPT is not safety critical, but the investigations have shown that some evidence in a safety case can be provided by system fault models. From their experience, the ADAPT fault model can provide the following information as evidence for a safety case [138]:

- "Evidence from the contingency analysis (e.g. the hazard identification)"

- "Evidence from review of the fault model"

- "Evidence from tool-supported static analysis of the fault model"

- "Evidence from running the model".

They not only adopt the fault model as evidence, but have justified the use of the fault model through extensive review by domain experts, by comparison of simulation of scenarios in the model with actual system outputs, and by extensive exercising of the model. However, they treat the justification of the fault model as evidence in the same way as the modelling results are used, but do not distinguish it as the backing of evidence. It is necessary to treat the backing arguments separately according to Toulmin's structure.

Both system models and safety assessment models typically change and evolve during the system development and implementation cycles. As well as the updating of models, the primary and backing arguments over these models need to be updated too. There are no studies on how to support the construction and management of safety arguments over several generations of safety models.

On the whole, the importance of safety arguments concerning models has been recognized by practitioners [45] and required by regulators [149], but only limited research has been undertaken on the interrelationship between models and safety arguments. More research efforts should be established in this area, since the quality of these models is crucial for the quality and success of the whole safety case. We need to argue for safety with safety-related models as items of evidence, and to have a reasoned explanation of why a model can be considered ‘fit for purpose’.

2.8 Summary

This chapter presents a review of the existing literature related to model validity, safety assessment and justification. It is clear from the review that the validity of safety assessment models is important; however, it is also clear that this validity is difficult to examine and to justify. There is also a lack of established practice and guidance on how to assess safety assessment models as evidence in safety cases, except for informal review by human experts on the basis of their domain knowledge and experience.

In reality, incomplete, flawed, unjustified or incompatible evidence may potentially lead us to a false conclusion as to the level of system safety achieved, which would be extremely dangerous for decision-making and for safety management. Therefore, especially in the light of the challenges posed by the number, scale, variety and complexity of safety assessment models in modern system development, we are obliged to explore potential problems and solutions regarding confidence establishment in using safety assessment models as evidence in safety cases. In the following chapters, we begin to develop a structured view of safety assessment evidence in Chapter 3 and Chapter 4; then we focus on model inconsistency analysis as a rigorous approach for the evaluation of safety assessment models in Chapter 5; and finally we integrate the use and justification of safety assessment evidence more systematically within structured safety case development processes in Chapter 6.

3 Safety Assessment Meta-Modelling

3.1 Introduction

In the safety domain, there are many types of safety assessment modelling techniques which vary from each other in terms of complexity, power, inputs, and assumptions. In addition, the safety assessment process within a product lifecycle encompasses miscellaneous interrelated safety assessment activities with multiple forms of modelling outputs. In order to evaluate these safety assessment modelling outputs comprehensively and to adopt them appropriately as evidence in safety cases, we must have a clear and comprehensive understanding of both the information elements contained in the safety assessment models and the domain factors involved in the safety assessment modelling processes used to produce them.

However, existing guidance materials on safety assessment usually place considerable emphasis on safety assessment techniques and prescribed system safety processes for implementation purposes. They are generally not concerned, however, to introduce a specific view of safety assessment modelling processes from the perspective of factors that may influence the content and validity of safety assessment models. Moreover, existing models and meta-models for safety assessment are so diverse in terms of the modelling purposes and modelling constructs they employ that it is difficult to understand the relationships between model instances. Additionally, existing meta-models detailing the data elements in safety assessment models have not taken in to account all the data elements needed for evaluating the quality of these models. Some important but implicit aspects of models have been left out, such as the scope of a model or the assumptions made during modelling. With insufficient understanding of the data and the process of safety assessment modelling, it is difficult to have confidence in the safety assessment models that are used as evidence in safety cases.

Safety assessment models are a type of common source data of safety evidence adopted in safety cases. In this thesis, the structured view of safety assessment evidence is divided into two parts: the view of safety assessment models (explored in this chapter) and the view of safety evidence (explored in Chapter 4). This chapter addresses the shortcomings of current guidance and models for safety assessment by defining a domain model and a core data meta-model for safety assessment modelling. We first analyse different factors involved in the overall safety assessment modelling domain and the interrelationships between these factors. After that, a safety assessment core data meta-model (CoreDMM), which contains core information in major qualitative safety assessment artefacts, is presented in order to support the systematic justification of models in terms of their suitability of being addressed as items of evidence in safety cases.

In terms of the approach for illustration of the concepts and the relationships related to safety assessment, the technique of meta-modelling from the software engineering domain is adopted here, because it is well suited to expressing and organising language and information structures at different abstraction levels, from the conceptual components of a method to concrete safety assessment data.

3.2 Meta-Modelling

A model is "a representation of something else, designed for a specific purpose" [47]. The prefix 'meta' means something abstract or of a higher kind. A meta-model is thus "an explicit model of the constructs and rules needed to build specific models within a domain of interest" [167]. Meta-models tackle the problem of complexity with abstraction at the level of a problem domain rather than abstraction at the level of computing solution space.

A model or a meta-model is not absolute. A meta-model is a model itself and can be represented by another meta-model, which might be called as a meta-meta-model. The traditional meta-modelling infrastructure defined by the OMG consists of multi-level models with a four-layer structure [25], M3 Meta-Object Facility (MOF), M2 UML concepts, M1 User concepts, M0 User data. The left-hand side of Figure 11 depicts the four layers of the OMG meta-modelling infrastructure.

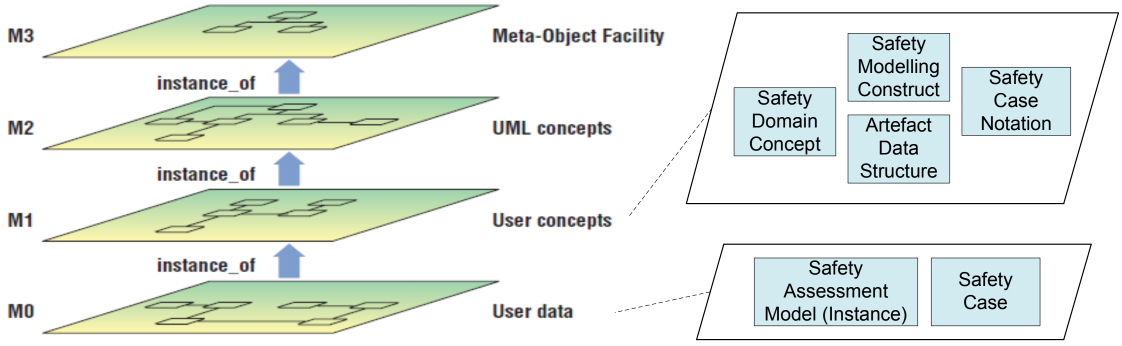


Figure 11 OMG Meta-Modelling Infrastructure (from [25]) and Safety Meta-Models

However, the OMG layered-model presents models primarily as instances of another model (a meta-model), from the perspective of language abstraction. A meta-model can also be viewed from another two perspectives [167]:

- as a set of building blocks and rules used to build models and
- as a model of a domain of interest.

It is clear that the safety models and meta-models (depicted in Figure 11) are not presented at the same abstraction levels if we view them from these two perspectives. Atkins and Kuhne [25] identify the need for two separate and orthogonal dimensions of meta-modelling: the linguistic instantiation and the ontological instantiation. From the ontological perspective, safety assessment meta-modelling addresses concepts and modelling constructs at varied abstraction levels. Considering the linguistic dimension, we only address two levels in the OMG meta-modelling structure in our domain-specific use of this technique in system safety. The M1 level is used for organising elements of safety modelling languages, concepts and the relations between these elements. As illustrated in Figure 11, safety domain concepts (e.g. hazard, accident), safety modelling constructs (e.g. failure mode in FMEA, basic event in FTA) that are associated with safety analysis techniques, safety artefact data structures (e.g. the data column in an FMEA worksheet), safety case notations (e.g. Goal or Context in GSN) are deemed as models at the M1 level. These M1 level models, in our application scenario, are all described in UML that is at the M2 level. The M0 level is used for hosting objects/instances of those M1 models. As shown in Figure 11, models at the M0 level may be concrete safety analysis results that represent the safety features (e.g. failure behaviour) of a concrete system from a certain viewpoint or a safety case for that concrete system. The models at the M1 level serve as the meta-model of the models at the M0 level.

As a higher-level abstraction of models, meta-modelling is adopted in safety assessment, model assessment, and safety cases (even they are presented at the same M1 level) in order:

- to provide a structured description and definition of the core concepts in system safety and the modelling constructs that exist in safety analysis techniques
- to serve as a bridge that provides assistance in the comprehension and comparison of data in safety assessment instances
- to structure properties of safety evidence and its interface with argumentation.

In recent years, meta-modelling has been gaining increased popularity in the field of system safety. A number of existing safety-related meta-models at varied abstraction levels have been developed, and there has been general progress in rigorous software engineering techniques and the increasing integration of safety and system modelling activities. The central concepts in these meta-models include safety [46, 79, 90, 141, 145, 212], dependability [64], reliability [122], and risk [186], depending on the purpose and the scope of the meta-models. The existing safety meta-models that we have identified generally have one or more of the following primary roles:

- as a model of the domain concepts and their relationships, such as the SEI safety information model [79].

- as a model describing the modelling constructs of safety analysis techniques, e.g. those presented by Briones and Miguel [46, 145], those presented by Mason [141]. A large group of safety meta-models are of this type, due to the diversity of safety analysis methods and the varied implementation of these techniques by domain users.

- as a model of safety process which is composed of various analysis and assessment activities, e.g. Habli's meta-model [90]. Only a few models cover this aspect since there are many workflow meta-models and business process meta-models that are capable of describing the safety analysis process.

- as a model of the output of safety analysis, e.g. the one presented by Wilson et al [212]. Few models are designed specifically from this perspective. This approach is quite useful for the bridging of domain concerns and analysis data types covered by specific safety analysis techniques.

However, the role of safety assessment models as evidence in safety cases is not considered in these existing meta-models. To ease the communication of the understanding of models in the assessment of models and their integration with safety arguments, the two structured models presented in this chapter address two distinct aspects: the contextual factors in safety assessment modelling for evaluating models and the data elements associated with the evidential role of models in safety cases.

3.3 Safety Assessment Modelling

3.3.1 Domain Context of Safety Assessment Modelling

To have a better understanding of what has happened during the construction process of safety assessment models, a collection of 'elements' or 'concepts' need to be identified and the relationships between these concepts described and analyzed. A domain model[4] that depicts the major concepts and their interrelationships in system safety modelling is presented in Figure 12. It describes the factors involved in and associated with common safety assessment modelling activities. It serves as the context of safety assessment modelling activities. As we have explained in Section 3.2, this domain model is a model at the M1 level that can be viewed (and addressed) as a meta-model for models at the M0 level. This meta-model is developed on the basis of the models/meta-models proposed in [30, 79]. The key elements identified by this model are the safety assessment model and its modelling context - the target system, the safety concern, the modelling method, the modelling tool, the modeller and the safety modelling process.

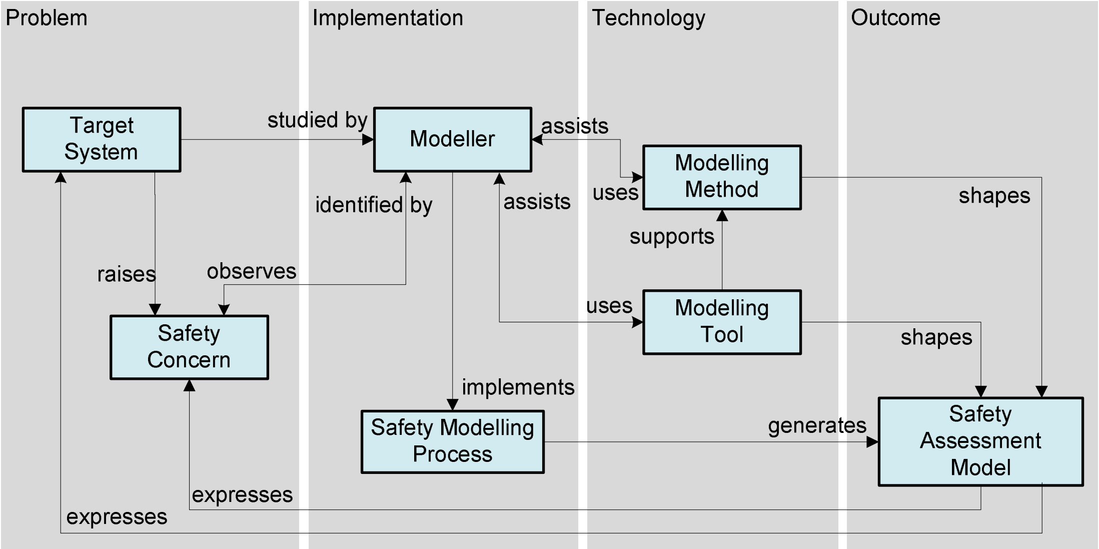


Figure 12 A Domain Model of Safety Assessment Modelling

In Figure 12, the overall safety assessment modelling domain is divided into four sectors – the problem sector, the technology sector, the implementation sector and the outcome sector. The problem sector and the technology sector supply information, knowledge, and support to the implementation sector, through which modellers carry out the modelling process and

[4] The term 'domain model' in this chapter means 'a model of the domain of modelling'. It is more abstract than the same term used in model-driven engineering and in software engineering, where a 'domain model' usually refers to the representation of concepts and relationships associated with the problems, knowledge and/or requirements of a system, service or business.

generate safety assessment models as modelling outputs. Safety assessment models, in the outcome sector, represent the safety concerns regarding the specific target system. The modeller, in the implementation sector, plays an essential and active role in safety modelling, by communicating and synthesizing messages from various sources and abstraction levels.

The elements in the domain model are described in the following subsections.

Target System

The target system is the system to be modelled. The system that we are interested in can evolve from an intended system with an oral description, to a designed system with engineering drawings, or to a real system in operation. The intended system defined at the very early stage of a system life cycle always has 'function' as its primary attribute and has 'safety' as a secondary attribute, if safety attributes have indeed taken shape at that time. We cannot start the system safety modelling before we have at least a conceptual model of the system (to represent how the intended system will work). The safety modelling of a real system in operation largely involves modelling on the basis of observational operation data or accident data. The focus of this study lies in the target system to be modelled at the system development stage, which we named as a 'designed system'. However, we must be conscious of the distinction between the intended system, the designed system and the real system we finally have. They are three different systems, concerning each of which we have some but incomplete knowledge. Differences exist between each grouping of any two of the three possible target systems. But in safety analysis practice, modellers sometimes mix them unintentionally.

Safety Concern

Safety is a property of the target system being modelled. Broadly speaking, safety concerns are factors or situations that we do not want to arise. Their existence or occurrence can lead to a negative impact on system functions or missions or even to injuries and loss of human life. Some safety concerns are expressed as concrete initiating hazardous factors related to the target system or its operational context, such as component failures or unintended/unsuitable discharge of poisonous materials; whereas some other safety concerns are expressed as overall unsafe situations involving the target system, such as mission failures or loss of properties. During safety analysis, we need to decompose higher level safety concerns into more concrete and specific ones or to trace them to their causes or effects in order to set control measures. In safety assessment, we are extending the safety analysis to check the achieved level of safety of the envisioned target system. The collection of safety assessment model instances that are output from safety assessment processes systematically addresses various levels of safety concerns relating to the target-level system from different perspectives.

Modelling Method

As we have seen in Chapter 2, there is a variety of safety modelling approaches that can be adopted at system development stages, depending on the purpose of the safety assessment and the nature of the system to be modelled. Each modelling approach has its own particular limitations and strengths. Modellers need to have sufficient knowledge of a method prior to a specific modelling activity. The nature of the methods adopted will shape the content of the safety assessment results. Various modelling approaches have relationships between them. Some approaches have overlapping modelling intents. For example, both FTA and Markov analysis can be used for predicting the probability of a system failure. Some approaches have shared modelling constructs, e.g. a failure mode in FMEA may appear as a basic event in FTA. Most of these modelling methods - qualitative or quantitative - are supported by software tools in terms of data management, computational analysis or integrity checks.

Modelling Tool

Tools are usually software products which can support the modelling process in terms of graphical representation, computational processing, documentation and data retrieval. With a tool, large scale models can be more easily and efficiently manipulated. However, different safety tool environments can vary a lot in implementation details, data format and accessibility, even if they are serving the same modelling technique. The safety assessment models, if generated with a tool support, cannot normally be directly manipulated in another tool. The capability and limitation of a tool should be considered during the evaluation of safety assessment results. Different tools could have different data formats of safety assessment models based on the same modelling technique.

Safety Modelling Process

The process of safety modelling is characterized by modellers' gradual description of their increasing understanding of the target system in the language of a specific safety assessment modelling approach. During this process, the modellers need to make a number of decisions and/or assumptions concerning the problem being modelled, in order to constrain the scope of the safety assessment model and to make the representation in the safety modelling feasible and manageable.

Modeller

The people who construct safety models play an essential role in the process of safety modelling. Their understanding of the target system and their expert knowledge of the safety modelling approaches will fundamentally influence the quality of safety models. Although it is impossible to have total knowledge of a human-made system in design or in operation, modellers strive to have adequate understanding of the identified or envisioned parts of a target system as far as possible. They also try to avoid the inclusion of any accidental errors in the model.

Safety Assessment Model

As the outcome of the safety modelling effort, a safety model can take various forms such as text, diagrams, tabular format, or equations. Some safety models have only descriptive power in themselves, which means that these safety models represent exactly and only the information that has been input in safety modelling, e.g. an FMEA worksheet. Whereas some other safety models have some predictive power in that further results can be given after processing of the original input data in the modelling, e.g. the generation of minimal cut sets or the probability of the top level event in a fault tree.

A general safety modelling process will run across all the elements in Figure 12. The specific steps in a safety modelling process will vary according to the various procedures of the different safety modelling methods adopted. For a target system with specific safety concerns or safety requirements, we should start the safety modelling task with a clear intent and scope (in terms of its purpose with the safety case and its desired contribution to knowledge). The intent of safety modelling originates from our concerns about the safety characteristics or behaviours of the system to be modelled. We are expecting to deepen our understanding of the target system through the planned safety modelling actions. Four typical purposes of the construction of safety assessment models are: a) for documentation and organization of our understanding of the safety of target systems; b) to generate safety requirements; c) to verify safety requirements; d) for design evaluation and selection. After that, modellers with the knowledge of the system to be modelled will adopt a certain safety modelling approach for the construction of their models, during which both explicit and implicit assumptions, abstractions and simplifications are made in order to make the problems in safety modelling tractable for the modelling intent. A system safety assessment model will be generated through this process. The safety assessment model and associated results (the model itself and sometimes further results defined and generated by input-data processing and model implementation) will provide information to illustrate that the safety concerns are controlled or mitigated or that the safety requirements are satisfied.

3.3.2 Contributing Factors to ‘Wrong’ Models

Figure 12 describes the overall context of the safety assessment domain. There are four interim blocks in Figure 12 between a *Safety Assessment Model* and the *Target System* and *Safety Concern* represented by the *Safety Assessment Model*, namely, *Modeller*, *Modelling Method*, *Modelling Tool*, and *Safety Modelling Process*. The validity of a *Safety Assessment Model* is influenced by these four blocks. Typical factors associated with the four blocks that may contribute to the errors made in models are grouped and presented below.

- Systematic factors
    - The *Modeller*’s incomplete understanding of the target system
    - The *Modeller*’s intention of presenting ‘safe’ systems
    - The *Modeller*’s conceptual errors
    - The limitations and assumptions of the *Modelling Method*
    - The limitation and configuration of the *Modelling Tool* (if a tool is adopted)
- Random factors
    - The *Modeller*’s performance in the *Safety Modelling Process* (appropriate usage of the method and the tool)
- Judgmental factors
    - The decisions and assumptions made in the *Safety Modelling Process* (regarding to the scope, representation and reasoning)

Additionally, safety assessment models can be used in a ‘wrong’ way, if put into an inappropriate context or used beyond their validity envelope. It means that we need to be careful of our interpretation of the safety modelling results when we employ a model as evidence in any decision-making process.

The three types of factors indicate that information concerning *Modeller*, *Modelling Method*, *Modelling Tool*, and *Safety Modelling Process* should be captured to assist the evaluation of safety assessment models in terms of their trustworthiness. However, existing practice shows that such information is usually implicit, informal, or insufficiently or fragmentally documented from the information on our understanding of the problem domain, rather than being systematically documented as a part of a safety assessment artefact. Therefore, we suggest that descriptive information elements concerning modelling processes and decision information elements generated during modelling processes should be explicitly integrated with the descriptive information elements concerning a problem domain in a single generic model of safety assessment artefacts. The core data meta-model introduced in Section 3.4 takes into account the data elements that are associated with the context of the modelling domain illustrated by the domain model in Figure 12. These data elements are useful for more comprehensive evaluation of safety assessment evidence (including examination of the models and justification of thie usage as evidence items in safety cases).

3.4 Safety Assessment Core Data Meta-Model

In this section, we propose a core data meta-model (CoreDMM) for the purpose of evaluating safety assessment models as evidence in safety cases. This model represents the structure of information elements associated with a specific model instance and its modelling process. It covers more than would a meta-model of an individual safety method or technique and it is more generic in terms of the representation of building blocks of various models. The features of CoreDMM include:

- Increased transparency of data elements affecting or associated with the validity of safety assessment models

- Highlighted data elements that determine the evidential capability of safety assessment models

- Enhanced communication between different models via a set of generic core modelling constructs.

CoreDMM is a UML model at the M1 level in the OMG meta-modelling structure. It is neither a meta-model at the M2 level like the UML concepts model nor a meta-model at the M3 level like the MOF, but is a meta-model of the instances of specific safety analysis in particular analysis techniques. This is unconventional. But it is unrealistic to define a core date model at M2 level that is capable of representing all variants of existing meta-models of safety analysis techniques with diverse modelling constructs. Depicting CoreDMM at M1 level is convenient for a direct view of its relationships with various model instances. However, it may bring difficulties in terms of making use of the advantages of model-driven engineering (MDE), because some model operations (e.g. the concept mapping between CoreDMM and existing safety meta-models) are within the M1 level.

The reasons for not employing existing meta-models of safety analysis techniques directly are two folds. Firstly, there are a number of safety analysis techniques (with varied modelling constructs), each of which has multiple versions of meta-models developed by different researchers. Although it is disadvantageous to omit some technique-specific details in model representation, unifying the core concepts and in a common core data model will ease the communication and comparison between models in different techniques, which is valued by the model inconsistency analysis in Chapter 5. Secondly, existing meta-models derived from safety analysis techniques have not explicitly taken into account the data about the safety assessment process and the validity context of the modelling output. But those data are indispensable for evidence justification in safety case development. Therefore, we develop CoreDMM on the basis of extracting core data elements from the existing safety meta-models (e.g. [46, 145, 212]) for safety analysis techniques, but enclosing more data elements in CoreDMM than the ones presented in them, considering the need of use and evaluation of safety assessment evidence within safety cases according to the domain model presented in Section 3.3.1.

3.4.1 Core Data Meta-Model (CoreDMM) Overview

The proposed CoreDMM of safety assessment is depicted in Figure 13. A safety assessment model comprises four principal groups of data elements – the meta-data, the validity context data, the substance elements and the construction elements. There are two types of 'whole-part' relationships in Figure 13. The filled diamonds in Figure 13 depict 'composition'[175], which means that the deletion of a part will be triggered if the whole is deleted. The unfilled diamonds in the figure depict 'aggregation'[175], which means that a part can belong to a whole but can also stay even if the whole is deleted.

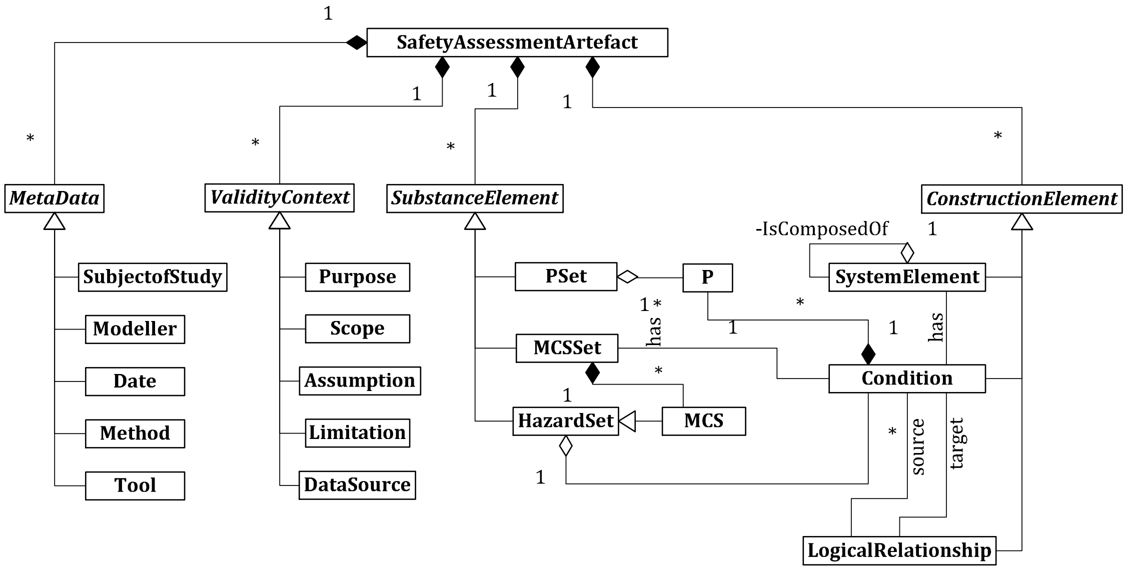


Figure 13 Safety Assessment Core Data Meta-Model (CoreDMM)

The *SafetyAssessmentArtefact* is the overall class that contains all of the structured data elements or information emerging throughout a safety assessment modelling process. Some are associated with the description of the implementation and technology sectors of the modelling domain (as shown in Figure 12); whereas others are formed, dictated or generated to represent understanding of the safety concerns of the subject system (real or envisioned) in the problem domain (as shown in the domain model in Figure 12).

The *MetaData* of *SafetyAssessmentArtefact* are data elements or information about a safety assessment model. These *MetaData* present the key facts related to the *SafetyAssessmentArtefact*, such as what is the subject of a model, who created the model, when is the model created, which modelling method is adopted, with which tool is the model constructed, and so on. It can also be used for the identification of a *SafetyAssessmentArtefact*. Usually, the *MetaData* elements are not controversial in terms of their concrete values, which are determined with the commencement and completion of modelling. They are not information about or associated with the problem domain. But they are not always explicitly documented or stored with other elements of a *SafetyAssessmentArtefact*. At the implementation level, there are no constraints on whether to store *MetaData* with data together or separately, as long as *MetaData* have been documented and are easily accessible during safety case development. The subtypes of the *MetaData* contain only simplified content for identification, e.g. the name of a modeller, the name of a tool, rather than more precise details such as the competency of a modeller, the features of a tool or a method.

The *ValidityContext* of *SafetyAssessmentArtefact* are also data elements or information about a safety assessment model. But they are different from the *MetaData* in that they constrain the overall validity of a *SafetyAssessmentArtefact* and they are directly associated with the problem domain being modelled. This set of data elements are often overlooked or not explicitly documented in real practice. Some of them are also relatively 'resistant' to structured documentation and can only be depicted in an informal way using natural language (e.g. the assumptions made by a model). However, they are important when we want to use a model in safety cases as an item of evidence. Many standards and guidance documents have addressed the documentation of these data elements in the requirements on models or usage of models, e.g. [68, 151]. But whether these data about a *SafetyAssessmentArtefact* have been sufficiently elicited is highly dependent on the domain knowledge and expertise of the modeller. They are valuable for the proper comprehension and correct interpretation of a model in its application context, even though these data are based on the modeller's declaration or assumptions concerning the purpose, scope, assumption or limitation of a safety assessment model. For example, we may need to know the scope of a safety assessment model in order to avoid abuse of the model or its results beyond its claimed scope. The adoption of safety analysis results for a different purpose from its original intent should be carefully justified.

The *SubstanceElement* of *SafetyAssessmentArtefact* is the information element about the essential contents of safety assessment models, which are conclusive data elements which depict the safety characteristics of a domain objects. They are usually associated with the purpose of modelling and the capability of the modelling methods. From the existing literature [46, 79, 212], we have found that the three most common types of results expected from existing safety assessment models are: a set of identified hazardous conditions, the minimum combinations of conditions that can lead to an undesired consequence, and the probability of a specific undesired condition. A *SafetyAssessmentArtefact* may provide multiple substance elements; they can be grouped in to three set types – the set of probabilities (*PSet*), the set of Minimal Cut Sets (*MCSSet*), and the set of identified hazards (*HazardSet*). Explicit representation of The *SubstanceElement*s of a *SafetyAssessmentArtefact* will ease the communication between the content of safety assessment artefacts and the claims of safety arguments in safety assurance process. For large-scale safety assessment models for complex systems, in particular, the *SubstanceElement*s will enable clear and easy access to the core evidential content expressed by models. It is worth noting that *PSet* and *HazardSet* are the 'aggregation's of *P* or *Condition*. It is because the *SubstanceElement* is highlighted specifically for eliciting parts of *SafetyAssessmentArtefact* that are employed as evidence source data. The

SafetyAssessmentArtefact, rather than the subtypes of *SubstanceElement*, is the physical container of the elements aggregated by *PSet* and *HazardSet*. In addition, *P* and *PSet* are typical *SubstanceElement* of quantitative safety assessment models. Although quantitative modelling results are not considered in this thesis, they are included in CoreDMM for the overall representativeness of the model.

The *ConstructionElement* of *SafetyAssessmentArtefact* represents the key building blocks in the domain of safety assessment, which are fundamental and shared across different qualitative safety analysis techniques. Here we carry on the style of Wilson's safety data model [212]. The *ConstructionElement* presented in CoreDMM is more abstract than the concrete building blocks in safety analysis techniques. *Condition* is the kernel of qualitative safety meta-models. Although it is depicted with varied terms (such as hazard, accident, failure, failure mode, failure event) in other safety meta-models, the essence of this notion is to depict the failure behaviours of the subject under analysis or factors that are contributors to these concerns. The 'state' and 'flow' notions in AltaRica language are not modelled as a condition in this meta-model, because they are formal modelling constructs and are different from the common cause-effect modelling constructs shared by other qualitative safety analysis techniques. The *LogicalRelationships* between conditions represent our comprehension and knowledge of system safety behaviours, which are valuable for decomposing safety objectives and prioritizing the focus of safety activities. These relationships are an important and inherent part of both structural safety analysis techniques and their results. Whether or not a system is an element of the core data model is arguable. Some safety meta-models do not present it explicitly as a component block in the models, e.g. the FTA meta-model in [46]. Nevertheless, many safety meta-models treat a system as an explicit element, which eases the integration of system structural or functional modelling data with the corresponding system safety analysis data. *SystemElement* presented in CoreDMM indicates the system elements that are considered in safety assessment modelling. None consideration of a higher level system element would not prohibit the consideration of its lower level components. Thus it is modelled as an aggregation of itself, which is different from the composition relation between system elements usually presented in a system model. *ConstructionElement* covers only the core generic modelling constructs in safety assessment modelling. If other details of a model in a particular safety technique are needed, we still need to use meta-models of specific techniques to document the model.

3.4.2 Representation of Typical Safety Assessment Models

CoreDMM is capable of expressing the core analysis data of safety assessment models based on major qualitative safety analysis techniques. Some examples of models represented with CoreDMM are presented in this section. Safety assessment models vary a lot in the structure of their construction elements, but not much in the structure of the meta-data, validity context and substance elements. Therefore, one example is presented to demonstrate the instantiation of the *MetaData*, the *ValidityContext* and the *SubstanceElement* of CoreDMM; while three examples are presented to demonstrate the instantiation of the *ConstructionElement* of CoreDMM. Unlike the conventional way of using object models, the four example models presented are not used for expressing object sequences or activities in an application scenario, but are used for illustrating the expressive capability of CoreDMM through instantiating it with data from concrete safety analysis artefacts.

In practice, some of the *MetaData*, the *ValidityContext* and the *SubstanceElement* of a safety assessment model may have been described in natural language in the safety assessment report that is delivered after a safety assessment modelling process. To ensure that none of the required elements is lost and that there is an easy means to access these information elements, we suggest documenting meta-data according to the structure in CoreDMM. The example instantiation presented in Figure 14 is based on a hypothetical safety analysis report, developed only for the purpose of illustrating the instantiation of the three types of data elements. Each block in Figure 14 represents an object at the M0 level that instantiates an element of CoreDMM at the M1 level. For example, ‘SAM-Hypo: SafetyAssessmentArtefact’ in Figure 14 is an object instantiation of *SafetyAssessmentArtefact* of CoreDMM; ‘SAM-Hypo:SafetyAssessmentArtefact’, a concrete analysis artefact, uses ‘FMEA’ as the safety analysis technique. ‘FMEA:Method’ in Figure 14 is the instantiation of *Method* of CoreDMM.

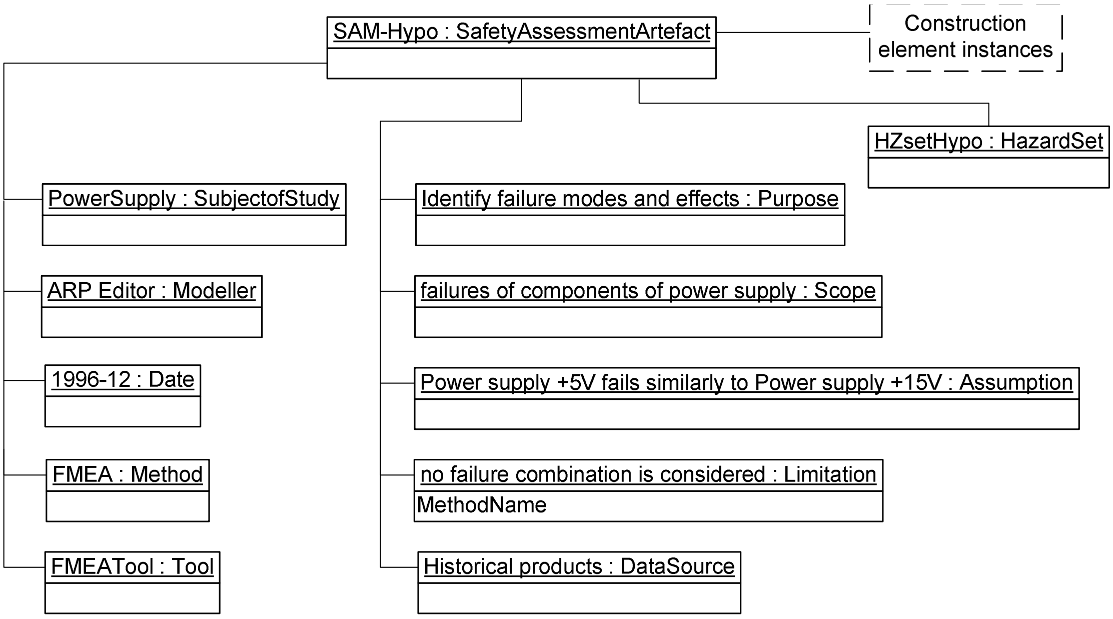

SAM-Hypo : SafetyAssessmentArtefact
Construction element instances
HZsetHypo : HazardSet
PowerSupply : SubjectofStudy
ARP Editor : Modeller
1996-12 : Date
FMEA : Method
FMEATool : Tool
Identify failure modes and effects : Purpose
failures of components of power supply : Scope
Power supply +5V fails similarly to Power supply +15V : Assumption
no failure combination is considered : Limitation
MethodName
Historical products : DataSource

Figure 14 CoreDMM Instantiation of a Hypothetical Analysis

Table 3 illustrates a record from an FMEA table (adapted from ARP 4761 [181]).

Function Name	Failure Mode	Flight Phase	Failure Rate (e-6)	Failure Effect	Detection Method	Comments
Power Supply of +5 Volt	+5 V out of spec.	All	0.2143	Power supply shutdown	Power supply monitor trips	BSCU channel fails

Table 3 An FMEA Record (excerpted from [181])

The partial safety assessment model of the FMEA data presented in Table 3 is illustrated in Figure 15 as an instantiation of CoreDMM. Each block in Figure 15 represents an object corrsponding to the core analysis data in Table 3. For example, ‘Power Supply: SystemElement’ in Figure 15 represents ‘Power Supply of +5 Volt’ in the first column of Table 3. Similarly, the blocks in Figure 15 are objects instantiating elements of CoreDMM. For example, ‘Power Supply: SystemElement’ in Figure 15 is an instance of *SystemElement* in CoreDMM.

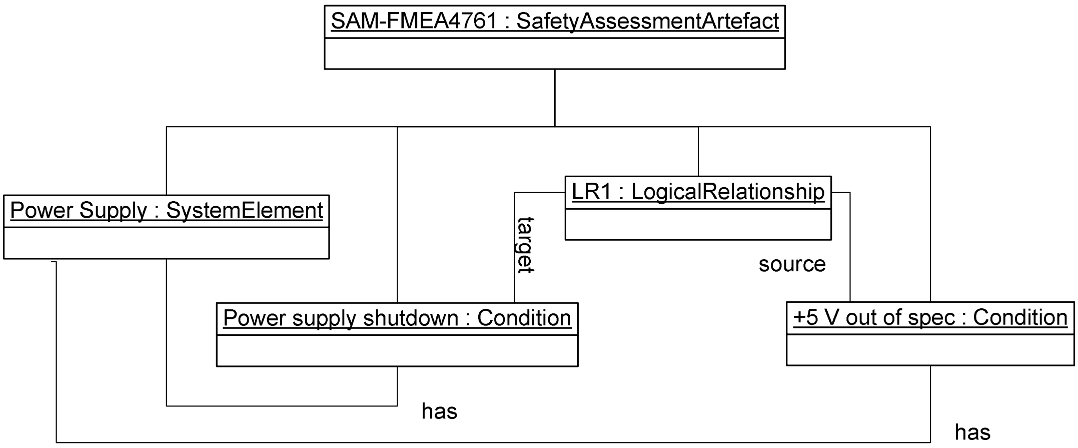

SAM-FMEA4761 : SafetyAssessmentArtefact
Power Supply : SystemElement
LR1 : LogicalRelationship
target
source
Power supply shutdown : Condition
+5 V out of spec : Condition
has
has

Figure 15 CoreDMM Instantiation of a Record in FMEA

A branch of a fault tree is adapted from ARP 4761 [181], as illustrated in Figure 16.

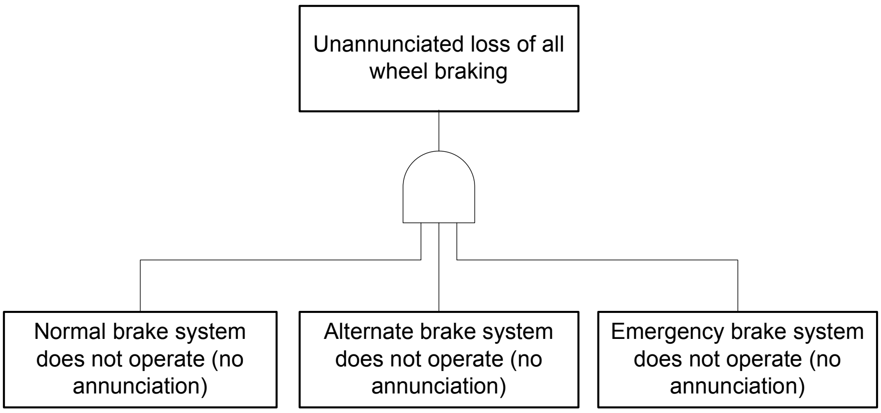

Unannunciated loss of all wheel braking
Normal brake system does not operate (no annunciation)
Alternate brake system does not operate (no annunciation)
Emergency brake system does not operate (no annunciation)

Figure 16 Part of a Fault Tree (excerpted from [181])

A partial safety assessment model for the fault tree analysis data presented in Figure 16 is illustrated in Figure 17 as an instantiation of CoreDMM. The instantiation aims to illustrate how the construction element instances of a fault tree analysis artefact are expressed and organized on the basis of CoreDMM, rather than to achieve good communication with field engineers, which is better achieved with the traditional graphical fault tree notation. By contrast with the previous example of FMEA instantiation, some of the blocks in Figure 17 use information extracted from the fault tree in Figure 16. The data about system elements are embodied in the description of events in the fault tree in Figure 16, but are represented explicitly as individual blocks in Figure 17. For example, the 'Normal brake system does not operate' event in Figure 16 is mapped into two objects in Figure 17 – 'NormalBrakeSystem:SystemElement' and 'NBS-not operate:Condition'. In this way, the four events in the fault tree in Figure 16 have been recorded as four instances of *SystemElement* and four instances of *Condition* in Figure 17. Although the extraction of the data needed by CoreDMM requires considerable input and knowledge from users, the efforts is repaid by clearer relationships between data elements than otherwise occur in some analyses (e.g. the association between system elements and failure events hidden in the informal description of failure events in a fault tree). CoreDMM is not designed for automatic model transformation between varied safety assessment models, but to document concerete analysis results with the granuality to support further analysis of safety assessment models.

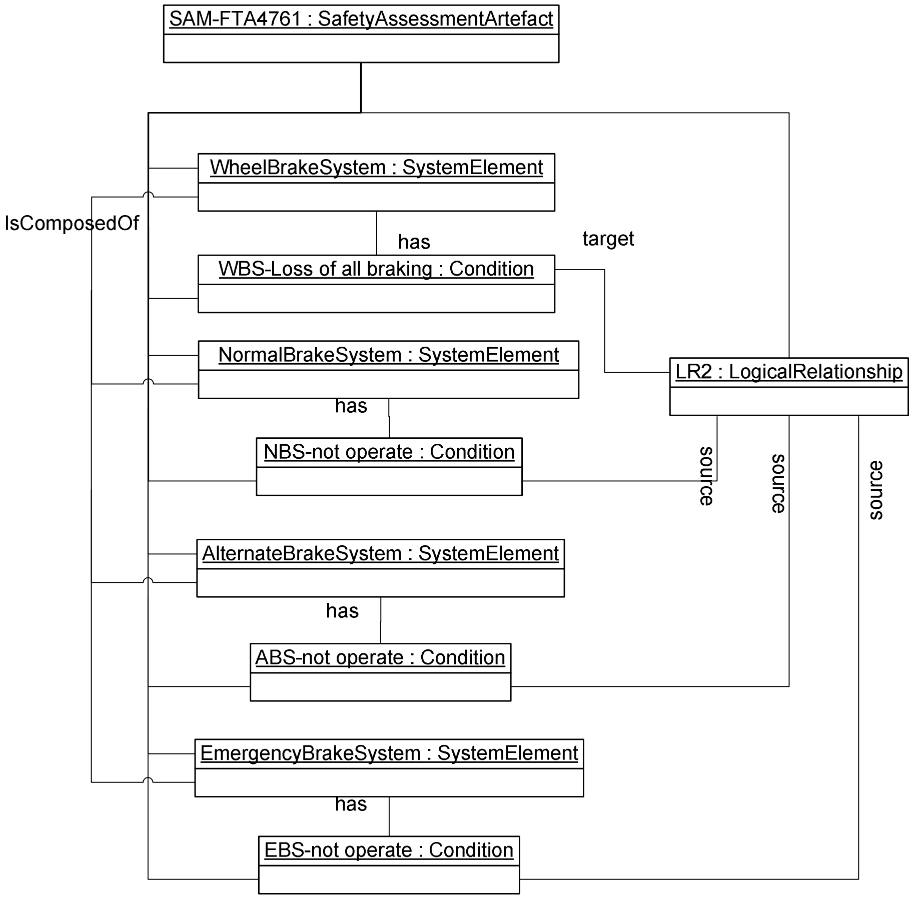

SAM-FTA4761 : SafetyAssessmentArtefact
WheelBrakeSystem : SystemElement
IsComposedOf
has
target
WBS-Loss of all braking : Condition
NormalBrakeSystem : SystemElement
LR2 : LogicalRelationship
has
NBS-not operate : Condition
source
source
source
AlternateBrakeSystem : SystemElement
has
ABS-not operate : Condition
EmergencyBrakeSystem : SystemElement
has
EBS-not operate : Condition

Figure 17 CoreDMM Instantiation of a Part of a Fault Tree

The construction elements of an AltaRica model are different from the traditional FMEA or FTA model. The *LogicalRelationship* Class in CoreDMM is not suitable for representing the logical relations embedded in formal propositions in the AltaRica Language. But the data elements shared by traditional models and AltaRica models, such as the substance elements, the system elements and conditions, can be represented on the basis of CoreDMM. These shared data are the foundation of potential analysis carried out across the different types of models.

Figure 18 presents a typical AltaRica data flow declaration of a *Node*, which represents the nominal and failure behaviour of a valve [169]. From the example, we can see that the particular features of a node are described in state variables: the internal changes of a valve, triggered by ‘events’, follow the transition rules defined in the ‘trans’ section of the declaration; the communication content of the node with other nodes is defined as flow variables; the constraints on the node description variables are described as assertions.

```
node Valve;
   state open:bool, stuck:bool;
   flow i:bool:in, o:bool:out;
   event open, close, fail;
   trans
      open and not stuck |- close -> open:=false;
      not open and not stuck |- open -> open:=true;
      not stuck |- fail -> stuck:=true;
   assert open => i=o, not open => not o;
init open=true, stuck=false;
edon
```

Figure 18 An Example of AltaRica Data Flow Declaration (from [169])

Figure 19 depicts a model of a valve instance in a specific system, originally defined by the AltaRica Node description in Figure 18, represented with CoreDMM instances. For example, three events declared in Figure 18, ‘open’, ‘close’, ‘fail’, are represented as three objects of ‘Condition’ in CoreDMM, shown as ‘open:Condition’, ‘close:Condition’ and ‘fail:Condition’ in Figure 19.

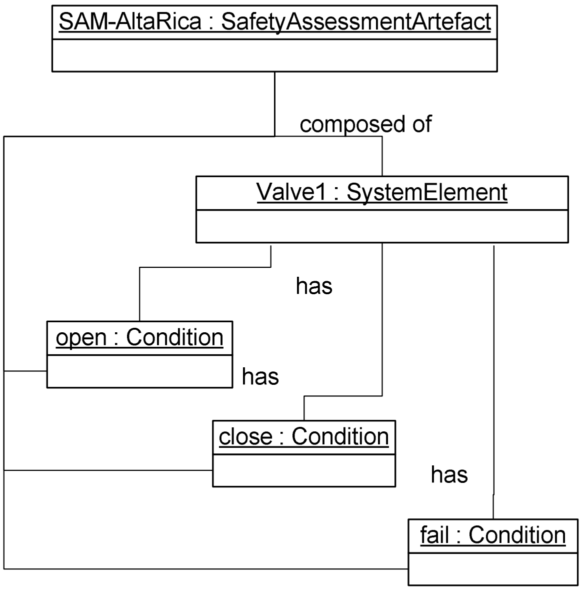


Figure 19 CoreDMM Instantiation of a Valve Defined in AltaRica

Some of the traditional analysis information is not depicted in the instantiation of CoreDMM, such as the detection method of a failure mode in the FMEA model, the type of logical relationship AND between events in a fault tree. But these can be recorded as an attribute of the class objects. The examples aim to show the relationships between the key construction elements in CoreDMM and the potential real model data only.

3.4.3 Relations with Other Safety Assessment Meta-Models

As we have stated in Section 3.2, there are some existing safety meta-models. In this section, we outline three such models, paying particular attention to their core characteristics and their relationships with CoreDMM.

SEI Safety Information Model

Firesmith, in [79], presents an information model which identifies and defines the core functional concepts underlying safety engineering and emphasizes their similarity to the concepts which underlie security and survivability engineering. As a meta-model for core domain concepts, this model aims to provide a standard terminology and a set of concepts that explain the understanding of safety, but it is not intended for direct instantiation by real specific safety analysis scenarios. In this model, the concept of *safety* has been treated as a quality factor, which brings together the notions in safety with the notions in requirements engineering. No other safety meta-models have *safety* as an individual node in the model, which implies the high ontological abstraction level of the model that is determined by the nature of this model.

The subjects in common addressed by safety analysis approaches are presented in this model as *Safety Risk, Hazard, Accident,* and *Harm*. Causal linkage, which is a focus of many safety analysis approaches, is presented between the four concepts in the model. In the context of [79], a *System* in the model is the software product. But the model could be adapted to the safety of other (non-software) systems, for example it could mean other technological systems designed and operated by human being. In this model, *Asset* and *System* are linked together but presented separately in the model as two nodes. It is beneficial to differentiate *Asset,* which covers things of value and need to be protected, from *System,* which generally covers only the object under analysis. The concept of *Asset* helps us describe the subjects of our safety concerns more precisely.

Wilson's Safety Data Model

Wilson et al develop an abstract data model [212] that aims to integrate safety analysis data from different safety analysis techniques. It is not designed according to a specific safety analysis technique, but instead extracts common core data elements from ten approaches regarding to hazard identification, cause analysis, consequence analysis, risk analysis and system modelling. The flow of data between different safety analyses is managed and the rules governing interactions between the data are maintained in order to have a coherent view of the system and its safety characteristics along with the construction of safety analysis. However, the safety assessment process is invisible in this model and the model does not distinguish the function of data elements even if the data elements were generated on the basis of different modelling techniques.

System has been identified as a necessary entity in this data model as a result of the circumstance that all the safety analysis techniques under this study are carried out with relations to a certain kind of system model, implicitly or explicitly. So the minimum requirement is that the data model can model systems and their components. The core entity in the model is named as *Condition*. A condition "*is an abstraction used to capture some 'state of affairs' in the system, be it an event or system state*"[212]. Conditions can be faults, failures, hazards, and accidents, which play the same role as the corresponding three notions adopted in the aforementioned SEI safety meta-model. The causal relationships between conditions are depicted in more detail, given that the cause and the consequence are all presented as individual entities. On the basis of this data model, the pair-wise dataflow rules between entities provide assistance to the completeness and consistency between safety analyses, e.g. the consistency between safety analysis and the system model, or the completeness of the consequences considered in HAZOP and FTA.

Briones et al's Safety Model

Many safety analysis techniques have their language syntax expressed by meta-models. The meta-models in [46] represent the two most widely-used structural analysis techniques and are developed to complement the system architectural modelling languages, FMEA and FTA. In contrast to the previous two safety meta-models, Briones et al place their safety analysis meta-models at the M2 level of the OMG modelling infrastructure. Both system modelling languages and safety analysis languages are viewed as instances of MOF. Their meta-models assist the transformation of safety-annotated system architecture models to safety analysis models of the system.

The first feature of the two meta-models is that both of them have a specific element for the description of the analysis as a whole, the *FmecaSystem* and the *FtaSystem* in the meta-models. These two blocks serve as containers, which hold all of the global analysis properties and the main blocks of the system or the root block of a fault tree. The *Block* in the FMECA meta-model represents the system component hierarchy, whereas there are no implicit system blocks or component blocks in the FTA meta-model. The *Event* block in the FTA meta-model can be a failure of a system element, but its relation to the system entities is not depicted in the model. The system element and associated relationships are useful if we want to integrate safety models with system models.

The *FailureMode* depicts how the system fails at different hierarchical levels. The causal relationship between failure modes is depicted as an association on the concept of *FailureMode* itself. Similarly, the *Event* represents the core analysis subject in the FTA meta-model. However, the relationships between events are not self-pointing, but between the *Event* and the subtype of *DerivedEvent*. An extra feature of the FTA meta-model is that the analysis results generated after the processing of FTA inputs are also presented in the model, e.g. *MinimumCutSet*. For safety analysis methods that have some processing power, the form of processed results turn out to be the major transferable information queried and referenced for further usage. To present them as blocks separately but not to generate items afresh is more convenient for the evaluation of safety analysis.

Our core data meta-model differs from the three meta-models discussed here by considering the following three issues. Firstly, the detailed description of the safety assessment model as a whole (with metadata and validity context data) is not included in other safety analysis meta-models (except the ones in [46], some of the data are elicited as modelling blocks). But the overall description of safety analysis results is very useful throughout the evaluation of the results. Secondly, the pre-existing safety meta-models have been developed for varied purposes, but few incorporate the idea of integration of safety modelling and assurance needs beforehand. Therefore, the substantial elements indicating the safety assessment outcomes, which directly relate to safety requirements or safety claims, have not been addressed with appropriate emphasis in these models. In short, the information needed for evaluation of safety assessment models is not fully covered and the models are not organised with the consideration of facilitating and supporting the task of evaluation. Thirdly, the core construction elements are in a more abstract form than the construction elements in other meta-models for safety analysis techniques (except for the model in [212]). It is not sufficient to represent all analysis data completely, but forms a common basis for understanding and establishing relationships between different models.

3.4.4 Relations with System Assurance Meta-Models

From the viewpoint of system safety assurance, it is necessary to understand how safety assessment models are related to or integrated with safety arguments as safety evidence in safety cases, since they are one important type of information widely used to support the argument presented in safety cases. The structure and content of data elements in CoreDMM is designed with the purpose of being clear and sufficient to meet the need of using safety assessment models as evidence. In this section, we will explicate the relationship of CoreDMM with argumentation and evidence, basing our discussion on the analysis of existing assurance meta-models presented earlier.

As we described in Chapter 2, the OMG has published ARM, the meta-model for argumentation, and SAEM, the meta-model for evidence, in order to facilitate and normalize data exchange and communication in software assurance cases. The Structured Assurance Case Metamodel (SACM), which integrates ARM and SAEM, is shortly to be released in its first version.

Currently, ARM provides an element *InformationElement* which can be instantiated as a reference description to link argument elements with "the citation of a source of that relates to the structured argument" [156]. The *InformationElement* in ARM serves as a placeholder for connecting real information sources with the argument via different subtypes of *ArgumentLink* in ARM. However, there are no constraints or recommendations presented in ARM regarding what kind of information sources are prohibited or expected and which subtype of *ArgumentLink* should be used for a particular situation.

Currently, SAEM is generic and it combines the entity of evidence and other evidence-related data into a whole package. But it depicts the entity part in such a simple way that it has not provided the features necessary to record the details of a complex item of evidence, such as a safety assessment model. CoreDMM presented in this chapter can serve as a special case of the entity part of evidence, while other evidence-related data (such as propositions made on entities and evidence properties) can be separated from the entity part and be addressed by a particular evidence meta-model (this will be presented in Chapter 4 below). This enables more practical manipulation of complex information concerning evidence and its relationships with arguments. The relationships between the three meta-models are depicted in Figure 20.

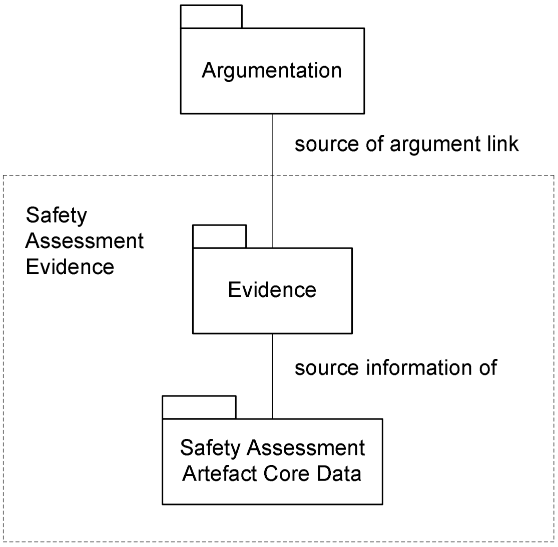


Figure 20 Relationship of Safety Assessment Artefact and Assurance Meta-Models

In this thesis, we will study the relationship of safety assessment evidence with argumentation on the basis of ARM. But in terms of the evidence itself, as we can learn from Figure 20, it is explored with two distinct meta-models (for the entity part of evidence and for the conception part of evidence respectively), but not on the basis of SAEM, which overlaps partially with each package depicted in Figure 20.

3.5 Application of CoreDMM

The safety assessment CoreDMM presented in Section 3.4.1 is formulated on the basis of other safety assessment meta-models and the modelling domain model presented in Section 3.3.1. CoreDMM aims to provide support for structuring the key data elements that are needed in using safety assessment models as evidence and evaluating safety assessment models in the context of safety arguments. To be more specific, CoreDMM can be applied in the following areas:

- To support the communication and integration of models as evidence within safety cases. CoreDMM separates different data elements in safety assessment models into four groups, which facilitates accessing the most appropriate data elements during the development of both the primary safety argument and the confidence argument.

- To support the examination of consistency between different safety assessment models. CoreDMM enables bypassing of the inconsistent concept frames of different modelling techniques, and makes model comparison for inconsistency identification more manageable at the model instance level. The interpretation and justification of inconsistencies so identified will also be supported by the validity context data required by CoreDMM.

- To support the evaluation of safety assessment models in the context of safety arguments. Beyond the primitive modelling constructs of a model (e.g. failure event) CoreDMM integrates additional data elements that are necessary for describing a safety assessment model as an overall entity, such as the requirements on its metadata, its validity context and its substance results. Evaluating a safety assessment model with respect to the four groups of data elements comprehensively will increase our confidence in using a model appropriately in safety cases.

3.6 Summary

In this chapter, a domain model is presented for a better understanding of the context of safety assessment modelling and a generic data model (CoreDMM) for describing the common content of safety assessment modelling artefacts. The domain model of safety modelling captures the major factors that are potentially hazard sources of flawed models. It brings forward the importance and need of explicit consideration of 'MetaData' and 'ValidityContext' in CoreDMM in support of the evaluation of safety analysis results. Besides that, CoreDMM also takes into account construction elements of typical safety assessment models (as other meta-models of safety analysis techniques have done) and highlights the substance elements of safety assessment models for their evidential role. The set of data elements in CoreDMM provides a structured view of safety assessment artefacts, which forms a common basis for analyzing the content of different model instances that are based on different safety analysis techniques. Chapter 4 provides a structured view of generic evidence that signifies that safety evidence is more than safety artefacts. In combination, Chapter 3 and Chapter 4 establish a detailed view of safety assessment evidence. Chapter 5 demonstrates how inconsistency analysis of safety assessment models can be conducted with the support of CoreDMM. Chapter 6 describes how the justification of the usage of a model as evidence can be structured around the factors presented in the domain context of safety assessment modelling.

4 A Model of Argument-Evidence Interface

4.1 Introduction

In Chapter 2, we presented the history of safety cases and recent developments in theory and practice in that field. The concept of an argument presented using informal logic, as a means of demonstrating system safety and facilitating safety management, has been adopted alongside system development and operation in many different industries. The argumentation part of safety cases, such as issues regarding safety claims or the inferential relationships between them, has been well-developed in the past two decades. By contrast, the concept of evidence in safety cases, which is also an important component of safety cases as required by regulations, has received less attention in existing academic work, especially its role within safety cases and its relationship with arguments.

From the published literature, standards and guidance, we observe that the confidence in safety evidence is significantly threatened by the following issues.

- No widely-accepted definition of evidence in safety. Various definitions focus on different aspects of evidence, e.g. its source data, its documentation or its role in supporting an argument.
- A simplified view of the relationship between evidence and argument. The interface between evidence and argument is usually presented only as references to source data that are associated with domain safety claims. However, the reasoning linking what we can obtain from evidence to the domain safety claims being supported by the evidence is unclear. It is difficult, with current documentation and representation of evidence in safety cases, to determine how, and to what extent, the items of evidence fit their role in a specific application context.
- Unstructured justification of evidence. This issue is caused, on one hand, by an unclear understanding of confidence in safety. Actually, existing guidance and practice on review and evaluation of safety deliverables and activities have not distinguished sharply between demonstrated safety and demonstrated confidence in the adequacy of evidence. On the other hand, given the previous two deficiencies, there has not been a sufficiently clear and structured view of the features of evidence that are expected for establishing our confidence in safety cases.

In this chapter, we define the concept of evidence in the context of safety cases on the basis of comparison of definitions in several disciplines. We also define a model of evidence (EviM) in order to have a clear view of the grounds on which established confidence associated with safety cases is based. Within this model, the notion of the 'evidence assertion' is introduced as the interface element to help integrate safety assessment evidence and argument effectively. This model of evidence will motivate a more comprehensive documentation of items of evidence as objects linked with arguments, rather than simply as data source references that embody the links to the items of evidence. The data elements within this model are designed to support a more explicit evidential role of each individual item of evidence and facilitate the potential reuse of an item of evidence in other application context.

Before we introduce EviM in Section 4.5, we analyse the nature of evidence in Section 4.2, the classification of safety evidence in Section 4.3, and the evidence-argument relationship in practice in Section4.4. The relationship of safety evidence to confidence in safety cases is finally discussed in Section 4.6.

4.2 The Concept of Evidence

In practice, the concept of safety evidence is not well-elaborated in guidance and the understanding and usage of evidence is diverse. As presented in Chapter 2, a commonly-cited definition of a safety case is from DS 00-56 [149].

> *"A structured argument, supported by a body of evidence that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given operating environment"*.

It is clear from the definition that 'a body of evidence' is a part of a safety case. However, DS 00-56 does not provide a definition of 'evidence'. This, sometimes, leads to inconsistent and arbitrary usage of this notion by practitioners. For example, evidence in safety may be viewed as artefacts, documents, facts, or statements of facts in different situations. This, unsurprisingly, causes confusion and sometimes misconception of safety evidence in safety engineering practice. It is also harmful for the development of compelling safety cases.

In recent years, the importance of evidence has been highlighted in the development of dependable software systems [106] and more people are concerned about the inspection, analysis, and requirements on evidence presented in safety cases [92, 96, 106, 170].

However, the meaning of this concept, which is crucial for the proper usage of evidence in safety cases, has not been addressed adequately.

The meaning of a term or concept has two dimensions [146]:

- The *connotation* of the concept, which is also referred as intension, essence or nature. It depicts the abstract meaning of a term, which serves as shared principles and characteristics that apply to *all* objects of that concept.

- The *denotation* of the concept, which is also referred as extension or reference. It depicts the specific meaning of a term and the individuals to which the term is referred, which addresses the features of a group of individuals of the concept that are not possessed by other objects of the concept.

In this section we will probe the *connotation* meaning of evidence in different disciplines and explore the common understanding of this concept within the domain of safety cases. This is the foundation for proper comprehension, interpretation, usage and documentation of evidence in safety. The classification of safety evidence, which addresses the denotation of the concept of evidence, is presented in Section 4.3.

4.2.1 Evidence in Other Domains

Evidence is a notion that has been studied in the fields of philosophy and law for a considerable time. In the past two decades, evidence-based medicine and evidence-based health-care have grown in popularity. In order to gain an initial understanding of the concept, we refer to different definitions of evidence in three domains for insights – from the fields of philosophy, law and medicine.

> ***Definition 1***. *That which tends to prove the existence or nonexistence of some fact. It may consist of testimony, documentary evidence, real evidence*[5]*, and, when admissible, hearsay evidence.*

(From A Dictionary of Law [128])

> ***Definition 2***. *The assembled information and facts on which rational, logical decisions are based in the diverse forums of human discourse, including courts of law, and in the practice of evidence-based medicine among many others.*

(From A Dictionary of Public Health [127])

[5] Real evidence is “Evidence in the form of material objects (e.g. weapons)” [128].

> ***Definition 3**. That which raises or lowers the probability of a proposition. The central question of epistemology is the structure of this process and its ultimate rationale.*

(From The Oxford Dictionary of Philosophy [37])

The rigour of the function of evidence in the three disciplines is not the same. In the realm of law, evidence is presented to help establish (to the court) that something existed or happened in the past. Once admitted, evidence tends to work as a foundational proof that substantiates subsequent reasoning or tests hypotheses towards a truth or fact. In medicine, especially in evidence-based medical decision-making, evidence is collected from various sources and evaluated for its applicability and validity in order to determine whether it is suitable for supporting the treatment decision of a patient at hand [88]. Matching the available evidence and the specific application scenario to confirm the ‘fitness of usage’ is a primary task of evidence-based medicine. The user of evidence needs the information concerning how the evidence is generated, but is not responsible for the generation of such evidence. In philosophy, the definition has not constrained the form or content of evidence, but places emphasis fully on the *intent* of presenting evidence. The power of evidence in philosophy is of some *degree*; it confirms or refutes a proposition, but not in an absolute sense, instead changing the probability of the proposition only.

Despite the subtle differences between the three definitions, there are also some common points. Firstly, evidence is something that contains information. The information may come in different forms and from varied sources in each domain, e.g. from observation and measurement, or from expert judgment or testing and analysis. Secondly, evidence is the grounds and starting-point of subsequent reasoning towards a claim or conclusion. The information, for which we adopt something as evidence, does not need support by other evidence for its content. But testimony to the quality of an item of evidence can be supported by other evidence. Even though it is not reflected in aforementioned definitions, literature in law, philosophy and medicine unanimously highlights the importance of evidence *evaluation* or *appraisal* with significance. Because evidence can be fallible, trust in evidence must be settled by rigorous scrutiny or examination of evidence in the context of its usage. We should not attach more responsibility on an item of evidence than that which goes beyond its capability or use it in an unsuitable or inapplicable context.

The functional role of evidence in safety cases is close to the definition in philosophy, but different from the ones in law and medicine. In legal cases, evidence is used to ‘prove’ a hypothesis regarding things that happened or existed in the past. In evidence-based medicine, evidence is used to ‘inform’ something, existing like a knowledge base, which is browsed, filtered and adopted after applicability and validity checks are made within the target application context. Depending on the type of evidence, the authenticity, the relevance or other features of source information, data or material objects are challenged before its admission as supporting evidence for medical treatment decisions [88]. In safety, evidence is used to ‘justify’ something, usually the achievement of safety goals or safety claims elicited. In this regard, the trustworthiness of the source data of evidence and the evidential features of a single or multiple items of evidence are of our concern and should be justified for a compelling safety cases. We will depict these features in Section 4.5.

4.2.2 Evidence in Safety Domain

There are several guidance materials [15, 73, 158] that provide definitions of evidence in the safety domain. But each of them is presented in a particular context and has its limitations.

> ***Definition 4**. Which is used as the basis of the safety argument. This can be either facts, assumptions, or subclaims derived from a lower-level sub-argument.*

(From Adelard Safety Case Development Manual [15, 33])

> ***Definition 5**. Safety Evidence is information, based on established fact or expert judgement, which is presented to show that the Safety Argument to which it relates is valid.*

(From EUROCONTROL Safety Case Development Manual [73])

> ***Definition 6**. A document or other exhibit that provides justification to a certain claim.*

(From the OMG SAEM Software Assurance Evidence Metamodel [158])

Definition 4 addresses the concept of evidence from the perspective of its functional role in safety cases. This definition, which is proposed in context of the CAE notation [17], however, is unclear about the nature of evidence. The examples of evidence presented in this definition, e.g. facts, assumptions and subclaims, are debatable. For instance, treating subclaims as evidence may lead to confusion in safety case development.

Definition 5 clarifies both the nature and function of safety evidence clearly. Information is the core. However, it leaves out the possibility of counter-evidence which can challenge claims in safety arguments. Another issue is that a potential misconception may follow from the definition - 'if there is evidence presented, the argument is true'. It would be good to clarify that both justification of an argument structure and justification of evidence are necessary to understand the level of truth expressed by a safety argument. One way to avoid the potential misconception is to build relationships between evidence and claims, rather than between evidence and argument.

Definition 6 focuses on the documentation aspect of evidence. It is convenient for data management of items of evidence. But the nature of being an information element and its role as the grounds of argument are underspecified in the definition. There is another relevant definition in SAEM, the term of 'Evidence item', which is "*A unique element of the body of evidence, such as an exhibit, a claim, or other element of meaning associated with an exhibit, an evidence attribute of one of the predefined relations between evidence elements representing assertions made during the evidence collection and evaluation of evidence*" [158]. The 'Evidence item' addressed actually means a data element presented in SAEM, which is different from the meaning of an 'item of evidence' as we use it in assurance cases.

4.2.3 Common Basis

From the discussion presented in the previous two sections, we can see that evidence is defined in various ways and it is difficult to achieve a general definition with all features presented for all types of evidence. Nevertheless, we argue that the following aspects need to be agreed as common bases for understanding the nature and role of evidence in safety.

- Evidence is *information*, but usually more than simply just the actual source data of evidence. The source data of evidence may come from a mixture of different sources, e.g. established facts, expert judgment, outcomes of engineering activities, or field service. The source data of evidence can only properly be termed 'evidence' when it is in use for a specific purpose, e.g. supporting or challenging a specific safety claim, which may or may not be different from the initial intent of generating the source data. The propositional information associated with the use of evidence source data should be addressed as part of evidence.

- Evidence is not the same as truth. It is something that we produce and adopt to represent some degree of truth (as depicted by **Definition 3** in Section 4.2.1) or merely understanding of potential truth (in the past or in the future) from a specific perspective in a certain scope, in order to justify various safety goals. The degree of truth represented by an item of evidence is uncertain and must be subjected to rigorous evaluation within an application context.

- Evidence does not simply equate to documents or artefacts. It is more about the information that we can draw out and use as evidential grounds, rather than the physical instantiation. Artefacts from system development or safety assessment may contain more details than, and should contain, the information that is necessary for the judgment involved in designating and evaluating evidence during the development of safety cases.

- Evidence is the *grounds* and starting-point of arguments. It serves (either *supports* or *challenges*) claims within a safety argument.

- Evidence should be examined in context of safety arguments. Whilst it is possible to perform some evaluation of evidence outside the context of a specific safety case (e.g. examining the rigor of a safety assessment method) it should be recognised that this is only part of the justification of evidence that is required in the context of a safety case. Other issues to be addressed include relevance, coverage, and consistency.

- The association between evidence and safety claims is a multiplicity relationship. One item of evidence can support more than one claim; one claim can be supported by multiple items of evidence.

- The association between items of evidence and physical artefacts being cited is a multiplicity relationship. One physical artefact may provide two items of evidence. For example, a fault tree report may contain both the quantitative analysis result of a fault tree and the human review results of that fault tree. The partition and organization of information into artefacts is dependent on particular practice in the systems engineering life cycle.

The working definition of evidence proposed in this thesis is:

> *Evidence is information that serves as the grounds and starting-point of (safety) arguments, based on which the degree of truth of the claims in arguments can be established, challenged and contextualised.*

4.3 Classification of Evidence in Safety Cases

This section presents a discussion on the classification schemes of safety evidence that extrapolates the denotation of the concept of evidence.

Schum proposes a ‘substance-blind’ approach of classifying evidence [183], which sets classification schemes of evidence regardless of the substance or content of an item of evidence. The recurrent classification schemes developed by Schum (such as classification based on believability, relevance, or inferential force) are based upon the ‘inferential credentials or properties’ of evidence, within its argumentation context. In the safety domain, there are two popular classifications of safety evidence presented in safety standards and guidance [48, 73, 148]. One classification stresses the directness or indirectness of support provided by evidence, which is similar to the classification scheme set by Schum based on the ‘relevance’ property of evidence; another stresses the data source of an item of evidence, which presents the varied denotations of evidence in safety.

Firstly, items of evidence in safety cases can be divided into two groups, according to their relationships with the claims being supported – direct evidence and backing evidence [48, 73]. Direct evidence is articulated as the evidence of system safety. Backing evidence is usually only indirectly relevant to system safety. It is used as evidence for increasing confidence only, rather than demonstrating the level of safety achieved. It may be process evidence, evidence related to the qualification or features of a tool, a method or personnel associated with the development of a system, or evidence from the review of specific analysis results. This way of classification of evidence focuses primarily on the ‘relevance’ between evidence and claims in arguments. Where the arguments in safety cases are clearly distinguished as ‘safety argument’ and ‘confidence argument’ [97] it is clear that direct evidence belongs to the safety argument and backing evidence belongs to the confidence argument. Direct evidence and backing evidence have complementary roles in safety cases. Without direct evidence, system safety cannot be demonstrated sufficiently. Without backing evidence, the confidence in safety cases cannot be well-established.

Another kind of evidence classification is based on the type or feature of the source data of evidence. But the individual classes are not the same in different guidance materials, depending on the nature of target systems covered by the guidance. Schum states that the task of evidence classification may be endless or fruitless if it is categorized by the substance of evidence [183]. However, it is common practice in many specific domains (such as law and safety). In terms of providing guidance concerning generation, collection and use of items of evidence in a specific domain, it is beneficial to consider the types of substantial content of evidence in classification. As we mentioned before, evidence in safety may come from a variety of sources, such as test, analysis, judgment, demonstration, field service, management, standard compliance, specific validation or verification, or good practice. The factors to be considered in the evaluation of items of evidence must take into account the characteristics of that specific type of evidence, usually differing from the characteristics of another type of evidence.

For example, evidence for demonstrating the satisfaction of the applicable safety requirements commonly comes from four major sources [195]:

- **Analytical evidence** (including results from simulation, hazard analysis, cause analysis, consequence analysis, behaviour modelling etc.)

- **Empirical evidence** (observation and measurement of behaviours from various types of testing, historical operation, or real practice.)

- **Adherence evidence** (adherence to standards, guidance, design rules, prescribed process, accepted best practice etc.[6])

- **Engineering judgement** (inspection, review, or expert opinion based on personal knowledge, engineering experience and creative thoughts.)

Besides the classification scheme presented above, evidence can also be classified according to the types of the safety claims being supported. In [204], evidence is categorised into three groups: evidence for requirements validation, evidence for requirements satisfaction, and evidence for requirements traceability. However, if the higher-level argument structure of a system are not decomposed with respect to the safety claims (as requirements validation, satisfaction, traceability), this type of classification of evidence is not helpful in terms of understanding, planning or selecting evidence during the development of safety cases.

In addition, new types of evidence will emerge with the advance of new methods, new objectives, new technology and new problems. The types of evidence that can be used to underpin safety claims in safety cases should be recommended as part of best practice by regulators to help the comprehension, use and management of evidence in a specific domain. However, due to the diversity and complexity of potential evidence types, a clear argument-evidence interface is needed and must be based on the clarified connotation of evidence.

4.4 Relationship between Evidence and Argument

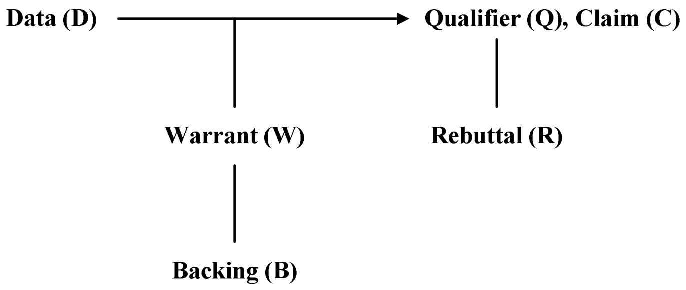
This section explains the need of a model of evidence for safety cases, through discussing the relationship of evidence and Toulmin's argument model, the inadequacy of existing view of

[6] Sometimes it is referred to as 'qualitative evidence'. But this label may confuse with other types of evidence, such as qualitative analytical evidence or qualitative judgement from experts.

evidence-argument relationship, and the limitation of current representation of evidence with structured argument notations.

4.4.1 Toulmin's Argument Structure

Much of the work on structured arguments in safety cases stems from the conceptual frame and layout of argument proposed by Toulmin. The general layout for arguments presented by Toulmin [198] (as shown in Figure 21) describes the elements that exists in arguments and their function in the argumentation process. This argument model addresses the logical representation of arguments explicitly as a rational justification rather than a formal inference according to a set of fixed mathematical principles. It provides a good foundation for the analysis and construction of many kinds of arguments, regardless of the domain in which they are applied.



Claim the statement we wish to justify;

Data/ground the fact we appeal to; the grounds or information on which our claim is based;

Warrant a statement authorising the step from data to claim is true; an inference rule;

Backing a reason for trusting the warrant;

Qualifier a term or phrase reflecting the degree to which the data supports the claim, e.g. generally, probably

Rebuttal specific circumstances in which the argument will fail to support the claim as exceptions.

Figure 21 Toulmin's Argument Model

According to Toulmin's structure, we do not attempt to establish truth through argumentation, but to establish reasonable justification for the acceptance of a claim. Much work has been developed on the basis of the original Toulmin's argument model [99]. It has been used to represent the reasoning process in a variety of disciplines, such as law, education, medicine and artificial intelligence.

The key function of Toulmin’s argument model is not to strengthen the grounds on which the argument is founded, but rather to show how to proceed from them as a starting-point to the claim. One significant contribution of Toulmin’s model is the explicit representation of the ‘warrant’ and ‘backing’ elements of an argument. Knowledge of the data and the claim alone does not necessarily convince us that the claim will be drawn from the data. A mechanism is required to work as a justification of the inference from the data to the claim. That is the function of a warrant. A backing is used to provide grounds for a warrant. Toulmin has stressed the importance of backing by stating that “*the soundness of our claims to knowledge turns on the adequacy of the arguments by which we back them*” [198].

Besides the strength of the inference rule, the credibility and acceptability of data or grounds is equally important for the soundness of argument. Justified grounds is one of the important conditions of good reasoning based on Toulmin’s model [100]. However, the notion of data/grounds is broad. It can be similar to the notion of an acceptable proposition in logic and it can also be viewed as the concept of evidence in law. In the safety engineering domain, the grounds of argument are not generally well-presented in safety cases, perhaps due to the wide variety of formats and scale of items of safety evidence. Most items of safety evidence are themselves complicated artefacts from system design, analysis or test activities. Direct references to these artefacts do not communicate clearly why they are capable of supporting a claim. The particular information from the evidence source data used as the ground for determining the truth value of a claim is not evident. In addition, the rationale of the adoption of items of evidence and the justification of the suitability of items of evidence adopted cannot fit into one single block in a notation. Therefore, thinking of the clarity issue, regarding both comprehension and representation, we need a model of safety evidence that can help us organise evidence-related information and interface safety evidence with arguments in a structured manner. This is the subject of Section 4.5.

4.4.2 A Simplified View of Evidence

In a safety case, there are usually a large number of evidence items presented in support of the argument for the top level safety claim. A simple view of the relationship between safety evidence and argument in safety cases is illustrated in Figure 22 in GSN terms.

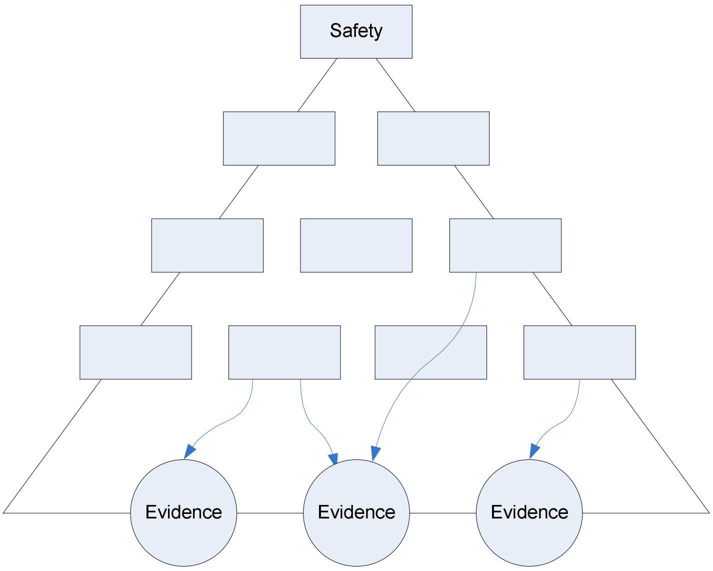


Figure 22 Simplified View of Argument-Evidence Interface

In Figure 22, the triangle depicts the overall safety case. In the upper part, there is the structured argument that consists of safety claims at various abstraction levels. At the lower part of the figure lie references to the items of safety evidence that support the safety argument. In the figure, it may seem that the relationship of argument and evidence is fairly simple, just as links that connect the references to items of evidence with the safety claims being supported.

However, the view is not so neat and simple in reality. Firstly, evidence is not only presented simply at the 'bottom' of a graphical representation, in GSN terms, as *solutions* to bottom level claims. Some evidence may also support higher level claims directly. Sometimes, evidence is needed and used as *context* to support the decomposition of safety claims. For example, the results from an aircraft-level FHA may be used for setting up the safety objectives of aircraft functional systems. Secondly, for the ultimate aim of obtaining a compelling safety case, evidence itself should be justified for its role of evidence in the context of specific safety arguments in order to establish confidence in the grounds of an argument. The justification of the evidential properties of evidence may be separate lines of argument by themselves that are associated with the argument structure presented. These backing lines of argument are also not shown (as an explicit part) in Figure 22. Finally, there is a question concerning whether the interface between argument and evidence can be represented in a unified format, because safety evidence may be of a variety of forms and from many different sources.

4.4.3 Evidence Representation in Notations

Argument and evidence can be represented in both textual and graphical forms in safety cases. Two graphical notations are currently in wide use – GSN [89] and CAE [17]. Both of them are supported by software tools, which greatly ease the management and review of structured safety cases.

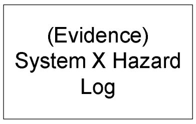
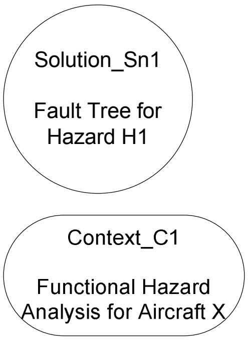
Notation	Example
Text-based notation (from [102])	Claim 1.1.1: H1 has been eliminated. Evidence 1.1.1: Formal verification
CAE [17]	
GSN [89]	 Solution_Sn1 Fault Tree for Hazard H1 Context_C1 Functional Hazard Analysis for Aircraft X

Table 4 Examples of Evidence Represented in Notations

Let us look at three examples of evidence representation in Table 4. In the textual form, the item of evidence is depicted in natural language and numbered in correspondence to the claim that it supports. In CAE [17], the item of evidence is represented by a rectangle with a description of the item of evidence. In GSN [89], an item of evidence may be represented by two types of graphical elements. It may be presented as *Solution*, which is represented by a circle, if it supports a safety goal. It may also be presented as *Context*, shown as a round-cornered rectangle, if it contextualises the decomposition of safety goals.

We can see from Table 4 that the *citation* of an item of evidence in a safety case is different from the actual source data of that item of evidence. However, it can be unclear as to how and why an item of evidence fulfils its particular usage instance, solely from such a graphical representation. We should not take for granted the content of evidence items and their 'fitness-for-usage'. A citation or reference enables us to have access to source data, but not to grasp the part of information that is embodied by the source data of evidence and is being used in the context of an argument.

Similarly, the relationships between an argument and a body of evidence can be represented in either a textual format or in a graphical format. The links between argument and evidence, if represented in a textual form, are implied by the identification numbers attached with the references or descriptions of items of evidence. In graphical forms, the relationships are always represented by directed lines between graphical symbols.

As we have mentioned in Table 4, in GSN, evidence can be represented by *Solution* or *Context*. An item of evidence as a *Solution* will connect with a *Goal* that represents a claim to be supported. The connection between them is represented by an arrowed line called *SupportedBy* (historically, also called *SolvedBy*). Figure 23 (a) illustrates this case of representation. A *Solution* only provides a reference to an item of evidence. The generation, collection and management of the source data of evidence is usually beyond the capability and responsibility of argumentation tool support. Some evidence is presented as *Context* in support of the decomposition of safety objectives. A *Context* is linked with another graphical symbol by a hollow-arrowed line called *InContextOf*. Figure 23 (b) is an example for this kind of usage of evidence.

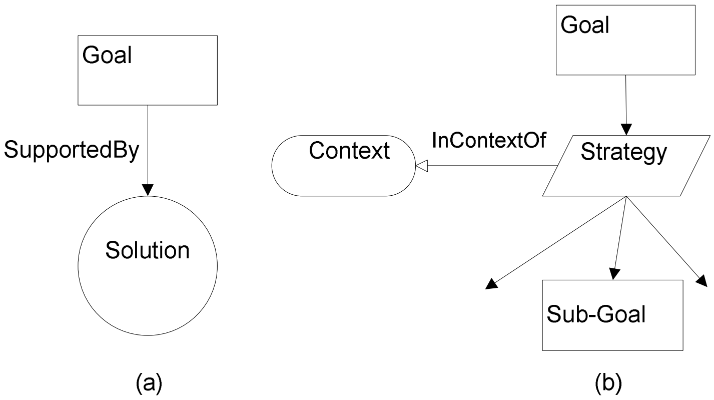


Figure 23 Representation of Argument-Evidence Relationships in GSN

The CAE notation for safety argument construction has three building blocks – *Claim*, *Argument* and *Evidence* [15, 17]. The relationship between an *Evidence* node and other argument elements – *Claim* and *Argument* nodes – is simple. The function of an item of evidence serving as context is not represented in CAE. The link between an item of evidence (represented as an *Evidence* node) with the *Claim* node to be supported can be connected directly by a linking line named as *IsEvidenceFor* or indirectly with intermediate annotation by an *Argument* node for the rationale of adopting that item of evidence. *Argument* nodes are optional [17] and may be presented if the links between *Evidence* and *Claim* nodes are not straightforward. Figure 24 presents two views of evidence-argument relationships depicted in the CAE notation. Similar to *Solution* in GSN, the *Evidence* node in CAE is also a description of the citation or reference to the source data of an item of evidence, which should not be confused with the concept of evidence presented in Section 4.2.

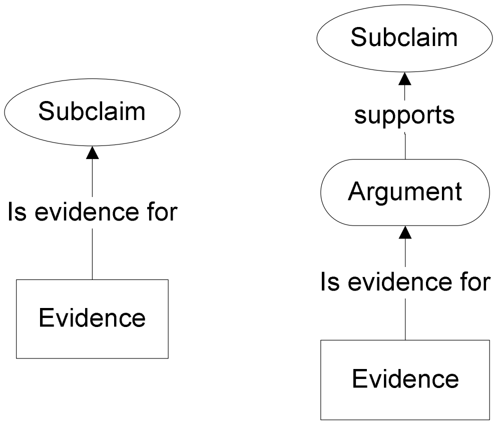


Figure 24 Representation of Argument-Evidence Relationships in CAE

Therefore, it is obvious that current representation of evidence in safety cases with graphical notations does not address the essence of items of evidence. The essential content of evidence that exhibits its evidential power is not explicitly shown or stated in the existing forms of graphical representation, but preserved by the actual data source of evidence somewhere else. If an item of evidence is complicated or information-rich, such as a safety analysis report of hundreds of pages, it would be difficult to understand and assure the logical connection between an item of evidence that is cited in a structured argument and the specific claim that is supported by that item of evidence.

4.5 A Model of Evidence (EviM)

This section defines a model of evidence (EviM) for capturing the relationship between evidence items and safety arguments. Three viewpoints that are integrated within the model are introduced in Section 4.5.1 before the presentation of EviM in Section 4.5.2. Section 4.5.3 explains the relationship of EviM and a structured argumentation model. Section 4.5.4 ~ Section 4.5.7 elaborate the interface element - ‘evidence assertion’ in EviM. Additionally, ‘Trustworthiness’ and ‘Appropriateness’, two evidential properties associated with evidence items, are explained in Section 4.5.8.

4.5.1 Three Perspectives

Based on the definition of evidence in safety cases in Section 4.2, we claim that the model of evidence in safety cases should integrate views of evidence from three distinct perspectives.

- The **content** perspective
- The **utilisation** perspective
- The **evaluation** perspective

When we talk about the content of an item of evidence, sometimes it is interpreted merely as the content of the source data of evidence that embodies the information to be used as evidence. However, it is inappropriate to neglect the role of evidence in an argument that is also part of the content of an item of evidence. First of all, different observations can be made concerning the source data of evidence depending on viewpoint. If we observe the evidence source data from the viewpoint of being an item of evidence for a particular domain safety claim, for example, the content of our concern is quite specific. Secondly, an assertion or proposition is a different concept from a data item. The evidence source data may contain a variety of data items (e.g. the various data items in CoreDMM presented in Chapter 3)[7]. An item of evidence, for its intended role within a safety case, should clearly define assertions in order to connect it with proper argument elements (in addition to the source data of evidence or references to the source data). A proposition should be clearly separated from the concepts of individuals, objects, and properties etc. that are deemed as data items in the evidence source data. Data items (from evidence source data) will imply the truth value of propositions. Propositions that are contained in an item of evidence are unique in that they are designated with a value of 'True' inherently without further supporting argument or evidence. These propositions based on evidence source data are often implicit in existing practice. In the model of evidence presented in the following section (Section 4.5.2), these propositions are explicitly addressed and presented as *evidence assertions* (to be defined and elaborated in Section 4.5.4).

The utilisation perspective is primarily concerned with the linkage of an item of evidence with its source data and its argumentation context, which must be elicited and documented clearly in safety case development. From this perspective, we aim at answering the following two questions, "Where is the evidence from?" and "Where is the evidence used?". During the development stage of a safety case lifecycle, an item of evidence must be connected with a piece of evidence source data planned at the beginning of the stage, or realised at the end of the stage. Additionally, the connections between items of evidence with argument elements (claims or links between argument elements) within a safety case must be explicitly

[7] The complexity of data items is two-fold. Firstly, the number of different types of data items. Secondly, the number of data items of a same type. Some source data of evidence may contain few data items, e.g. the prescriptive measures defined by a standard for a specific hazard.

presented. Otherwise, an item of evidence is not yet actually adopted as a part in a safety case.

The evaluation of evidence includes the evaluation of the source data of evidence and the evaluation of its usage within the context of an argument. The evaluation of the usage of evidence places emphasis on the capability and sufficiency of evidence (the evidential properties of evidence items) in terms of its function of supporting claims. We should carry out the evaluation with consideration of the specific application scenario, whereas the evaluation of the source data of evidence (e.g. a fault tree or a software testing result) can be considered without associating it with a domain claim.

The following section introduces the evidence metamodel, EviM, which integrates relevant information of evidence from the three perspectives.

4.5.2 EviM Overview

Figure 25 depicts EviM, a conceptual model of evidence represented in UML, in the context of safety cases, which stresses the connotation of *evidence*. The data elements of EviM, which place emphasis on the essential content and the role of evidence in safety cases, have been established based on the analysis of the concepts of evidence performed in Section 4.2.

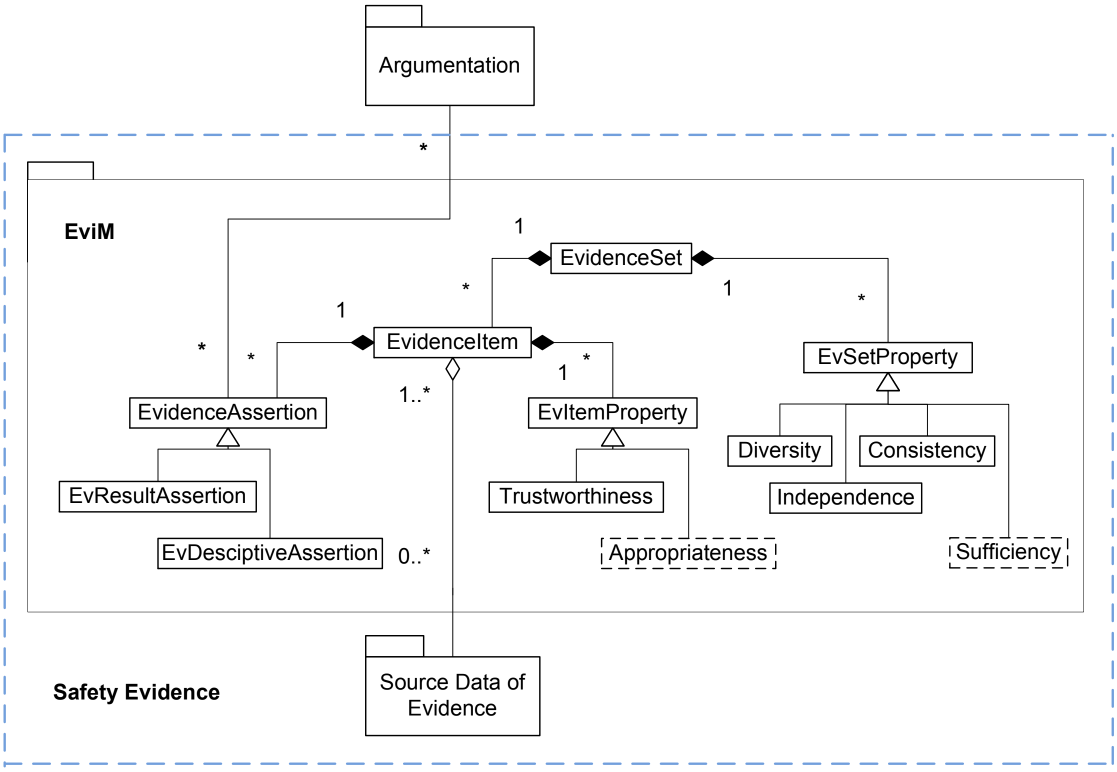


Figure 25 A Model of Evidence in Safety Cases (EviM)

EviM consists of five key elements: *EvidenceItem*, *EvidenceSet*, *EvidenceAssertion*, *EvItemProperty* and *EvSetProperty*. EviM is a model at the M1 level in the OMG meta-modelling architecture presented in Figure 11 in Chapter 3. It expands the 'Evidence' package presented in Figure 20 in Chapter 3, which describes the relationships between safety assessment artefacts and assurance meta-models. Importantly, EviM distinguishes the collective evidence set that can be used to support a particular safety case, and the items of evidence contained in that set, which themselves are composites of the evidence source data, the necessary metadata and the propositions and evidential properties.

EvidenceItem describes the items of evidence adopted or referenced in safety cases. This element is basically a container class of evidence-related information, including references to safety analysis artefacts, evidential properties for a single item of evidence, and assertions made for the information embodied within an evidence entity. *EvidenceItem* references data from safety assessment artefacts, but does not *contain* the source data. As described in Section 4.2.3, evidence should not equate to documents or analysis artefacts. However, it is popular for safety assessment artefacts to be termed 'evidence' by safety practitioners, because we have some prior knowledge of their intended usage as evidence for safety claims. But we must understand that, in fact, safety assessment artefacts are *source data* of evidence without explicitly-stated evidential roles and evidential properties.

A collection of evidence items for a safety claim or an argument module [119] can be packed up as a set of evidence items, depicted as *EvidenceSet* in Figure 25. The objects of *EvidenceSet* can possess a different set of evidential properties to be considered from the ones under concern for objects of *EvidenceItem*.

The *EvidenceAssertion* in Figure 25 represents the core propositional content of an item of evidence that is obtained from the source data of evidence. *EvidenceAssertion* is a subtype of *Claim* in ARM [156]. It is proposed specifically to clarify the usage of information embodied by an evidence entity in argumentation. *EvidenceAssertion* is further subtyped as *EvResultAssertion* and *EvDescriptiveAssertion*. The notion of *EvidenceAssertion* and its subtypes are explained in Section 4.5.4 ~ Section 4.5.7.

As shown in Figure 25, we define the properties of an item of evidence (*EvItemProperty*) and those of a set of evidence items (*EvSetProperty*) as individual classes by themselves. The reason for this is to clarify that the evidential properties are characteristics specifically concerned for *EvidenceItem* or *EvidenceSet* in argumentation context that need thorough consideration in safety case construction and reviews.

EvItemProperty and *EvSetProperty* should be obtained through the evaluation of the usage of *EvidenceItem* and *EvidenceSet* rather than the evaluation of the source data of evidence by themselves without the context of argumentation. In a compelling safety case, each individual item of evidence and its relationship with the argument presented should possess two properties – *Trustworthiness* and *Appropriateness*. Moreover, a set of evidence items should also exhibit some special properties, such as *sufficiency* (or coverage), *independence*, *diversity*, and *consistency*. These properties concern more with the interrelationships between items of evidence and the factors influencing their collective supportive capability. The properties of *EvidenceSet* have not been explored further in this thesis, except the property of *Consistency*, which is studied later in Chapter 5.

All these evidential properties, if achieved, help ensure the level of confidence we can have in the grounds of a safety argument. Two of the properties presented in Figure 25 are depicted with dashed-line rectangles, because they are in fact properties of the relationships between an item of evidence and an argument element, typically, a domain safety claim. We present them in EviM primarily for a comprehensive view of various evidential properties that are relevant to items of evidence.

EvidenceAssertion in Figure 25 is associated with elements in an argumentation model (e.g. a domain safety claim, or a relationship between two argumentation elements in ARM [156]). The explanation of relationships between *EvidenceAssertion* and argumentation elements in ARM is presented in Section 4.5.3. The source data of an object of *EvidenceItem*, comes from system development and operation, e.g. safety assessment models under study in this thesis.

We have not presented administrative data about evidence in EviM, because EviM is a conceptual model. During safety argument construction, we concern more about the metadata of the source data of evidence such as who performed the analysis and the method that was used in generating the source data of evidence, rather than the metadata associated with the application of specific items of evidence (such as who linked the source data of evidence with the claims in safety arguments, or when the source data of evidence was designated as an item of evidence for a claim). For management of evidence data at the implementation level, administrative data elements associated with items of evidence could be added to EviM.

4.5.3 Relations with ARM

The overview of ARM is presented in Section 2.7.3. The two core argument elements in ARM that interact with EviM are *Claim* and *AssertedRelationship*. *Claims* are recorded propositions within a structured argument [156]. *AssertedRelationships* are abstract associations that connect structured argument elements.

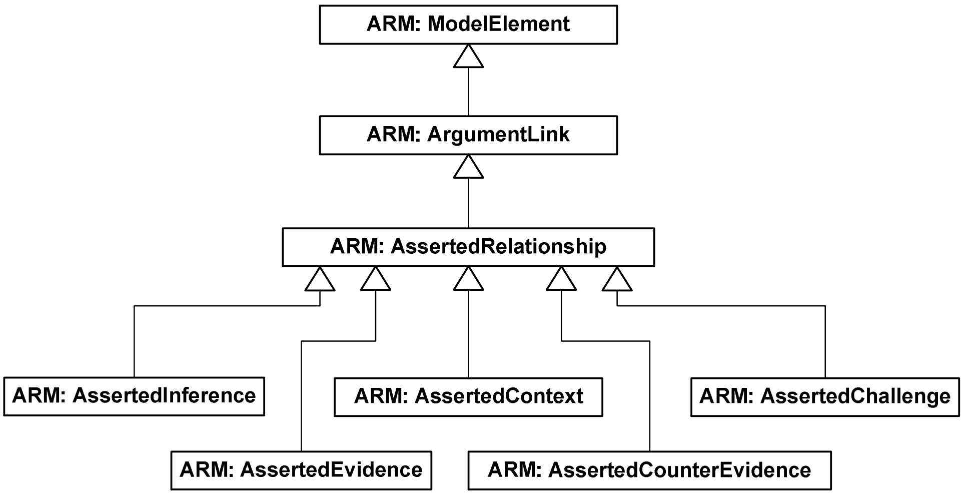


Figure 26 Asserted Relationships in ARM

There are five subtypes of *AssertedRelationship* in ARM [156], as depicted in Figure 26. The cited source data of an item of evidence is linked with other argument elements through three subtypes of *AssertedRelationships* - *AssertedEvidence*, *AssertedCounterEvidence*, *AssertedContext* (as defined in [156]).

> *"The **AssertedEvidence** association class records the declaration that one or more items of Evidence (cited by InformationItems) provides information that helps establish the truth of a Claim. It is important to note that such a declaration is itself an assertion on behalf of the user. The information (cited by an InformationItem) may provide evidence for more than one Claim"* [156].

> *"**AssertedCounterEvidence** can be used to associate evidence (cited by InformationElements) to a Claim, where this evidence is being asserted to infer that the Claim is false. It is important to note that such a declaration is itself an assertion on behalf of the user"* [156].

> *"The **AssertedContext** association class declares that the information cited by an InformationElement provides a context for the interpretation and definition of a Claim or ArgumentReasoning element"* [156].

The relations of EviM and ARM are twofold. Firstly, *EvidenceAssertion* proposed in EviM is a subtype of *Claim* in ARM. Figure 27 illustrates the relationships between *EvidenceAssertion* in EviM and *Claim* in ARM (the shaded blocks are elements in EviM; the blocks with a white background are elements in ARM).

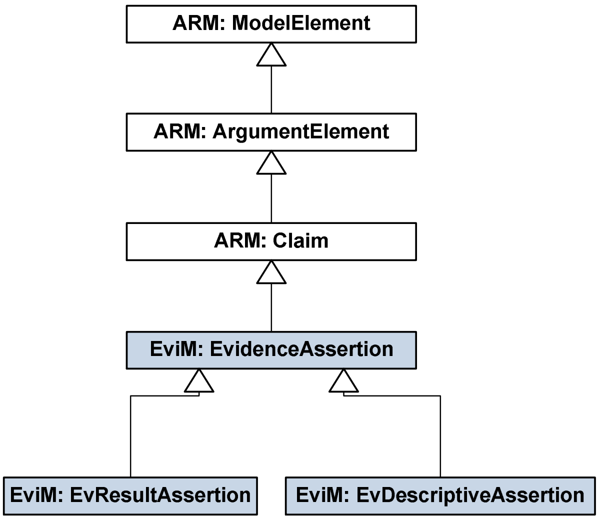


Figure 27 Evidence Assertion – a Subtype of Claim

Both *EvidenceAssertion* and *Claim* are propositions, which can be true or false in value. One prominent difference between them is the origin of the propositions. *Claim* abstracted in ARM for argumentation is the abstraction of expected claims in a problem domain, which are propositions, with their values determined or undetermined, about the real world subjects; whereas *EvidenceAssertion* in EviM is drawn from and for the source data of evidence, which are true propositions about modelled subjects on the basis of the content of the source data of evidence. We only present true propositions that can be directly[8] established from the evidence source data. The reasons for it include: a) they are part of the meaning exhibited by the evidence source data that is of our interest; b) the potential false propositions that can be associated with evidence source data are pointless and boundless. It is worth noting that *EvidenceAssertions* are components of an *EvidenceItem*, but not components of an argument.

Secondly, *EvidenceAssertion* in EviM is one of the external target elements linked with the *ArgumentElement* or *ArgumentLink* in ARM. In ARM, the source and target links associated with *ArgumentLink* are connected with the top level *ModelElement* (which is of the highest level of abstraction). This enables the powerful expression of all kinds of potential connections between argument elements of different subtypes. However, it also makes the permitted and prohibited connections between various subtypes of argument model elements less clear. Figure 28 illustrates the subtypes of *ArgumentLink* that can be connected with the

[8] 'Directness' in terms of not extrapolating beyond the nature of the source information itself.

evidence assertions of an item of evidence. Regarding the interface between evidence and argument, we can see that instances of *EvidenceItem* (citing source data of evidence outside of an ARM model) are special in that they can serve only as a *source* object of an *ArgumentLink.*

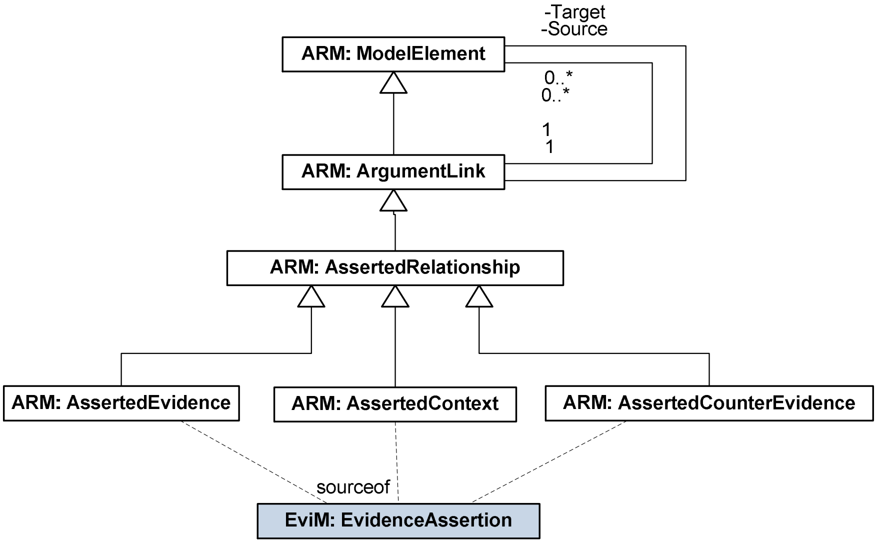


Figure 28 Argument Links between Evidence and Argument

4.5.4 Evidence Assertion

The notion of the 'evidence assertion' has been suggested in the OMG ARM [156] and GSN community standard Version 1 [89]. However, the explanation of this notion is not yet sufficient for practical application. The following sections explain this concept further.

An evidence assertion is a statement that we can take as a *true* proposition according to the content of the source data of evidence. Representing evidence assertions drawn from source data explicitly can provide us a clear view of what is apparent from an item of evidence. Evidence assertions are not intended to record *judgements* about the source data of evidence, but instead document propositions that can be established *directly from* the information embodied by the source data of evidence. The truth value of an evidence assertion is not intended to be debatable. However, *true* evidence assertions do not directly mean that the corresponding *domain claims* are true, unless the trustworthiness of items of evidence and the appropriateness of claim-evidence relationships are justified.

The description of *evidence assertion* in ARM includes the following key points [156]:

- The nature of an evidence assertion is a claim.
- An evidence assertion is minimal and does not need supporting argumentation.

- An evidence assertion is the interface element to integrate argument and evidence.

Based on these key points, we propose the definition of Evidence Assertion in this thesis as:

> *An Evidence Assertion is a minimal proposition that describes straightforward 'factual information' concerning an item of evidence. It does not need support from further arguments or evidence and it directly concerns the source data of an item of evidence without involving subjective judgment.*

As a specific type of claim, an evidence assertion is unique in its source and function. An evidence assertion is drawn directly from the content of the source data of evidence. It is a true proposition according to what is presented in the source data of evidence that is about subjects under our concern. An evidence assertion can be used as grounds or context for a domain safety claim.

Evidence assertions should be distinguished from domain safety claims in arguments. A domain safety claim is what we want to state in the problem 'application' domain; it is a statement concerning the subjects (or concerns) in a real problem domain. Unlike an evidence assertion, the truth of a domain claim is *uncertain* unless supporting argument and evidence are provided. Domain safety claims may form a hierarchy of claims which represent how higher level safety goals are decomposed into more concrete ones; they can be supported by either claims or evidence. By contrast, an evidence assertion is a propositional statement on the subjects in an item of evidence that model or represent subjects in the real problem domain; it does not need any further support, either from claims or from evidence.

An evidence assertion differs from the data items contained in the source. It is a claim that is *about* what is embodied by those data items. The true or false value of an evidence assertion is not determined by the facts of a problem domain in reality, but endowed by the facts of presence or absence of specific data items in the source data of evidence. Our confidence in terms of whether an evidence assertion can infer the True value of a domain claim in the real problem domain depends on the trustworthiness of an evidence item and the appropriateness of its usage.

There are two subtypes of evidence assertions:

- Evidence Result Assertion
- Evidence Descriptive Assertion

The Evidence Result Assertion is described in Section 4.5.5; the Evidence Descriptive Assertion is explained in Section 4.5.6.

4.5.5 Evidence Result Assertion

An evidence result assertion is a proposition that can be made from the source data of evidence and can be used to support domain claims in safety arguments. It answers the question, "What does an item of evidence say?". For example, we may use a fault tree as an item of evidence. Then the *EvidenceResultAssertion* contained in this item of evidence could be 'The probability of modelled EventX is 1.0×10^{-4}".

Formulating evidence result assertions from items of evidence directly has two advantages.

- It may help to clarify the role or function of potential items of evidence as early as possible.
- It may also ease the management of items of evidence in parallel to the management of safety cases.

EvidenceResultAssertion can serve as the 'data' element in Toulmin's model directly. It is the starting-point of a line of safety argument. The subject of an *EvidenceResultAssertion* addresses some aspect of the source data of evidence which represents the subject in the problem domain (e.g. a modelled subject in a model). For example, the principal noun of a domain claim may be "the probability of an undesired event E_x"; E_x is the undesired event in the problem domain. By contrast, the principal noun of a corresponding evidence result assertion of an item of evidence (e.g. a fault tree) may be "the probability of a modelled undesired event E_m"; E_m is the top event in that fault tree that *models* or *represents* E_x. In reality, *EvidenceResultAssertion* of safety assessment evidence may have features or styles determined by the types of safety evidence (e.g. the substantial analysis outputs of a safety analysis technique). As the definition of *evidence assertion* indicated, the Boolean value of an evidence result assertion is *true* if the item of evidence has been generated and is in use or *expected to be true* if the item of evidence is planned to be available.

There are two ways through which we can identify potential features of evidence result assertions. The first way is to examine the lower level domain safety claims that are presented in a safety case or a safety case pattern. For example, if there was a goal in a safety case, depicted as "the probability of a failure condition X does not exceed 1×10^{-6} per flight hour", then the evidence result assertion being expected would be that "the probability of the modelled failure condition X is $\{P_x\}$ per flight hour". The $\{P_x\}$ in the expected evidence result assertion should be a number less than 1×10^{-6}. They are close in terms of the format and the subject of the expression.

The example above is almost self-evident; however, it is not always effective to identify, in this way, the potential features of evidence assertions in the domains that are not rich in prescriptive requirements and good practice. Because there could be a trap that might lead us to unrealistic expectation of the forms of evidence result assertions desired, when a domain safety claim contained in a safety case has not been well-decomposed to a proper level.

For example, CS25.1309 (b) [68] has set safety goals for failure conditions that are classified as Catastrophic. It can be easily derived from CS25.1309 (b) that two forms of evidence result assertions are expected for potential items of evidence to be used for justification of the sufficient control of Catastrophic failure conditions:

- "The probability of Catastrophic Failure Condition X presented in {Evi} is {P_x} per flight hour", where {Evi} is an item of evidence and {P_x} is a number less than 1×10^{-9}.
- "There is no single point failure identified in {Evi} that can lead to the Catastrophic Failure Condition X", where {Evi} is an item of evidence.

If a domain safety claim in a safety case under study was at a higher abstraction level, such as 'Equipment X is fail safe' or 'The configuration logic is acceptably fault free', the granularity of these domain safety claims needs to be refined before reasonable features of potential evidence result assertions can be derived. From engineering practice, we understand that it is more plausible to read out whether a failure scenario has been considered and how it is considered from an item of evidence rather than whether a domain subject of concern is 'fault free' or 'fail safe'. For safety assessment evidence, normally, the subject of an evidence result assertion is within a range of permitted substance elements (e.g. the probability or MCSs of an undesired event) and construction elements that are contained in a safety assessment model. The domain claims with concepts (e.g. 'fail safe') that resist being directly modelled as a part of a model, should be decomposed into lower level sub-goals that *can* be supported directly by items of evidence. A domain claim with safety concepts or features of a system that cannot be addressed by a model directly is not permitted to be supported directly by evidence result assertions.

The second way of identifying evidence result assertions is to observe available evidence presently in use or recommended by standards and guidance. For example, we can ask ourselves questions such as 'what do we expect to learn from a specific safety analysis (e.g. FTA)'?, 'what do we expect to learn from a specific test'?, 'what can we assert about the safety characteristic of a domain subject according to an item of evidence'?. The answers should be a concrete statement that is about a modelled subject (which is of the corresponding subject in the problem domain) and is related to the purpose of generating that item of evidence. The forms of evidence result assertions formulated in this way rely upon our understanding of the purpose of various types of evidence and their potential use in safety cases. For example, Table 5 presents the main forms of evidence result assertions that we can make from the safety analyses as recommended by the safety assessment process in ARP 4754A.

Types of Safety Analysis	Examples of Evidence Result Assertion
FHA	• Failure condition FC_x modelled in FHA_a is a Catastrophic failure condition of Aircraft AC (according to an aircraft level FHA model- FHA_a)
FTA	• The probability of failure condition FC_x modelled in FTA_b is P_x. • Failure condition FC_x modelled in FTA_b was caused by more than one failure event in FTA_b. (according to a fault tree model - FTA_b)
FMEA	• Component X modelled in $FMEA_c$ has three failure modes. • Failure Mode A of Component X modelled in $FMEA_c$ may lead to Failure Effect B. (according to an FMEA - $FMEA_c$)
Markov	• The probability of sub-system X being in an operational state, modelled in MAR_d, is P_x. (according to a Markov model- MAR_d)
CMA	• Failure event A and failure event B are independent in CMA_e. (according to a common model analysis- CMA_e)
ZSA	• There is a hazardous Failure Condition FC_y in Zone X in ZSA_f. (identified in a zonal safety analysis - ZSA_f).
PRA (e.g. lightning)	• The lightning interaction modelled in CM_g (between the skin and structure of Aircraft AC) is acceptably safe. (according to a Computational Model - CM_g)

Table 5 Examples of Evidence Result Assertion

But we must keep in mind that matching and re-examining evidence result assertions are indispensable tasks when we adopt an item of evidence in a safety case. Some evidence result assertions that are drawn before integrating evidence with arguments might be unsuitable for a certain usage; some other evidence result assertions might have been neglected before the integration. Explicit documentation of the evidence result assertions of an item of evidence, especially the one used as direct evidence for a domain safety claim, is helpful for accumulating both the existing and the expected usage of the evidence source data, which can provide an easy understanding of the evidence-argument relationship and help facilitate the potential reuse of evidence items in some new context. For the safety assessment evidence under study in this thesis, the evidence result assertions should be based on the instances of *SubstanceElement*s of safety analyses (as depicted in CoreDMM presented in Section 3.4.1).

The elicitation of evidence result assertions may present some extra work during safety case development. However, it is important to make these result assertions as clear as possible according to the 'facts' conveyed by an item of evidence. It is dangerous to support a domain safety claim with an item of evidence that is much less capable and effective than that which is needed. The gap between what we can say according to the source data of evidence and what a domain safety claim is about, if unacknowledged or unresolved, could undermine the overall confidence in safety cases significantly. In existing practice, we commonly are unaware of such gaps that are implicit with evidence result assertions unstated.

In graphical representation of structured safety arguments, it has not been stipulated as necessary to explicitly present evidence result assertions as concrete argument components with a symbol. However, it is necessary to understand the role of an evidence result assertion within the graphical view. In a safety case lifecycle, evidence result assertions should be included as a required data item for every item of evidence. Figure 29 (a) depicts the common view of evidence-claim interfaces (in GSN terms) in existing practice. The *Solution* cites the '*Evidence*' for the *Claim* in the pictorial view. Figure 29 (b) presents the functional position of an evidence result assertion that links an item of evidence with a domain safety claim. The argument link L_{a1} is broken down into three new elements, L_{b1}, a *goal* depicting an evidence result assertion, and L_{b2}. It can be clear to examine the relationship between a claim and an item of evidence in two steps: 'does the item of evidence contain the expected form of evidence result assertions required for support of the claim?' and 'could the evidence result assertion of the item of evidence infer the truth or falsity of the claim?'. Figure 29 (c) presents the position of an evidence result assertion in a common graphical view as Figure 29 (a) illustrated. The link between a claim and a solution, L_{c1}, is unchanged from L_{a1}. However, as we know, the *Solution* is only a reference placeholder, which possesses no evidential power logically. It is the evidence result assertion of the item of evidence being referenced that provides the support to the claim. The L_{b2} in Figure 29 (b) is a relation that does not need support or justification (according to our definition of evidence assertions). So to be succinct in a graphical view, the evidence-claim relationship can be represented as it were in existing practice. But the evidence result assertion (as it is used in the fashion illustrated in Figure 29 (c)) should be explicit and accessible as a part of the item of evidence being cited.

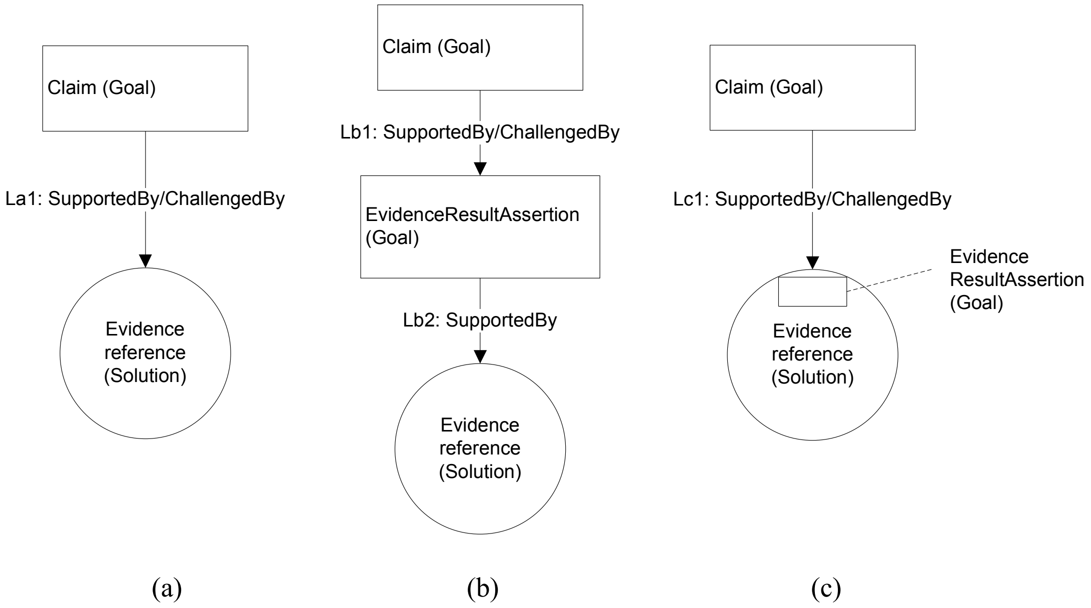

Claim (Goal)
La1: SupportedBy/ChallengedBy
Evidence reference (Solution)
Claim (Goal)
Lb1: SupportedBy/ChallengedBy
EvidenceResultAssertion (Goal)
Lb2: SupportedBy
Evidence reference (Solution)
Claim (Goal)
Lc1: SupportedBy/ChallengedBy
Evidence ResultAssertion (Goal)
Evidence reference (Solution)
(a)
(b)
(c)

Figure 29 Role of Evidence Result Assertion Presented in GSN

For the discussion so far, we have viewed evidence result assertions in the context of citing the source data of evidence to support or challenge a domain safety claim. As described in Section 4.4.2, the source data of evidence items is sometimes used as the context of argument elements. In this situation, in fact, evidence result assertions are not obligatory information elements of the application scenario. Because the source data of an item of evidence (e.g. a FMEA model), is used for providing raw data (e.g. a list of failure modes) to assist the generation or decomposition of safety goals, rather than be used for inferring the truth or falsity of a domain safety claim. However, the concept of evidence assertions is still important. In this situation, the evidence assertions of an item of evidence still exist, but not as the interface elements that connect the item of evidence with other structured argument elements in a primary safety argument. Each item of evidence can play a multiple role in safety cases, as context or supporting evidence, or both.

4.5.6 Evidence Descriptive Assertion

Alongside evidence result assertions, there are also other assertions we can make according to (and about) the content of the source data of an item of evidence. *Evidence descriptive assertions* are propositions that describe an item of evidence but that cannot be directly observed *from* the source data of an item of evidence. They are not used to support domain safety claims, but for providing support for the confidence argument associated with the primary safety argument elements (e.g. the backing argument that supports the applicability of a fault tree that is used as direct evidence in a safety case).

Evidence descriptive assertions are statements about the source data of evidence or the process of which the source data of evidence is generated. The subject of an evidence descriptive assertion can be a wide variety of things. In reality, it is impossible and unnecessary to try to elicit them completely. Instead, they should be formulated according to the details required by the backing arguments of the safety case. Typically, we are concerned of the evidence descriptive assertions that address factors that may influence the confidence in the usage of evidence. For example, the modeller, the tool, the method (the contributing factors for 'wrong' models as depicted in Section 3.3.2) of an item of safety assessment evidence should have corresponding evidence descriptive assertions elicited. These evidence descriptive assertions can help present factual information that is necessary for the evaluation of the trustworthiness of an item of evidence and the appropriateness of the usage of that item of evidence in a safety case. Evidence descriptive assertions can introduce clues and facts that help us to make decisions during evidence evaluation. For example, consider again a fault tree (FT_x) as an item of evidence. Two examples of evidence descriptive assertions of the fault tree (FT_x) are – "The repair events are not considered in FT_x" or "Operator errors have been considered in FT_x". The normal metadata (for example, the metadata depicted in CoreDMM in Section 3.4.1) from the source data of an item of evidence can be addressed by evidence descriptive assertions if needed, e.g. "FT_x is constructed by Engineer Y" or "FT_x is constructed with FaultTree+ Tool".

An item of evidence may have many evidence descriptive assertions. It is difficult to enumerate all potential descriptive assertions completely and it is unreasonable to ask for all details without a focus. For safety assessment models under study in this thesis, we observe some common contextual factors that are shared in the backing of safety assessment evidence, e.g. the scope of an item of evidence, the administrative metadata of an item of evidence. Table 6 presents some typical types of evidence descriptive assertions we can make from safety analyses.

Subjects of Evidence Descriptive Assertion	Examples of Evidence Descriptive Assertion
Scope	• Human factors are considered in Evidence Evi • System component C_x is considered in Evidence Evi
Modeller	• Evidence Evi is created by Engineer X.
Limitation	• Repair events are not considered in Evidence Evi • Timing issues are not considered in Evidence Evi
Tool	• Evidence Evi is generated with $Tool_x$
Data source	• The essential failure data used in Evidence Evi is from Handbook H_x

Table 6 Examples of Evidence Descriptive Assertion

4.5.7 Eliciting Evidence Assertions

In system safety processes, evidence assertions can be elicited either with the generation of the source data of evidence or with the adoption of an item of evidence within a safety case. The important thing is to take evidence assertions into account properly in order to demonstrate our understanding of various items of evidence and their application context, to ease the integration of evidence with argument, and to support the development of confidence argument associated with evidence.

Based on the common understanding of typical outputs of safety engineering activities and the features of evidence in typical public safety cases, we propose a classification scheme of typical evidence assertions. Figure 30 illustrates the typology of evidence assertion that should be considered in practice. It is not intended to be an exhaustive list, but can act as an aid to thinking, in particular, guiding the formulation of evidence assertions in terms of the common subjects they might address.

Evidence result assertions are classified into two classes – quantitative result assertions and qualitative result assertions. Quantitative result assertions are statements based upon safety analysis results that are in a numerical manner. Qualitative result assertions are statements based upon safety analysis results addressing the demonstration of qualitative safety features (e.g. levels of redundancy).

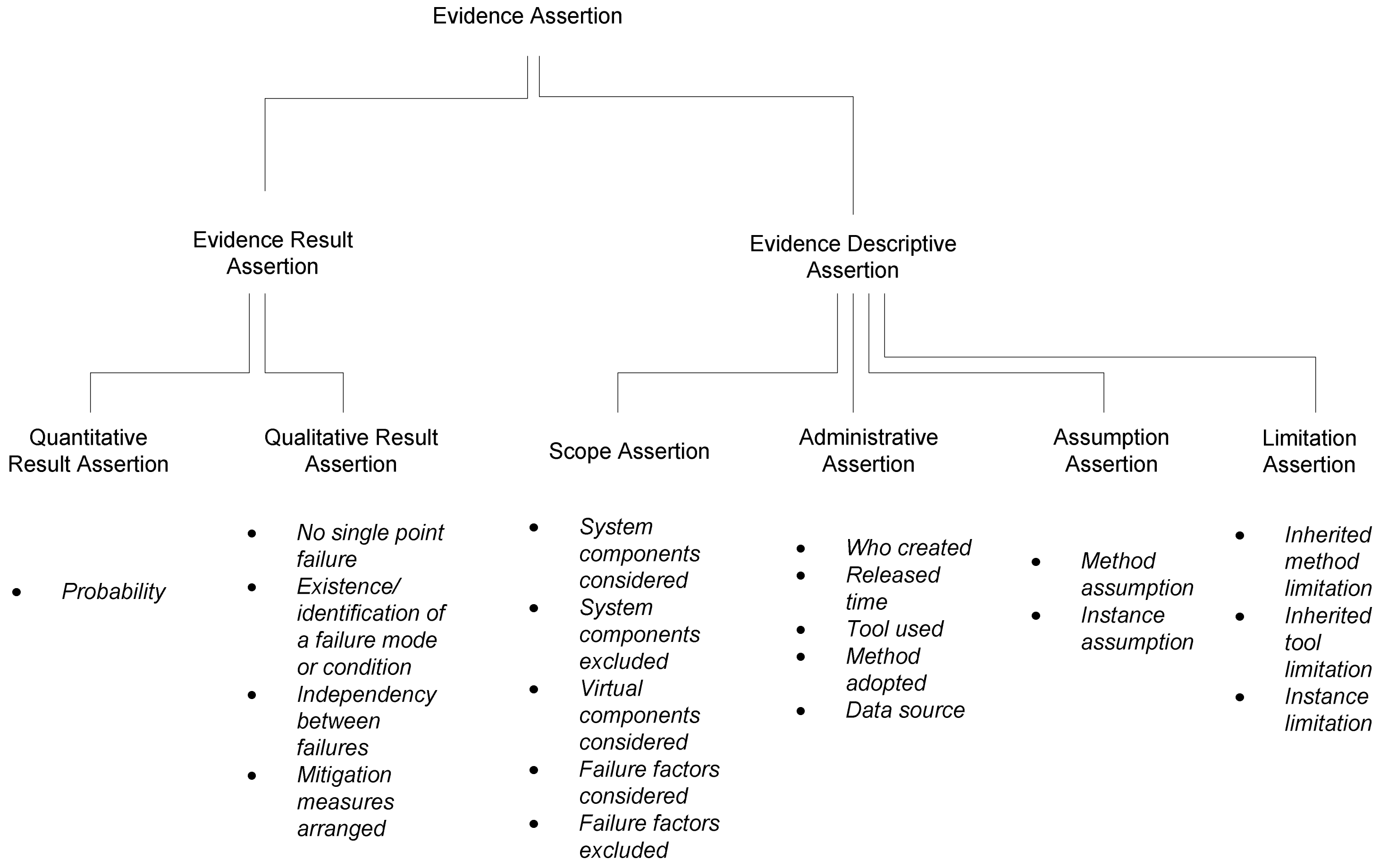

Evidence Assertion
Evidence Result Assertion
Evidence Descriptive Assertion
Quantitative Result Assertion
Qualitative Result Assertion
Scope Assertion
Administrative Assertion
Assumption Assertion
Limitation Assertion
• Probability
• No single point failure
• Existence/ identification of a failure mode or condition
• Independency between failures
• Mitigation measures arranged
• System components considered
• System components excluded
• Virtual components considered
• Failure factors considered
• Failure factors excluded
• Who created
• Released time
• Tool used
• Method adopted
• Data source
• Method assumption
• Instance assumption
• Inherited method limitation
• Inherited tool limitation
• Instance limitation

Figure 30 A Typology of Evidence Assertions

Evidence descriptive assertions are classified into four classes – scope assertions, administrative assertions, assumption assertions and limitation assertions. Some examples are given in Figure 30 for each type of descriptive assertion. While eliciting evidence assertions for a specific item of evidence, users need to consider the types of evidence assertions (as recommended in Figure 30) for inspiration and to concretize the content of evidence assertions according to the feature of the item of evidence and its specific application context. For example, a fault tree model as an item of evidence will not have evidence result assertions on 'independency between failure events'.

EvidenceAssertion is a core component of EviM. It is useful for facilitating the application and justification of evidence in safety case development. Three questions should be considered in the process of evidence selection and justification [96].

> 1. *"Is the **type** of evidence capable of supporting the safety claim?"*
>
> 2. *"Is the particular **instance** of that type of evidence capable of supporting the safety claim?"*
>
> 3. *"Can the instance of that type of evidence be **trusted** to deliver the expected capability?"*

The two subtypes of *EvidenceAssertion* can help in the answers to these questions. The form of the evidence result assertion of an instance of a type of evidence should meet the need of a domain safety claim. The content of the evidence result assertion of an instance of a specific type of evidence determines whether the instance of evidence is supportive. The evidence descriptive assertions (e.g. one associated with assumptions of a model) of an instance of the specific type of evidence constrain whether the supportive relationship holds for the domain safety claim. Furthermore, the evidence descriptive assertions associated with the generation of the source data of the evidence are useful for determining the trustworthiness of the specific item of evidence.

4.5.8 Trustworthiness and Appropriateness

Trustworthiness, in social, political and economic contexts, intends to establish and maintain cooperation [93]. As a desired property of evidence in safety cases, trustworthiness *depicts whether we can have our belief in the content of an item of evidence as what is said by the source data of evidence*. It is affected by many factors [204] such as: 'bugs' in the item of evidence presented, the rigour of review, the qualification of a tool adopted, the experience and competence of the personnel.

Trustworthiness is a property of evidence that can be evaluated in its own right for the information embodied within the source data of evidence. It represents a degree of confidence that we have for the information embodied by items of evidence – in other words, whether we can believe that the evidence result assertion of an item of evidence is a true proposition. We argue that trustworthiness is based on both the external grounds and the internal grounds of an item of evidence.

The external grounds for trustworthiness (of safety assessment evidence) lies in the quality of process elements associated with the generation of an item of evidence, such as the qualification of modelling tools, the competency and knowledge of personnel, and the capability of an analysis technique. The quality of the process elements of a safety assessment activity may influence the overall number of systematic and random flaws contained in the outputs of a modelling process (e.g. a tool may introduce a computational error; a technique may exclude consideration of a specific failure mechanism; inexperienced modeller may introduce more wrong input data) . The trustworthiness of an item of evidence originated from the external grounds will imply our belief in that the overall number of potential flaws within the source data of an item of evidence is low.

The internal grounds of trustworthiness (of safety assessment evidence) lies in the rigour of scrutiny of the construction and substance elements of the source data of evidence, rather than reviews of the process elements contained in the source data of evidence. If the source data of evidence has been rigorously examined for potential representation and understanding flaws against our knowable truth in the real world and the result is positive (which means that only a few flaws identified within the declared boundary of a model), we may claim that the item of evidence is trustworthy. The internal grounds are distinct from the external grounds. Trustworthiness based on the internal grounds will imply our certainty in the absence or scarcity of errors or flaws of concrete types that have been checked in reviews. Trustworthiness based on the internal grounds is and must be claimed within the declared boundary of model instances. ‘Bugs’ identified in the review of the source data of evidence may damage trustworthiness completely; whereas weak external grounds can only undermine trustworthiness to some degree. In addition, the internal grounds for the trustworthiness of evidence also require further trustworthiness regarding the scrutiny activity and the scrutiny results regarding an item of evidence.

Appropriateness is a property associated with the links between different argument elements. As a property of these links, appropriateness covers a varied set of concerns, depending on the (sub)type of argument links. For an *AssertedEvidence* link between an item of evidence and a domain claim, appropriateness depicts the suitability (relevance and support) of an item of evidence to uphold the declared relationship with a specific argument element. Appropriateness of asserted evidence links will render the acceptable true/false values of *domain* claims. When the safety argument is constructed at the early stages of a project, the appropriateness property of an *AssertedEvidence* link is not a prominent issue[9], since we can always expect that there are one or more items of evidence which will provide sufficient support to the domain claims. We can present the links and the desired evidence items for them in the structured argument diagram. The desired evidence items, which are placeholders for future real instances, must be instantiated later when real items of evidence are completed and released formally. When we instantiate a citation of a desired evidence with a real instance of evidence, the appropriateness of the item of evidence must be carefully reconsidered.

Sometimes, appropriateness, as a property of the links between argument elements, has been misunderstood as a property of the class or objects of *EvidenceItem*. We represent this property in EviM with a dashed-line rectangle, to indicate that this property does not belong to *EvidenceItem* and must be evaluated in context of three concrete objects (an item of evidence, a domain claim, and the relationship between them).

There is no proportional relationship between the property of appropriateness and the property of trustworthiness. A trustworthy item of evidence may be inappropriate for supporting a specific claim; untrustworthy evidence should not be adopted even if it looks appropriate in the context of an argument structure. Even if an evidence result assertion supports a domain claim, the evidence-claim relationship may or may not be appropriate. If the declared validity boundary of an item of evidence (e.g. the scope, limitations or assumptions of safety assessment models required by CoreDMM) is sufficiently elicited and accepted for the usage instance of the item of evidence, the declared evidence-claim relationship may be appropriate.

[9] If there are COTS (Commercial-Off-The-Shelf) components or legacy parts adopted in the project, we may examine instantly the appropriateness of *AssertedEvidence* links in a safety case, which are associated with those components, if the items of evidence cited is available.

4.6 Relationship between Evidence and Confidence

4.6.1 Confidence in Safety Cases

The definition of *confidence* in Oxford Dictionaries is “the state of feeling certain about the truth of something” [2]. In safety domain, our confidence in safety may relate to the truth value of a claim, or the occurrence of an event, or the existence of a state. Confidence is not objective. It is dependent on what we know and how we think. However, even though confidence is a subjective issue, it is desirable and necessary to build it up systematically and to demonstrate it explicitly, rather than leave it unmanaged or take it for granted blindly.

It is increasingly recognized that we need justified confidence in system safety demonstrated in a safety case. In order to understand the contribution of safety assessment evidence to the overall confidence in system safety, we will start with an investigation of various factors on which we can establish our confidence. Figure 31 depicts a framework of confidence in safety cases. It decomposes our view of confidence in safety cases into two parts: the confidence established on safety cases processes and the confidence established on the output of a specific safety case process.

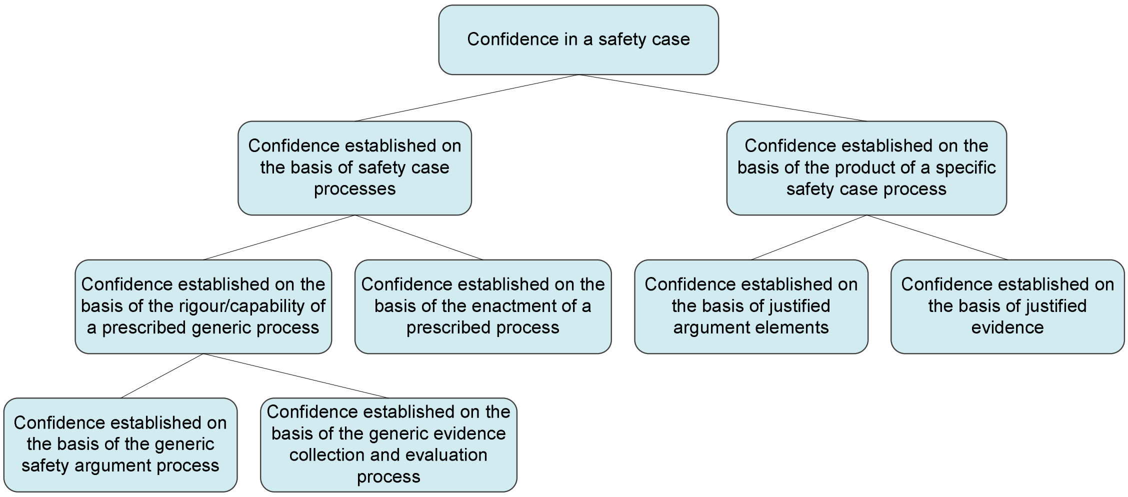

Confidence in a safety case
Confidence established on the basis of safety case processes
Confidence established on the basis of the product of a specific safety case process
Confidence established on the basis of the rigour/capability of a prescribed generic process
Confidence established on the basis of the enactment of a prescribed process
Confidence established on the basis of justified argument elements
Confidence established on the basis of justified evidence
Confidence established on the basis of the generic safety argument process
Confidence established on the basis of the generic evidence collection and evaluation process

Figure 31 Framework of Confidence in Safety Cases

The confidence that originates from processes can be divided into two types: confidence obtained from the rigour or capability of the prescribed generic process (e.g. whether a systematic argument construction process is employed in support of the argument development) and confidence obtained from the proper implementation or enactment of the prescribed process (e.g. whether the personnel implementing the argument construction are competent and experienced in the domain). Furthermore, confidence based on generic process capability involves two parts: the rigour or power of the argument construction process and the rigour or robustness of the evidence collection and evaluation process.

Biased selection of evidence or insufficient evaluation of evidential properties in the argumentation context will undermine our confidence in the safety case generated. In Chapter 6, we will introduce a more rigorous argument construction process to support the construction of compelling safety cases. From the process perspective, we can only have the confidence in the overall quality of a safety case, but appeal to *general* processes cannot guarantee or even justify the appropriateness of a specific part of an argument branch or the trustworthiness of a specific item of evidence.

Confidence in a system safety case may also be established on the basis of argument structures and a body of evidence (the content of a safety case), which are the artefact generated from a safety case development process in a system lifecycle. Through the evaluation of the detailed content of a safety case, including the claims and relationships in arguments and the body of evidence individually and collectively, we can obtain more confidence in addition to the confidence established on safety cases. Justifying evidence is part of the evaluation of a safety case artefact.

In practice, we suggest establishing confidence in safety cases from both the process perspective and the product perspective, because the two aspects are complementary, but not substitutes for each other. The evaluation based on the content should have more priority if the time and resources are permitted in the project lifecycle.

Fundamentally, confidence in the safety argued by a safety case is grounded on the quality (or validity) of the safety case itself. DS 00-56 [148] recommends “validated safety cases”, which requires rigorous scrutiny of both safety argument and evidence. If we deem a primary safety argument and associated evidence to be the source on which our confidence in the system safety demonstrated relies, we need to consider confidence in the following three aspects in the confidence arguments:

- Confidence in the strength of the primary safety argument structure
- Confidence in the appropriateness of the use of the various items of evidence
- Confidence in the trustworthiness of a body of supporting safety evidence

Where we say the primary safety argument structure is strong, it means the set of links (asserted inferences, asserted contexts) between argument elements are acceptable and sufficient to render the higher level claim from lower level ones. The argument structure can be evaluated initially with the assumptions that the end or ground level domain claims are all True. The ‘true’ or ‘false’ values of ground domain claims are determined later, influenced by the appropriateness of asserted evidence relationships in context of the specific argument. The trustworthiness of evidence depicts the extent to which we believe the information or propositions that are said by an item of evidence. The justification of safety assessment models as evidence in safety cases will be addressed in Chapter 6.

4.6.2 Importance of Evidence towards the Establishment of Confidence

We can learn from the confidence framework presented in Figure 31 that sufficient confidence in a safety case must be established on the basis of a body of adequate evidence. Our aim is to demonstrate in a safety case that the top level safety claim has been achieved. A sound and compelling safety case, ultimately, must have true premises with a high degree of confidence. The 'truth' value of the hierarchy of higher-level safety claims comes from the appropriateness of all asserted argument linking elements in a safety argument and the truth of the leaf-level domain safety claims. Importantly, the confidence in the true top-level domain safety claim is grounded on the support provided by the body of evidence connected through the primary safety argument. The origin of confidence in safety, therefore, finally settles down onto the trustworthiness of evidence items and the appropriateness of the usage of these evidence items.

In addition, considering and citing the source data of evidence properly is a vital part of safety case construction. Knowing how to handle and consider the different roles (e.g. supporting evidence or context) that are played by various items of evidence is important in an argument construction process. Although we address only the positive role of evidence (as supporting evidence) in this chapter, we will describe how to consider the potential negative role of some items of evidence in Chapter 6.

Safety cases can be viewed as a holistic model of system safety which synthesise many different forms of evidence, including safety assessment models. The desire to manage the issue of confidence in safety cases implies that we devote some resource to assuring the trustworthiness of evidence and the appropriateness of the usage of evidence and the task of incorporating counter evidence in safety cases. As a result, establishing confidence in safety assessment evidence is further studied from three aspects: the rigorous scrutiny of items of evidence, especially on cross-model inconsistency (in Chapter 5), the comprehensive exploration and consideration of counter evidence in safety case construction (in Chapter 6), and the structured justification of evidence items, in particular, qualitative safety assessment models (in Chapter 6).

4.7 Summary

In this chapter, we have studied some relevant definitions and elaborated the concept of evidence in the context of safety cases. A conceptual model of evidence (EviM) is proposed for the purpose of explicit integration of the source data of evidence and safety argument. EviM highlights the propositional content of an item of evidence and the evidential properties associated with evidence in any argumentation context. The notion of the 'evidence assertion', the interface element between argument and items of evidence, is described and illustrated with examples. EviM will help in explicitly considering the content, utilisation and evaluation of evidence and will facilitate more rigorous application and justification of safety evidence in safety cases. In addition, the confidence issue in safety cases is also discussed and the relationship of evidence with confidence is clarified and highlighted. In this chapter, evidence is primarily examined only from the perspective of its positive role in safety argument as *supporting* evidence. In Chapter 6, its potential role as counter evidence is discussed and considered in argument construction and argument patterns.

5 Managing Safety Model Inconsistency

5.1 Introduction

As explained in Section 2.6.2, evaluation of safety analysis is difficult and with insufficient practice. In this chapter, we focus on model consistency as a means of evaluating safety assessment models, a common type of safety evidence in safety cases. The issue of model inconsistency among safety assessment models is a pervasive problem throughout the development of modern complex systems. The consistency of models is easy to require but difficult to confirm. Safety standards normally require consistency in safety requirements and safety analysis, but with little further guidance of the notion and few recommended measures to examine consistency [194]. Most of the existing investigation of consistency among safety assessment models is performed informally through periodical reviews of safety assessment activities, which rely heavily on expert knowledge and experience[10]. In these reviews, the expected consistency relationships employed are usually implicit and the implementation process of consistency-checking is opaque. With this 'black-box' view of consistency between models, it is difficult to claim and to persuade others that the safety assessment models being used as evidence in safety cases are consistent. Moreover, the opaque view may hinder the accumulation and dissemination of the domain knowledge and experience on how to evaluate model inconsistency.

In this chapter, we present a structured method to managing inconsistency across safety assessment models. The chapter starts with the clarification of the meaning and classification of model consistency in system safety. Then typical consistency relationships between safety assessment models are analyzed. To reduce the informality and implicitness of consistency analysis in practice, the method defines explicitly the information requirement and the detailed steps of inconsistency analysis that can bring more transparency and structure to the model comparison process. The justification and utilisation of the inconsistency analysis itself are also considered in the method.

[10] The current state of consistency analysis between safety assessment models was obtained through personal communication with safety experts during MISSA project meetings.

5.2 The Territory of Consistency

5.2.1 Range of Meaning

The usage of the term of *consistency* varies significantly according to its context. Regulatory bodies have acknowledged the problem of inconsistent safety analyses in standards (e.g. [148, 180, 181]). However, most of the standards pinpoint only some specific scenarios required for consistency. In DS 00-56 [148], the “consistency of assumptions about operating procedures of risk classification” is required and contractors must “maintain consistency between safety-related documentation and the system configuration”. In ARP 4761 [181], it is stated that the “wording of failure effects need to be checked for consistency”. Whereas ARP 4754A [180] requires “consistency between the requirement set”, “consistency between functional hazard assessment results”, “common naming conventions” for events in safety analysis, and “consistency of analysis methods in the verification of safety requirements”. In practice, the occurrences of the term ‘consistency’ are linked with diverse subjects such as recommended safety processes, guidance of analysis techniques, safety planning, system design, system safety requirements, or the overall safety analysis results. The meaning of consistency in safety can be: a) having completed suggested or planned activities; b) having followed required or recommended analytical steps and syntax rules; c) having shown compliance with safety objectives; 4) having confirmed that the analysis is based on the ‘right’ design information; d) the logic and data in safety analysis are in agreement with each other.

In the context of this thesis, we address the term of consistency only as *model consistency* (as the last type of consistency mentioned above). The subjects that are associated with model consistency are the logic and data at different abstraction levels in safety assessment models. Consistent models imply that the relationships between overlapping or similar elements of two safety assessment models are in agreement with some relationship that is prescribed to hold. The prescribed relationship can be either expected similarity or expected differences. It is difficult to ensure that two models or more are consistent. Sometimes, the level of consistency achieved is analyzed and demonstrated through the investigation of potential inconsistencies across models or subjects that should be consistent, e.g. requirement inconsistency management [153], UML model and meta-model inconsistency in Model-Driven Development (MDD) [171, 191]. However, we understand that the diversity of the forms and data formats of safety analysis and the informality of many data items in safety assessment models (e.g. the textual description of failure events in a traditional fault tree) make it impossible to implement highly-automated consistency analysis across safety assessment models, as what can be achieved in software engineering (e.g. [171]).

In traditional logic, "*two or more statements are called consistent if they are simultaneously true under some interpretation*" [26]. This is the semantic meaning of consistency. In modern logic, the consistency of a set of statements means that "*no formula 'P & –P' is derivable*" from those statements by the rules of some logical calculus [26], which is the syntactic description. The semantic and syntactic definition of consistency underpins the investigation of inconsistency in different areas. Nuseibeh et al. [153] define an inconsistency as "any situation in which a set of descriptions does not obey some relationship that should hold between them. The relationship between descriptions can be expressed as a consistency rule against which the descriptions can be checked". This view is close to our mission of examining inconsistency between safety assessment models.

In light of Nuseibeh's definition of software requirements inconsistency, inconsistency between safety assessment models is defined in this thesis as *any situation in which two relevant descriptions in safety assessment models do not obey some relationship that is prescribed to hold between them*. The description can be a modelled subject, a modelled attribute, or a modelled relationship at different abstraction levels.

5.2.2 Typology of Safety Model Consistency

In this section, a typology of safety model consistency is proposed in order to facilitate the understanding of consistency issue among safety assessment models. This typology provides an overview of the types of consistency to be considered in safety reviews. Our primary concern is whether the system safety characteristics represented by safety assessment models are consistent with the real world situation. Figure 32 illustrates the structure of the typology.

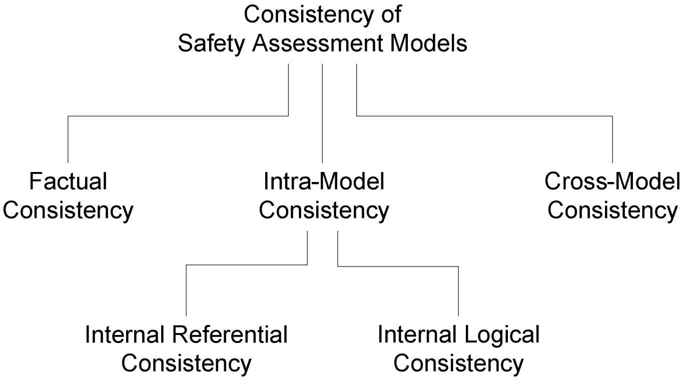


Figure 32 Typology of Safety Model Consistency

Depending on the nature and subject with which a safety assessment model is consistent or inconsistent, the consistency of safety assessment models can be divided into three groups:

- Factual consistency (C1)
- Intra-model consistency (C2)
- Cross-model consistency (C3)

Figure 33 illustrates the nature of and the relationships between the three types of consistency problems (denoted by the wide lines with arrows). Each type of model consistency is further described in later sections.

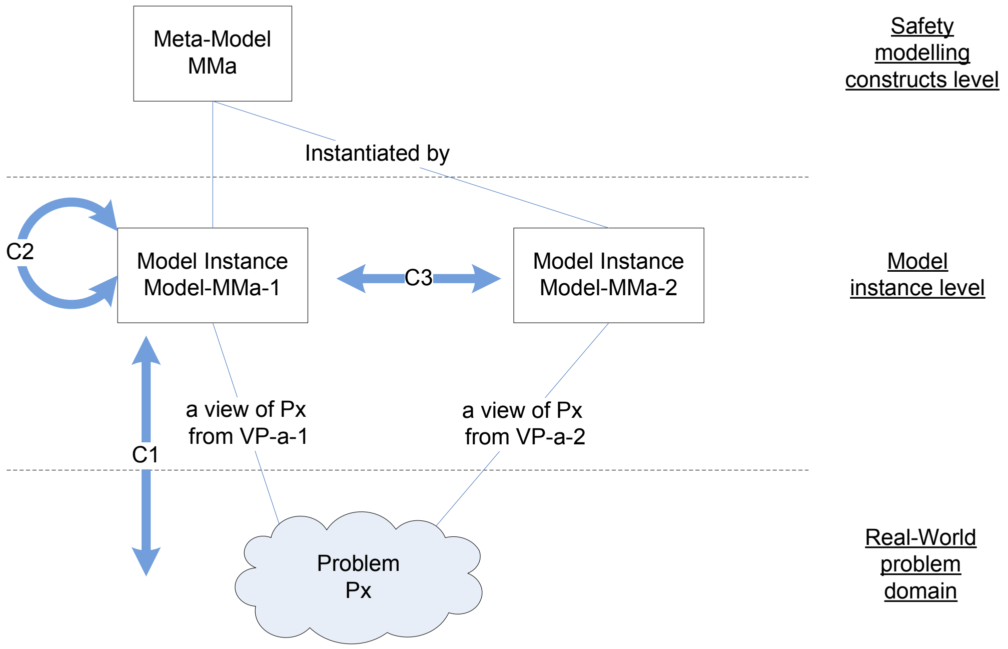


Figure 33 Consistency of Safety Assessment Models

As illustrated in Figure 33, the safety modelling methods that are used to deal with real world problems are presented as meta-models at the top level, e.g. the meta-model MM_a. Different safety techniques may have different modelling constructs, but the generic concepts in safety assessment models may be represented by a unified core data meta-model, such as CoreDMM presented in Chapter 3. It can be used as a bridge to communicate some common safety assessment models based on different techniques. Thus in Figure 33, only one block is presented at the safety modelling constructs level. A meta-model generally provides a view of a generic problem in reality, however not directly associated with concrete problem entities or instances. The safety assessment model instances developed on the basis of a safety modelling technique are presented as model instances in the middle level. For example, the model instance Model-MM_a-1 is a concrete instantiation of the meta-model MM_a. The problems or subjects under study, which are the subjects being modelled, are depicted in the bottom level. The model instances provide a view of a problem from a certain point of view, e.g. the model instance of Model-MM$_a$-1is a view of the problem P$_x$ from the viewpoint VP$_{-a\cdot1}$ If MM$_a$ represents the safety assessment CoreDMM, Model-MM$_a$-1 could be a concrete fault tree for a braking system of a car design. In this case, P$_x$ can be the 'safety of the braking system of a car design'. Model-MM$_a$-2 could be an FMEA of the braking system. In Figure 33, C1 denotes the factual consistency; C2 denotes the intra-model consistency; and C3 denotes the cross-model consistency.

In this thesis, our focus of study is not the consistency in relation to meta-models of safety analysis, which is sometimes named and classified as syntactical consistency and semantic consistency in software engineering literature [72, 191], but the consistency issues within the model instance level and between the instance level and the real world.

5.2.3 Factual Consistency

Factual consistency implies that there are no contradictions between the content of a safety assessment model and the logic or facts that exist in reality. It is represented in Figure 33 as C1. This is the central concern of model validity. Note that, asserting factual consistency of a model does not mean that there is no difference between the model and the real world problem domain, because abstraction and simplification during modelling are permitted and necessary to accommodate specific modelling purposes. In this thesis, the abstraction and simplification of models from reality that are accepted and undertaken by modellers are not treated as a factual inconsistency. Usually, factual consistency is difficult to observe and determine. We may not have sufficient understanding of both the model and the real problem represented by the model to confirm it. Field tests and operations, as the ideal way to check factual consistency, are usually not practical or are *too costly* and *too late* for the purposes of validating a model produced as part of a pre-operational safety case.

In some particular cases, if there are relationships or facts that are apparently and intuitively true (some component conditions are mutually exclusive or some relationships are impossible in reality), we can examine factual consistency partially, regarding the accepted relationships and facts, either manually or with tool support against the model directly. For example, if a component is in a 'failed' state, then it cannot also be in a 'working' state at the same time. A vehicle cannot be 'moving forward' and 'moving backward' at the same time. Some other intuitive factual relationships are based on methods. For example, circular logic between failure events is not permitted in fault tree construction. The check of circular logic is provided as a function in the FaultTree+ software by Isograph [105]. These factual relationships are self-evident and intuitive to humans, but not obvious for machines unless clearly defined. If human reviews have picked up such relationships in safety assessment models during review, the factual consistency of safety assessment models can be checked to some extent.

Above all, human review of safety assessment models is the primary means of checking factual consistency in the models directly. Domain knowledge and experience about the modelling subject under study and the axiomatic logic and facts on which the analysis is based and compliant with are the grounds for expected consistency relationships of possible checking.

Of course, the factual consistency of a safety assessment model could perhaps be recast as a problem of cross model consistency between safety assessment models and a domain model. However, this would breed another question as to the consistency of the domain model of the world.

5.2.4 Intra-Model Consistency

Intra-model consistency focuses on the consistency of information *within* one safety assessment model. It is represented in Figure 33 as C2. It is also sometimes referred to as self-consistency or internal consistency of a safety assessment model (e.g. internal semantic and syntactic consistency within fault tree models [135]). For example, within a FMEA worksheet, the same failure effect should be classified with the same severity level; within a specific fault tree, a repeated event should be depicted with the same name and identifier to avoid confusion. The examples imply that there are two major subtypes:

- Internal referential consistency
- Internal logical consistency.

As is known to any modeller, “naming can be one of the most difficult parts of modelling” [81]. Internal referential consistency is associated with the usage of object names in a model. It captures the requirements on consistent relationships between three aspects of an element of a model instance. The name of an element of a model instance, the meaning from interpreting the representation of the element of a model instance, and the entity being represented by the element of a model instance should all be consistent with each other within a single model instance.

Internal referential inconsistency can have two forms: *One-Name-Multiple-Referents, Multiple-Names-One-Referent*. One ‘name’ in a model instance should address one entity in reality, but not other entities at the same time. For example, consider a model element named as ‘Braking System’, representing a braking system in the design schema. If there are cases that both ‘front braking system’ and ‘rear braking system’ are named as ‘Braking System’ in the model, we cannot interpret the name of the element in the model correctly with respect to the corresponding real world entities. If one entity in reality is represented in the model with two names, we may interpret the two names as two real entities so that the model is inconsistent with the real situation. For example, a motor failure is named as ‘Motor failed’ and ‘Motor failure’ as two different basic events in a fault tree, then the fault tree might produce optimistic results through misinterpreting one event as two different events. Internal referential consistency should be checked by human reviewers manually. Some tools can provide some assistance for the situation of One-Name-Multiple-Referents, by listing relevant information on the ‘same’ model element according to the element name. For example, RAM Commander [12] by ALD can list the events connected with a repeated event in a fault tree for reviews by users.

From a case study of inconsistency between safety analyses results conducted in 2010 [194], it is observed that different wording and phrasing in safety analysis results is an important factor that hinders effective understanding during the consistency analysis of various safety analysis results. The referential inconsistency not only exists within one safety assessment model, but also exists between different models.

Internal logical consistency concerns whether two logical causal relationships presented in a safety assessment model are in agreement with each other. If internal logical inconsistency is identified, we *cannot* necessarily conclude which model is wrong in terms of representing the ‘real’ logic relations. But we can identify that at least one of the models under analysis is incorrectly defined due to the differences exhibited by these inconsistent logical situations. For example, in an aircraft FHA, if there are two failure conditions that have the same end effects at the aircraft level, then these two failure conditions should have been assigned with the same severity classification[11]. If that is not the case, the severity classification of one of the two failure conditions must be incorrect. If the internal logical consistency relations are clearly defined, tools can perform mechanical checks on a model if the naming problems had been sorted out.

[11] According to MISSA project partners, SARAA, a safety and reliability analysis database adopted by Airbus, can perform some of these checks, for the consistent usage of failure condition severity classification.

5.2.5 Cross-Model Consistency

This type of consistency concerns the relationships that should hold among two or more safety assessment models or their elements. It is represented in Figure 33 as C3. Model instances may vary from each other in many different ways due to many factors. Three principle factors resulting differences between models are: *Viewpoint* differences, *Knowledge* differences, *Representation* differences. Prior to the examination of the differences of knowledge on the system in question, the viewpoint differences and the representation differences between safety assessment models must be resolved.

Considering the domain context of safety assessment modelling that we introduced in Chapter 3, the differences between safety assessment models that are caused by inconsistent knowledge of a problem are of our particular interest during the *evaluation* of models. However, inconsistency across models caused by different knowledge or understanding is usually masked by and mixed with differences brought about by the viewpoint factor and the representation factor. Because of different viewpoints, the scope of model elements could be different, as can the abstraction level of model elements and the modelling constructs selected. These differences are allowable, but sometimes it may severely impede the identification of inconsistent understanding. In terms of representation, if safety assessment models were constructed based on different meta-models, or the model elements were named freely without prescribed naming conventions, the models would be different in 'appearance' naturally, even though humans can understand whether they are consistent in meaning. Despite the difficulties presented, however, the consistency between various models with respect to the content of the models (that represents our understanding of the problem domain) must be appropriately evaluated for the usage of models as an integral body of evidence in a safety case.

The degree of cross-model consistency may be demonstrated through the examination of models in terms of their compliance with pre-defined relationships. The relationships that hold between model elements in two safety assessment models are termed 'consistency relationships' in this thesis. They should be derived from engineering practice and experience, elicited by human experts in a transparent form. In this regard, cross-model consistency is similar to factual consistency and intra-model consistency. The foundation of evaluating all three types of consistency is the domain knowledge of expected 'consistency relationships'. Our understanding of the inconsistency between safety assessment models cannot exceed our knowledge of identified consistency relationships.

Similar to intra-model consistency, cross-model consistency, if achieved, cannot assure the factual consistency of models against reality. However, if cross-model inconsistency situations are identified, the factual consistency of the models would be undermined unless adequate explanation of the causes of inconsistencies can be offered.

From the discussion so far, we learn that model consistency is a complex issue. Human intervention must be part of the examination process, at least in support of the formulation of consistency relationships and the identification and resolving of naming inconsistencies. In practice, consistency is an open-ended issue that depends on the relationships we expect to hold for models. The common practice is to claim consistency according to the outcome of inconsistency analysis. There is little guidance on potential checking mechanisms and no explicit method for the inconsistency analysis of safety assessment models. Section 5.3 will describe some norms regarding consistency relationships summarised from engineering practice. Section 5.4 will describe our approach to address the wicked issue of cross-model consistency.

5.3 Consistency Relationships

5.3.1 Common Features

The consistency relationships between models usually exist as tacit knowledge in the engineering processes. In practice, the consistency between safety assessment models is usually expressed and explained in natural language. The recognition of consistency relationships is an important foundation of identifying inconsistency between safety assessment models. Regarding the consistency issue that exists between two safety assessment models, there are some common features of the consistency relationships to be specified.

- The subjects of the expected consistency relationships must be the same type of modelling concepts that are shared by the two model instances. The subjects of consistency relationships are the data elements in two safety assessment models that are to be examined for judgments on whether a defined relationship holds or not. For example, we can compare an object of *failure condition* with another object of *failure condition*; but we cannot compare an object of *failure condition* with an object of *system element*.

- The subjects of the expected consistency relationships must be the elements of safety assessment models that depict our understanding or prediction of behaviour in the problem domain being modelled, rather than the model elements depicting our knowledge or understanding about the models themselves or the modelling processes. For example, the assumptions or limitations of one model can differ from the assumptions or limitations of another model, but this is not a case of 'inconsistency' if all the assumptions and limitations are allowable statements needed for the modelling purpose of each model. Considering the safety assessment CoreDMM described in Chapter 3, the subjects of potential expected consistency relationships should be defined around the *substance elements* and the *construction elements* of the model instances of CoreDMM.

- The expected consistency relationships may be either *similarities* between data elements of two safety assessment models, or *differences* between data elements of two safety assessment models. For example, an expected consistency relationship can be 'Model A has less order-one MCSs than Model B does'. But some differences between models need not be viewed as inconsistency, e.g. 'the FMEA worksheets from different analysis groups can be of different data columns'.

- The expected consistency relationships can be very diverse. They may relate to the coverage or occurrence of a certain type of model elements, or the existence of a specified relation between two model elements. The complexity of consistency relationships can be different. Some can be examined directly; some need extra information to supplement the data directly available in the models; some need pre-processing of model data to enable comparison. Some consistency relationships are only expected to be true under certain pre-conditions.

The violation of the desired consistency relationships may be dischargeable. The consistency relationships are *ideal* relations that are potentially true according to the knowledge of the problem domain and the models. For example, we may naively expect that the failure effects of identical redundant pumps will be the same in a given safety assessment model. However, there may be good reasons as to why they are not. Even though it is claimed that they are desired relationships, some cases of the violation of the relationships are permitted or can be discharged if reasonable explanation of the cases of inconsistency can be supplied. The interpretation and justification of identified violation cases of the desired relationships should be conducted in later stages of inconsistency analysis.

Violations of expected consistency relationships may indicate that models are inconsistent in terms of a desired relation between two comparable models. However, if there is no violation of a set of predefined consistency relationships, we cannot claim that model consistency has been *fully* achieved. Because the predefined set of expected consistency relationships may be incomplete or insufficient, we can only evaluate and claim the level of consistency between models within the scope of identified consistency relationships after analysis.

5.3.2 Types of Consistency Relationships

Through discussion with safety experts in the MISSA project, we find that consistency relationships between models in engineering practice can be characterised into four groups (adapted from [194]).

- **Identity-Based Consistency Relationships**: most of the relationships that we claimed would hold between safety analyses are of this type. It means that the observation and description for one thing from a same viewpoint should not have gaps and contradictions. For example, the failure modes of a valve in one safety analysis should be consistent with the failure modes of the same valve presented in another safety analysis.

- **Analogy-Based Consistency Relationships**: some of the consistency relationships are based on the similarities between the two subjects or relationships. It is common engineering practice to evaluate analysis according to experience and knowledge of a similar counterpart. For example, similar components in a system may have similar safety features presented in safety analysis; new systems may bear some similar features of a similar historical system. This consistency relationship is less strict in terms of compliance due to that the differences of the similar parts may discharge some of the inconsistencies identified.

- **Correlation-Based Consistency Relationships**: some of the practical consistency checking is based on the heuristics of some correlations between analysis data elements by analysts or reviewers. The inconsistencies found against this type of consistency relationships must be further examined or it would be hard to determine whether the inconsistencies indicate inadequate or flawed safety analysis. For example, we may define a consistency relationship that ‘with detailed design information and concrete safety requirements, the system safety should be at least as good as we analyzed at the earlier stage’ or ‘the severity of a failure condition is proportional to the speed of a system or the redundancy configuration of the system’. In [133], a study of inconsistency between incremental safety analysis is performed according to this type of consistency relationships.

- **Agreement-Based Consistency Relationships**: there are requirements and assumptions in one safety analysis, which address the information to be provided or confirmed by another piece of safety analysis. For example, if there is an assumption associated with a fault tree FT_x concerning the independence between Component X1 and Component X2, this assumption could be discharged (or challenged) by a related common mode analysis (CMA). The actual commitment of the agreement should be checked before both of the analysis results are signed off.

During safety reviews, the consistency relationships being checked are important because they are the grounds for the search for inconsistencies and they should be considered when the associated inconsistencies are evaluated and treated. Some of them are domain-specific, context-specific, or instance-specific. We may heavily rely on human expertise to implement the checking of the defined consistency relationships for models.

5.3.3 Consistency Related to Conditions

Consistency relationships related to conditions have been adopted to assist the review of safety analysis in practice. An example of inconsistency FMEA and FTA is reported in [57]. Far from the ideal situation, some failure modes identified in an FMEA that contributed to the undesired top event in a fault tree were not shown in the fault tree; whereas, some component failures that would contribute to a high-level undesired event in the fault tree were not identified in the FMEA result of the corresponding system. The findings directly reveal the incompleteness of the FTA and the FMEA performed. This kind of cross-check between the conditions addressed by FTA and FMEA has also been recommended in the NASA Fault Tree Handbook [150].

From engineering practice, we found that three prominent concerns about the validity of most qualitative safety assessment results are the completeness of the conditions identified, the correct identification of causes and effects of conditions and the correct classification of conditions in terms of their severity. Correspondingly, there are three types of consistency relationships related to the ‘conditions’ of a system under safety analysis.

The first type of consistency relationship related to conditions in safety assessment models is about the completeness of the identified of conditions in models. For example, a basic event in a fault tree should be visible within an FMEA at a corresponding abstraction level. A failure mode of a component in one FMEA should be addressed in another FMEA if comparable FMEA analyses are available.

The second type of consistency relationship related to conditions in safety assessment models concerns missing or flawed logical relationships between conditions. For example, a failure mode in an FMEA that led to a catastrophic failure of a system is not addressed by the fault tree of that same catastrophic failure of the system.

The third type of consistency relationship related to conditions in safety assessment models is about the consistent classification of conditions in terms of their severity or occurrence frequency. One condition addressed by two models should be classified with the same severity or occurrence level according to a common classification scheme. This is usually related to the risk analysis part of safety assessment models.

It sounds simple to examine consistency related to conditions. But it is not so simple due to the following two reasons: a) the level of detail involved in different causal analyses may differ, preventing a straightforward comparison of conditions in two models; b) the conditions considered in different causal or consequence analyses may have varied focuses. For example, one model may focus only on electromagnetic effects, whereas another focuses on hydraulic failures. Equally, one model may focus exclusively on the cause of a single 'top event', whereas another may be addressing multiple outcomes. The abstraction level of conditions and the scope of conditions should be considered during the identification and diagnosis of inconsistency between conditions in two models.

5.3.4 Consistency Related to MCSs

MCSs are the minimal combinations of basic events that can cause the top event of a fault tree. It is an important type of analysis result originated from qualitative FTA. It is widely accepted that lots of information can be implied by the set of MCSs contributing to an undesired event [137, 150, 200], such as whether there are unexpected single-point failures or whether the design intent of failure control has been fulfilled [137]. The MCSs are the essential relationships between logical combinations of conditions and the top level event. Two fault trees with the same set of MCSs are in fact same in nature even if the tree structures of the two fault trees are presented differently (e.g. have different intermediate events or different logical relationships). Therefore, for two fault trees with the similar level of detail, comparing the set of MCSs is a reasonable way to determine whether two fault trees are consistent in their causal logical structures. The MCSs can be viewed as a converted form of causal structural relationships in fault trees that is obtained from the more complicated hierarchical causal relationships in the graphical tree.

In recent years, formal methods have been employed within system safety analysis (as reviewed in Section 2.4.2). However the validation of these formal safety assessment models is still in its infancy (refer to Section 2.6.2 for more details). Some attempts have been set on comparing modelling outputs from two safety assessment models based on formal languages, such as [133]. However, as explained in Section 2.4.3, these models are constructed on the basis of results of hazard identification or FMEA. In nature, they are bottom-up consequence analysis with analysis results provided in the format of MCSs. A cross check of these models with other *top-down* safety analysis results may disclose some hidden inconsistency between models as with the comparison of an FMEA and a fault tree. In this regard, MCSs, as a shared form of modelling substance results, can play an important role in the examination of the consistency of a fault tree and a safety assessment model based on a formal language.

Consistency relationships related to MCSs are a more complicated form of the second type of the consistency relationships related to conditions. They are more difficult and complex to define and examine, because the conditions addressed by two safety assessment models must have some overlapping and the correspondences between conditions of two safety assessment models must be specified beforehand in order to enable the comparison of MCSs. Consider a case of consistency relationships between two fault trees. Because FTA is very flexible, the definition of the top event could help to scope the conditions included in the analysis, we need to confirm the relationships between top events in two trees carefully to decide whether or not the MCSs of the two trees need to be consistent. Even if the undesired top event was the same in models, if the components of MCSs of the two trees were not be at the same level of detail, we need to sort out the corresponding relationships between conditions in two models before the consistency relationships between MCSs can be analyzed.

Despite these difficulties, it is still possible to examine cross-model consistency on the basis of MCSs, which may reveal inconsistent causal relations between conditions that cannot be observed by comparing other modelling elements.

5.4 Cross-Model Inconsistency Analysis

In this section, an overview and illustration is provided of a cross-model inconsistency analysis method for safety assessment models. The cross-model inconsistency analysis is walked through with two exemplar safety assessment models of a simple hypothetical system. A case study of the inconsistency analysis method is presented in Appendix C.4 for evaluation.

5.4.1 Method Overview

The method utilises the '*mismatch'* mechanism of human mind in logical thinking [61]. As humans, when we come across scenarios that are contradictory or inconsistent with our patterns of expectation, we will feel uncomfortable. De Bono states that this mechanism is important for preventing us from making mistakes [62]. From a different perspective, we take advantage of this mechanism to serve for the purpose of identification of inconsistencies that may undermine our confidence in safety assessment models.

Comparison is the core activity involved in the cross-model inconsistency analysis method. As an activity of putting things together and observing differences and similarities between individuals, comparison is the heart of many mental processes, but usually an implicit and spontaneous step. In the software engineering domain, comparison has been adopted to test model transformation and model composition [125] and to support inconsistency management [187]. For the consistency issue between safety assessment models, we also need to capture and transfer the implicit mental comparison process conducted by human experts in engineering reviews into explicit steps with a clear description of mechanisms and outputs.

The cross-model inconsistency analysis method consists of six concrete phases:

Phase 1: Selecting models to be compared.

Phase 2: Elaborating consistency relationships that are required to hold.

Phase 3: Implementing model comparison.

Phase 4: Interpreting comparison results.

Phase 5: Evaluating the inconsistency analysis.

Phase 6: Citing and justifying inconsistency analysis results.

A collection of safety assessment models, which is referenced as a set of evidence items in a safety case, can be examined through this inconsistency analysis process, which is beneficial to build up confidence in the trustworthiness of evidence items. Figure 34 presents a model of information elements contained in the inconsistency analysis process. The information model is depicted in UML as a model at the M1 level in the OMG meta-modelling structure.

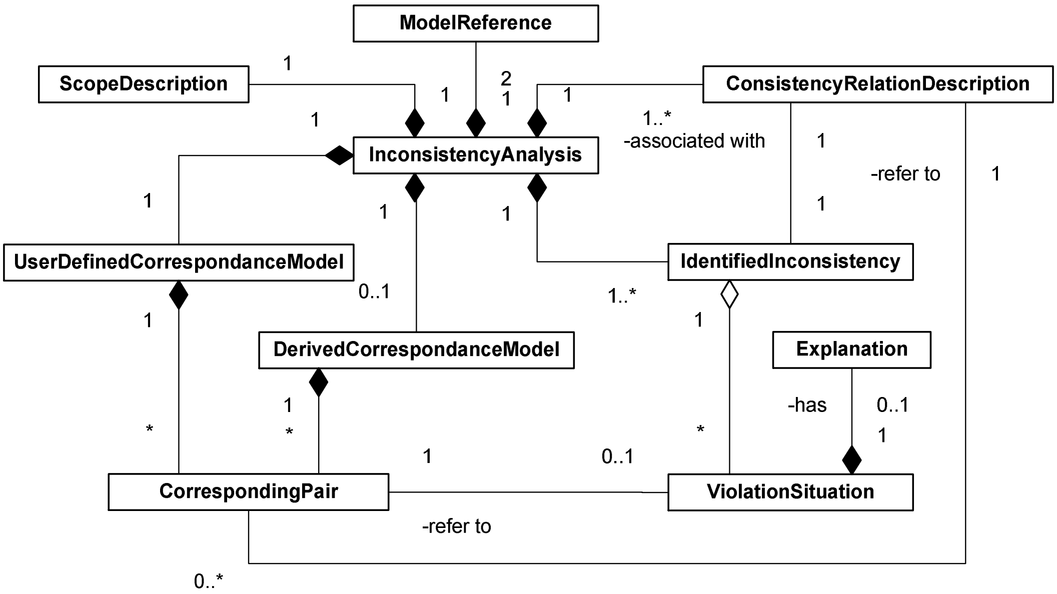


Figure 34 Information Model of Inconsistency Analysis

InconsistencyAnalysis is the container class that packages up all data elements in an inconsistency analysis. *ScopeDescription* and *ModelReference* are classes that are used for capturing the output data from Phase 1 of the analysis. *ScopeDescription* describes the scope of model elements that are addressed in an inconsistency analysis; *ModelReference* documents the references to the artefacts of the models under inconsistency analysis (e.g. references to *SafetyAssessmentArtefact* in CoreDMM presented in Chapter 3). *ConsistencyRelationDescription* documents the defined consistency relationships to be examined in the inconsistency analysis. *CorrespondingPair* contains the relationships between elements of two models under analysis, either defined by a user (composed of *UserDefinedCorrespondanceModel*) or derived through an algorithm (composed of *DerivedCorrespondanceModel*) according to a defined consistency relationship. Through observation on the *CorrespondingPair*s, the model element pairs that violate the defined consistency relationship can be identified and recorded as *ViolationSituation*. Users may provide 'Explanation' for *ViolationSituation*s. The *ViolationSituation*s without proper reasons are presented as the substantial inconsistency analysis results – *IdentifiedInconsistency*.

The relationships between the analysis phases and the output information are depicted in Figure 35. The flowchart on the left side illustrates the analysis method; the grey boxes illustrate the information elements obtained when the corresponding analysis phase finishes. The data in the top four grey boxes have corresponding parts in the information model in Figure 34. The two grey information boxes near the bottom are presented with dashed lines. Those two pieces of information are generated in the analysis process, but the documentation of the evaluation of the inconsistency analysis and the documentation of confidence arguments that use the inconsistency analysis results, are not included in the information model (as Figure 34 presented) of the inconsistency analysis. (Instead, we describe how the evaluation results and justification can be presented within the context of a safety case in Chapter 6.)

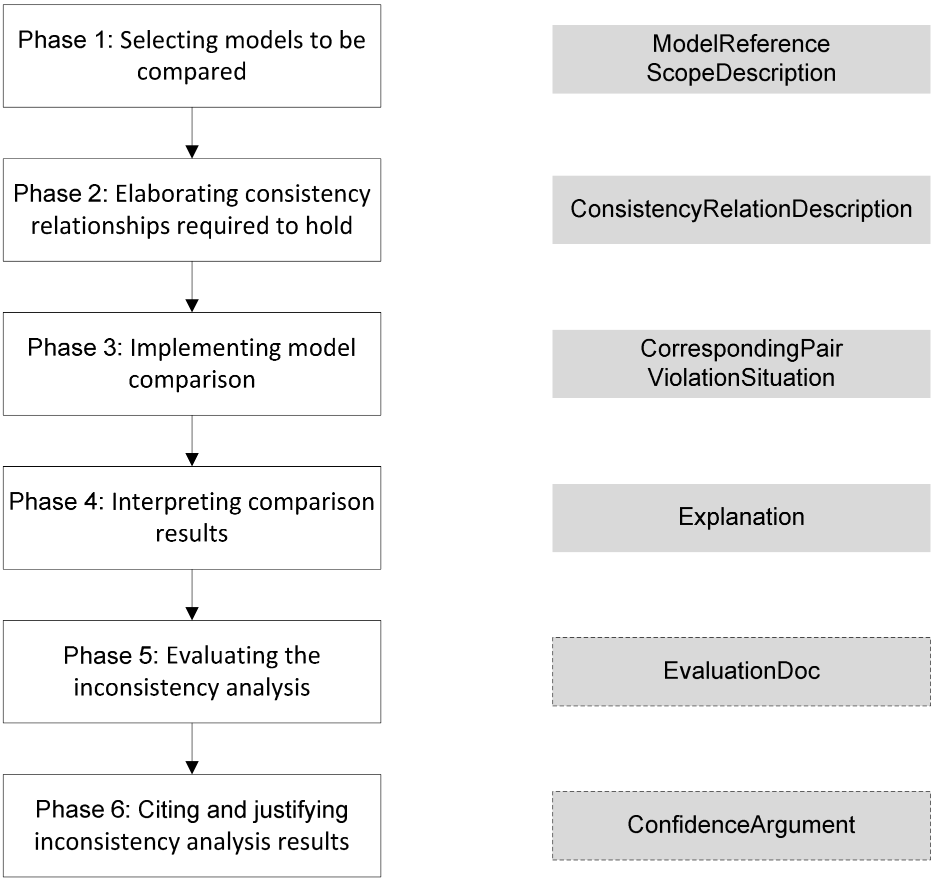

Phase 1: Selecting models to be compared
Phase 2: Elaborating consistency relationships required to hold
Phase 3: Implementing model comparison
Phase 4: Interpreting comparison results
Phase 5: Evaluating the inconsistency analysis
Phase 6: Citing and justifying inconsistency analysis results
ModelReference
ScopeDescription
ConsistencyRelationDescription
CorrespondingPair
ViolationSituation
Explanation
EvaluationDoc
ConfidenceArgument

Figure 35 Six Phases of Cross-Model Inconsistency Analysis

Section 5.4.3 ~ Section 5.4.8 expand on the details of each phase of the inconsistency analysis and explain how the information elements in Figure 34 are obtained during the analysis. Section 5.4.2 introduces the exemplar models that are used as example subjects under inconsistency analysis that are employed for illustrating the analysis phases in Section 5.4.3 ~ Section 5.4.8.

5.4.2 Exemplar Models for a Running Example

This section presents the description of two simple models of a hypothetical system that are adopted for the in-line illustration of the inconsistency analysis phases. The hypothetical system S_x consists of three components, h_a, h_b, and h_c. The hypothetical system functions by passing an input flow through the three components and provides an output flow. Figure 36 describes the hypothetical system structure.

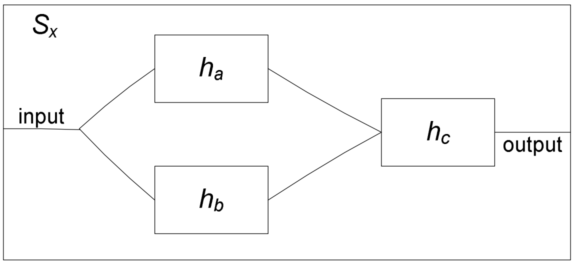
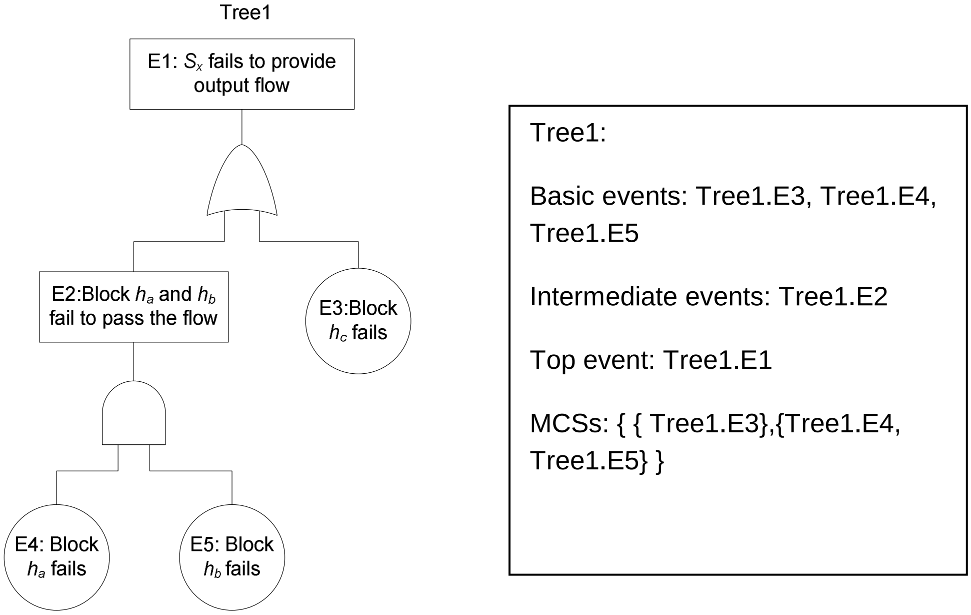


Figure 36 A Hypothetical System S_x

Assume that there is always a correct input to the system. We have constructed two models to understand the causes that may lead to the failure of the system – "*no output flow provided*".

The first model is a fault tree (labelled as 'Tree1' in the consistency analysis process). The manually constructed tree structure and the summary of events and MCS results of 'Tree1' are depicted in Figure 37. Conventionally, the labels of events in 'Tree1' in Figure 37 do not need to include the identification of the label 'Tree1' as a prefix. But, during the cross-model inconsistency analysis, the events presented in 'Tree1' are labelled as 'Tree1.E1', 'Tree1.E2', 'Tree1.E3', 'Tree1.E4' and 'Tree1.E5' respectively for easy identification of the model that they belong to.


Tree1
E1: S_x fails to provide output flow
E2:Block h_a and h_b fail to pass the flow
E3:Block h_c fails
E4: Block h_a fails
E5: Block h_b fails

Tree1:

Basic events: Tree1.E3, Tree1.E4, Tree1.E5

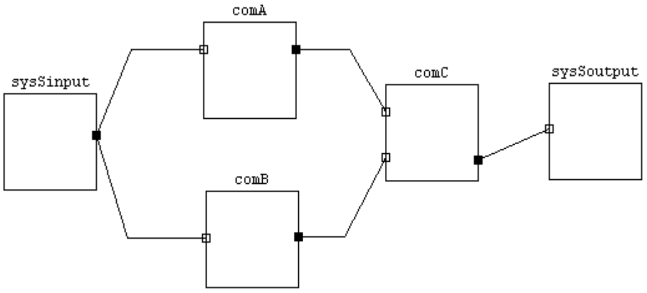
Intermediate events: Tree1.E2

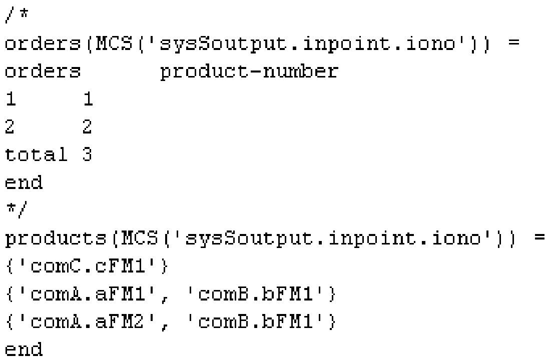
Top event: Tree1.E1

MCSs: { { Tree1.E3},{Tree1.E4, Tree1.E5} }

Figure 37 A Fault Tree Model of S_x

The second model is a failure logic model constructed with AltaRica in OCAS Cecilia (Section 2.4.2 and Section 3.4.2 have addressed a brief review of models and analysis associated with AltaRica). The graphical view of the model and the MCS outputs generated from OCAS Cecilia are presented in Figure 38.

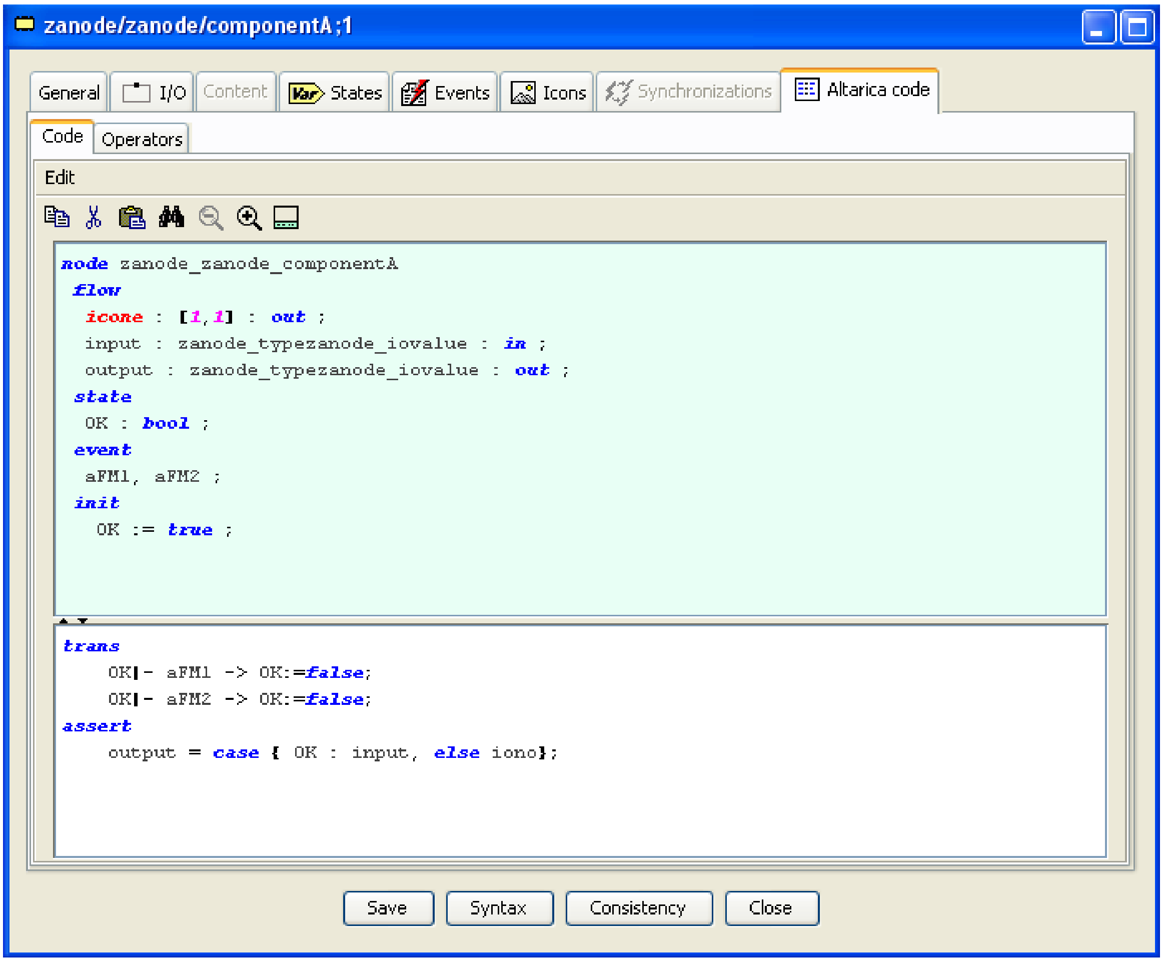




```
/*
orders(MCS('sysSoutput.inpoint.iono')) =
orders      product-number
1     1
2     2
total 3
end
*/
products(MCS('sysSoutput.inpoint.iono')) =
{'comC.cFM1'}
{'comA.aFM1', 'comB.bFM1'}
{'comA.aFM2', 'comB.bFM1'}
end
```

Figure 38 OCAS AltaRica Model and MCS Outputs

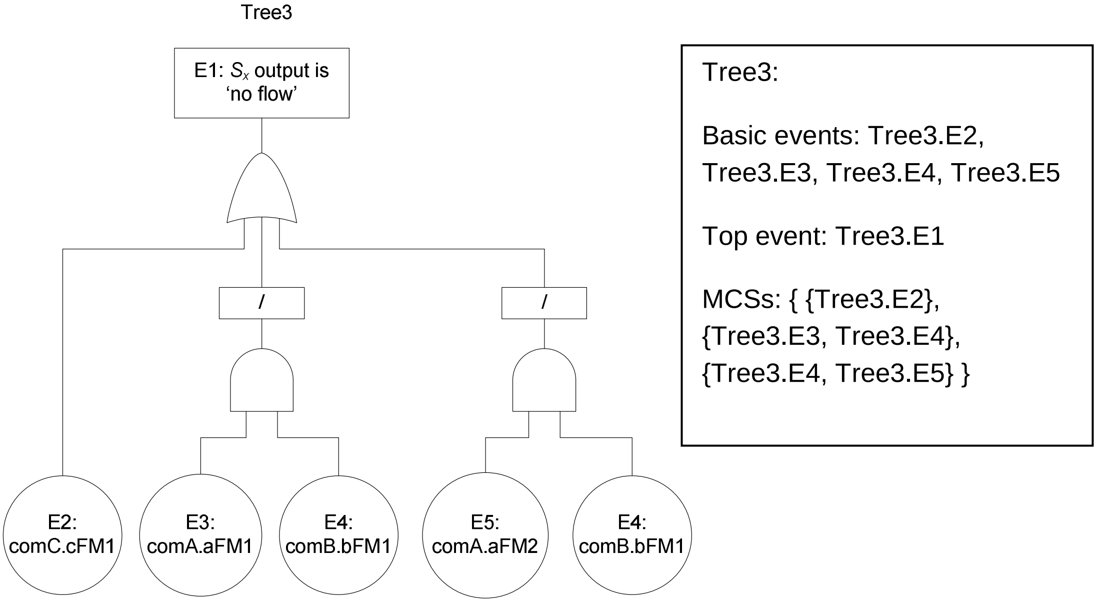
The ‘comA’, ‘comB’ and ‘comC’ in Figure 38 are the AltaRica nodes that model the components, h_a, h_b, and h_c in S_x respectively. The input and the output of S_x are modelled by another two nodes, ‘sysSinput’ and ‘sysSoutput’ in Figure 38. Each of the five nodes in the OCAS model is defined in AltaRica, which are codes in the similar shape as the node example in Figure 18. The screenshot of the definition of the AltaRica node ‘comA’ in OCAS is shown in Figure 39. During the inconsistency analysis, we will not use the source AltaRica codes of the model, so other source codes of the model are not presented in this section. Instead, we use the MCS outputs generated in the AltaRica modelling environment during the inconsistency analysis. To provide an easy understanding of the MCS modelling results of the AltaRica model (presented in the lower section of Figure 38), an equivalent fault tree (labelled as ‘Tree3’) of the results of MCS analysis of the AltaRica model, in the conventional fault tree view, is formulated, as shown in Figure 40.



```
node zanode_zanode_componentA
 flow
  icone : [1,1] : out ;
  input : zanode_typezanode_iovalue : in ;
  output : zanode_typezanode_iovalue : out ;
 state
  OK : bool ;
 event
  aFM1, aFM2 ;
 init
   OK := true ;
```

```
trans
    OK|- aFM1 -> OK:=false;
    OK|- aFM2 -> OK:=false;
assert
    output = case { OK : input, else iono};
```

Figure 39 AltaRica Definition of Node ‘comA’ in OCAS



Tree3:

Basic events: Tree3.E2, Tree3.E3, Tree3.E4, Tree3.E5

Top event: Tree3.E1

MCSs: { {Tree3.E2}, {Tree3.E3, Tree3.E4}, {Tree3.E4, Tree3.E5} }

Figure 40 Equivalent Fault Tree of OCAS MCS Outputs

In Figure 40, the failure events modelled in the AltaRica model are relabelled in the same style of events in ‘Tree1’. For example, a failure event ‘comC.cFM1’ of the node ‘comC’ is labelled as ‘Tree3.E2’.

5.4.3 Phase 1: Selecting Models

The starting point of the comparison process is to have two model instances identified as the subjects of the inconsistency analysis. The model instances may be selected from the outputs of planned safety assessment activities, or historical analyses, or specially-designed parallel analysis, but should not be selected randomly. The models under comparison should have inherent relationships between information elements of the two model instances, such as similar subjects being modelled, similar purposes, similar modelling scopes, or similar types of modelling results. Two safety assessment model instances with no aspects in common are not proper subjects for concern about consistency.

The model-selection decision process is presented in Figure 41. At the beginning of this phase, we need to determine the model (M_a) whose validity is of concern first. Once M_a is chosen, we can select another model (M_b) from the models generated in the same system development and operation process or the models of similar systems. We should know the modelling method that is adopted by each model (we may obtain it from the ‘Method’ data element of CoreDMM presented in Chapter 3); we should also know the domain subject and concerns of the models (we may obtain them from the *SubjectofStudy* data element and the *Scope* data element of CoreDMM). There are three decision points to help us to exclude irrelevant models that are not suitable for the comparison-based inconsistency analysis. At the end of this phase, two models (M_a and M_b) that are potentially suitable for comparison should be chosen for the next phase of the inconsistency analysis. The essential requirements of comparable models are that M_a and M_b have common system elements or substance elements or construction elements. However, if the essential requirements are satisfied, the two models selected in this phase still may quit the inconsistency analysis if no proper consistency relationships are identified in the next analysis phase.

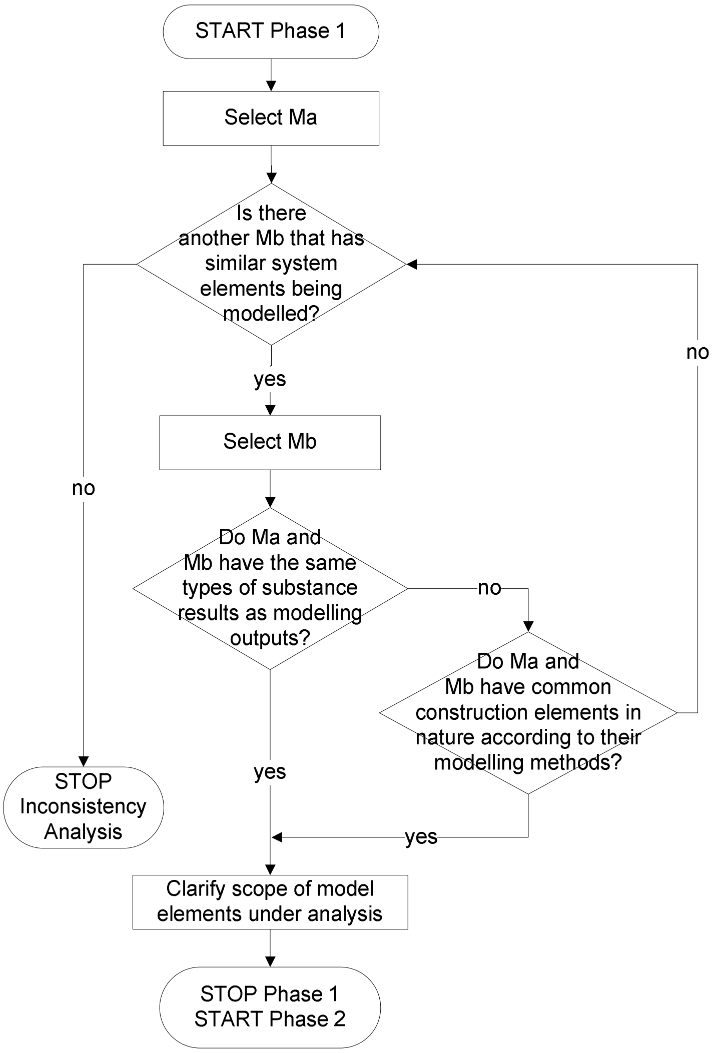

START Phase 1
Select Ma
Is there another Mb that has similar system elements being modelled?
yes
no
no
Select Mb
Do Ma and Mb have the same types of substance results as modelling outputs?
no
yes
Do Ma and Mb have common construction elements in nature according to their modelling methods?
yes
STOP Inconsistency Analysis
Clarify scope of model elements under analysis
STOP Phase 1 START Phase 2

Figure 41 Phase One Flowchart

The selection of the two comparable models, M_a and M_b, can be recorded using the *ModelReference* shown in Figure 34. Furthermore, the scope of the model elements (from M_a and M_b) to be addressed in the inconsistency analysis can be recorded using the *ScopeDescription* shown in Figure 34, if the inconsistency analysis only addresses some parts of M_a and M_b. The correspondences between model elements defined and derived in model comparison in Phase 3 are within the boundary of *ScopeDescription*.

Running Example of Phase 1:

Consider the safety analyses of the hypothetical system S_x (as presented in Section 5.4.2). We select the AltaRica model as M_a and the fault tree model as M_b. The information collected at the end of Phase one is depicted in Table 7.

ModelReference	M_a: an AltaRica model of S_x, with an equivalent fault tree - Tree3
	M_b: a fault tree of S_x - Tree1
ScopeDescription	The overlapping system elements include: S_x, h_a, h_b and h_c. The overlapping concern is 'S_x fails to provide the output flow' The shared type of substance results is: a set of MCSs. The common construction elements include: the condition of h_a, the condition of h_b, the condition of h_c, the condition of S_x. A condition means a 'failure' shown in the MCSs in M_a and means a 'basic event' in M_b.

Table 7 Phase One Output in the Running Example

5.4.4 Phase 2: Elaborating Consistency Relationships

Once we have selected the two models for examination, we need to clarify the consistency relationships that are desired between two models instances and the set of elements in the two models that should exhibit these relationships. This step requires human experts, on the basis of their understanding of the features of model instances and the relationships between model instances, to elicit the concrete consistency goals that they may adopt during an informal review of the model instances. Figure 42 shows the process of Phase 2. The elicited consistency relationships between M_a and M_b can be recorded using the *ConsistencyRelationDescription* shown in Figure 34.

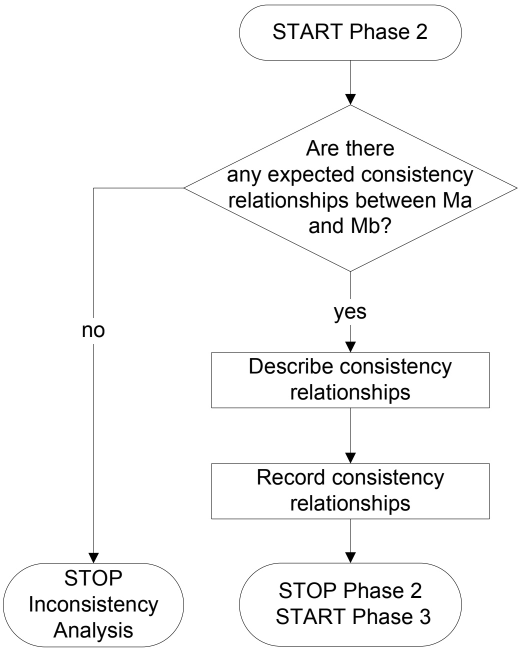

START Phase 2
Are there any expected consistency relationships between Ma and Mb?
no
yes
Describe consistency relationships
Record consistency relationships
STOP Inconsistency Analysis
STOP Phase 2 START Phase 3

Figure 42 Phase Two Flowchart

The statements on the relationships that are to hold between two comparable safety assessment models are also called as consistency checking rules [69, 153]. The domain experts who nominate the desired relationships should have knowledge and experience of safety and/or the system under study. The consistency relationships suggested by different experts should be reviewed for their reasonableness and accepted by safety analysts. The consistency relationships can be dictated either at the level of elements of model instances, or at the level of elements of meta-models. For example, if two fault trees for a catastrophic aircraft failure condition are being selected for inconsistency analysis, a human expert may want to examine whether the two trees are consistent in terms of the coverage of contributing conditions to the undesired top-level failure condition. If one failure event is presented in one tree, but not in the other, the analyst may claim that an inconsistency was identified. This phase can only be performed by a human. Some common types of consistency relationships are explained in Section 5.3.

Two comparable models can have a number of associated consistency relationships. Considering the typical subtypes of the substance results in CoreDMM (presented in Chapter 3), we may have the following expected consistency relationships for two safety assessment models (shown in Table 8). They are not meant to be an exhaustive list, but as examples for potential relationships.

Examples of Consistency Relationships (ExCRs)	
Consistency Relationship	Motivation
ExCR1: The cardinality of the smallest cutset in the MCS of M_a is the same as the cardinality of the smallest cutset in the MCS of M_b.	The concern here is whether the two models indicate a differing level of failure redundancy in the failures required for a top event.
ExCR2: Each condition in M_a has a corresponding condition in M_b that represents the same condition in reality.	The concern is the relative completeness of the conditions included in one model to those included in another.
ExCR3: Each MCS in M_a has a corresponding MCS in M_b that contributes to the same undesired top event.	Unlike the first example (ExCR1), which only considered the degree of redundancy, the concern here is whether the similar logical combination of conditions can be found in both models.

Table 8 Examples of Consistency Relationships

The exemplar consistency relationships are all associated with ‘condition’s, which are the core elements that are expressed by almost every kind of qualitative safety assessment models.

Running Example of Phase 2:

Based on the knowledge of the two models presented in Section 5.4.2, we decide to examine the two consistency relationships in this running example **- ExCR2** and **ExCR3** in Table 8.

A concrete definition of the consistency relationships between the selected M_a and M_b is presented below.

Let C_a be a set of 'failures' shown in the MCSs in M_a within inconsistency analysis scope; in our case, C_a ={ Tree3.E2, Tree3.E3, Tree3.E4, Tree3.E5}.

Let C_b be a set of 'basic events' in M_b within inconsistency analysis scope; in our case, C_b = { Tree1.E3, Tree1.E4, Tree1.E5}.

Let MCS_a be a set of minimal cut sets in M_a within inconsistency analysis scope; in our case, MCS_a = { { Tree3.E2},{ Tree3.E3, Tree3.E4},{ Tree3.E4, Tree3.E5} }.

Let MCS_b be a set of minimal cut sets in M_b within inconsistency analysis scope; in our case, MCS_b = { { Tree1.E3},{ Tree1.E4, Tree1.E5} }.

Let $R_{type}(x,y)$ be a correspondence relation between two model elements of a same *type*, from M_a and M_b respectively, depicting that they can be used for the representation of a same problem domain object. For example, R_{con}(*condition1*, *condition1'*) means *condition1* in M_a can be viewed as synonymous of *condition1'* in M_b. For $x\in C_a$ and $y\in C_b$, $R_{con}(x,y)\subset C_a \times C_b$ is defined by users according to their understanding of the two models, depicting the correspondence relation between conditions in M_a and M_b; for $x\in MCS_a$ and $y\in MCS_b$, $R_{mcs}(x,y) \subset MCS_a \times MCS_b$ is defined by users or inferred according to $R_{con}(x,y)$, depicting correspondence relations between MCSs in M_a and M_b.

Table 9 presents the final definition of the consistency relationships provided in Table 8. The definition of **ExCR1** is presented for a complete illustration of Table 8, but it is not examined in the running example of inconsistency analysis. Only **ExCR2** and **ExCR3** are examined in the running example.

ExCR1:	$(\forall mcs_x \in MCS_a, \exists mcsa_x \in MCS_a \mid \#\ mcsa_x \leq \#\ mcs_x)$ and $(\forall mcs_y \in MCS_b, \exists mcsb_y \in MCS_b \mid \#\ mcsb_y \leq \#\ mcs_y)$ and $(\#\ mcsa_x = \#\ mcsb_y)$
ExCR2:	$(\forall x \in C_a, \exists y \in C_b \mid R_{con}(x,y))$ and $(\forall y \in C_b, \exists x \in C_a \mid R_{con}(x,y))$
ExCR3:	$(\forall x \in MCS_a, \exists y \in MCS_b \mid R_{mcs}(x,y))$ or $(\forall y \in MCS_b, \exists x \in MCS_a \mid R_{mcs}(x,y))$, given $R_{con}(x,y) \subset C_a \times C_b$

Table 9 Elicited Consistency Relationships of the Running Example

A set of conditions defined above (e.g. C_a , C_b) corresponds to an instance of *HazardSet* in CoreDMM; a member of a set of conditions (e.g. Tree3.E2, Tree1.E3) corresponds to an instance of condition in CoreDMM; a set of MCSs defined above (e.g. MCS_a , MCS_b) corresponds to an instance of *MCSSet* in CoreDMM; a MCS member of a set of MCSs (e.g. {Tree3.E2}, {Tree1.E4, Tree1.E5}) corresponds to an instance of *MCS* in CoreDMM.

5.4.5 Phase 3: Implementing Model Comparison

If the aspects of two model instances to be examined for inconsistency are clearly defined, the comparison can be implemented by human experts directly or with some machine support in terms of data retrieval. Some of the comparison activities are simple and easy to implement. For example, if the failure rate of a specific failure mode in one model instance is expected to be higher than the failure rate of a failure mode in another model instance, it is easy to establish whether the expected relationship is true or false. However, some comparison tasks are more complicated. For example, if we want to compare whether the minimal cut sets of an undesired event in a fault tree and the minimal cut sets for the same undesired event generated in another model with formal language constructs are consistent, the implementation will involve multiple steps, considering the number of minimal cut sets and the varied data formats in the two model instances. The complexity of implementation is related to the nature of the consistency relationships and the data elements to be compared.

As we have discussed in Section 5.2.4, naming is a common problem we encounter while evaluating models at the instance level. A common naming convention in the construction of fault trees is recommended to reduce the inconsistency in fault trees and ease understanding of the referent being addressed [137, 150, 207]. However, it is almost impossible to force and ensure a same name is always used for the same thing in two different safety models in reality. A challenge of cross-model inconsistency analysis at the implementation stage is to resolve the different expressions or names of the same instance or object in the real world, either the name of a condition or the name of a system element.

To work around the naming problems, users are required to specify *correspondences* between the same types of model elements being compared in order to enable further examination of the compliance or violation of the consistency relationships expected. This is very different from most consistency studies in the MDE domain, in which a unified naming convention is often adopted and the name-referent consistency between models is assumed.

At the implementation phase of model inconsistency analysis, the *correspondence* between elements of models under comparison is an important concept. In ISO/IEC/IEEE 42010 [10], a correspondence is a relation between Architecture Description elements and can be used to express consistency, traceability, composition, etc. Similarly, a *correspondence* in inconsistency analysis defines a relation between elements of two safety assessment models, which is defined by users or derived on the basis of relationships between different types of model elements according to expected consistency relationships. The type of correspondences needed is governed by the consistency relationships elaborated in Phase 2. During the process of defining correspondences between two models, the user may identify model elements without proper correspondences. This may indicate violations of certain consistency relationships. For example, an identity-based consistency relationship -‘the failure modes of Element X_a in M_a should be of the same number and type as the failure modes of the corresponding element X_b in M_b’ – is violated if we cannot find corresponding failure modes for X_a in M_a from the list of failure modes of X_b in M_b. The derived correspondences are the correspondences inferred on the basis of user-defined ones, e.g. correspondences between MCSs inferred according to the correspondences between conditions set by users. After that, the user can examine the reasonableness of derived correspondences. The user may observe some derived correspondences that are against the content of expected consistency relationships. These derived correspondences will be identified as violation situations, which are treated as identified inconsistencies if no reasonable explanation is provided.

Tool support at the implementation phase of model comparison can be helpful for increasing the analysis efficiency, especially when the models under comparison are in large scale. The machine support can be desired for the following two tasks: the correspondence definition and the violation identification. But it can be difficult to design a common automated tool for all potential model comparison due to the variety of model data formats and desired consistency relationships.

Figure 43 shows the details of key implementation steps of Phase 3.

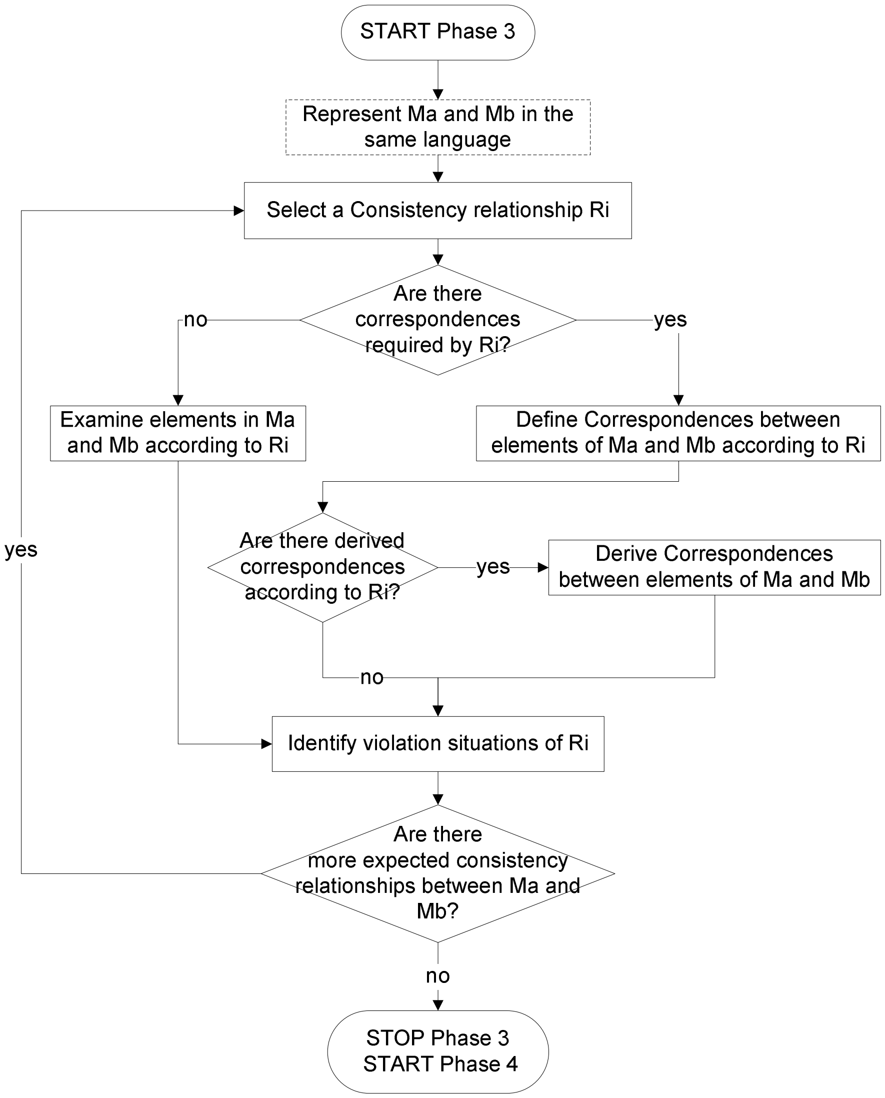

START Phase 3
Represent Ma and Mb in the same language
Select a Consistency relationship Ri
Are there correspondences required by Ri?
no
yes
Examine elements in Ma and Mb according to Ri
Define Correspondences between elements of Ma and Mb according to Ri
yes
Are there derived correspondences according to Ri?
yes
Derive Correspondences between elements of Ma and Mb
no
Identify violation situations of Ri
Are there more expected consistency relationships between Ma and Mb?
no
STOP Phase 3
START Phase 4

Figure 43 Phase Three Flowchart

The correspondence relationships between the overlapping elements in M_a and M_b, which are a set of *CorrespondancePair* in Figure 34, form the *UserDefinedCorrespondanceModel* and *DerivedCorrespondanceModel* in Figure 34. In addition, the situations that expected consistency relationships are violated can be recorded using the *ViolationSituation* shown in Figure 34.

Running Example of Phase 3:

This part of the running example illustrates how the examination of consistency relationships is performed. We present the results of user-defined and derived correspondences between model elements of M_a and M_b and the pseudo-codes of two algorithms that support violation identification for ExCR2 and ExCR3.

The first step of this phase is skipped in this running example. If needed, the CoreDMM presented in Chapter 3 can be adopted as the common language construct to resolve the differences between the modelling constructs of M_a and M_b. As described in the 'scope description' in Phase 1, a condition, in CoreDMM terms, means a 'failure' shown in the MCSs in M_a and means a 'basic event' in M_b. Some construction elements, such as the 'flow' or 'state' of a node in AltaRica model do not have corresponding concepts that can be mapped into CoreDMM and we do not use them during this inconsistency analysis.

Specific consistency relationships that are associated with only one specific model construction element in each model under comparison may be examined directly, e.g. 'Failure event Tree3.E3 has a unique corresponding part in Tree1'. Other consistency relationships may require user definition of correspondence between model elements for identifying violations of the consistency relationships. In this example, ExCR2 requires setting correspondences between 'condition's of M_a and M_b; ExCR3 requires deriving correspondence between 'MCS's of M_a and M_b on the basis of correspondences between 'condition's. With the defined or derived correspondences, we examine the violation of ExCR2 and ExCR3 manually or with tool support.

The correspondence pairs we set between conditions of M_a and M_b for ExCR2 are depicted in Table 10. It is apparent from the table that each 'condition' in Ma has a corresponding 'condition' in M_b. We can conclude that no violation of ExCR2 between M_a and M_b is found through the examination of the defined correspondences in Table 10.

Defined Correspondences between 'Condition's of M_a and M_b $R_{con}(x,y)$, $x \in C_a$, $y \in C_b$		
'Condition' in M_a ($x \in C_a$)		'Condition' in M_b ($y \in C_b$)
Tree3.E2:comC.cFm1	corresponds to	Tree1.E3:Block h_c fails
Tree3.E3:comA.aFM1	corresponds to	Tree1.E4:Block h_a fails
Tree3.E5:comA.aFM2	corresponds to	Tree1.E4:Block h_a fails
Tree3.E4:comB.bFM1	corresponds to	Tree1.E5:Block h_b fails

Table 10 Example Correspondences between 'Condition's in the Running Example

If the models under comparison are large in scale, we can perform the violation identification with tool support based on a algorithm for checking ExCR2 with the user-defined correspondences (as shown in Table 11).

```
Function CheckExCR2 as Boolean

// Ca is the set of conditions under inconsistency analysis in ModelA
// Cb is the set of conditions under inconsistency analysis in ModelB
//Rcon is a set of relations between members of Ca and members of Cb

violation=∅

For each x in Ca
    //find corresponding element in Cb for x
    tempRx=GetRcon(x,y)
    if tempRx = ∅
        violation =violation ∪ {(x, /)}
        // the slash symbol represents that
        //no corresponding element is identified
    else if tempRx.Gety ∉ Cb
        violation =violation ∪ {(x, tempRx.Gety)}
    endif
next x;

For each y in Cb
    //similar to finding correspondence item in Ca for y
    tempRy=GetRcon(x,y)
    if tempRy = ∅
        violation =violation ∪ {(/, y)}
    else if tempRy.Getx ∉ Ca
        violation =violation ∪ { (tempRy.Getx, y)}
    endif
next y

if violation=∅
    return true //no ExCR2 violation found
else
    return false // ExCR2 violation situations identified
endif


End function
```

Table 11 Example Algorithm for Checking ExCR2

For examination of ExCR3, we need to have derived correspondences between ‘MCS’s of M_a and M_b based on the correspondences between ‘conditions’ of two models defined in Table 10. The derived correspondence pairs we infer for MCSs in M_a and M_b are illustrated in Table 12.

Derived Correspondences between ‘MCS’s of M_a and M_b $R_{mcs}(x,y)$, $x \in MCS_a$, $y \in MCS_b$, given $R_{con}(x,y)$				
MCS_a of M_a ($x \in MCS_a$)		MCS_{ab} *		MCS_b of M_b ($y \in MCS_b$)
{Tree3.E2}	transformed into	{Tree1.E3}	corresponds to	{Tree1.E3}
{Tree3.E3, Tree3.E4}	transformed into	{Tree1.E4, Tree1.E5}	corresponds to	{Tree1.E4, Tree1.E5}
{Tree3.E5, Tree3.E4}	transformed into	{Tree1.E4, Tree1.E5}	corresponds to	
* : MCS_{ab} is an intermediate MCS set used for deriving correspondences between MCSs. It is generated by substituting the conditions in each MCS of Trees3 with the ‘condition’s in M_a according to $R_{con}(x,y)$ defined in Table 10.				

Table 12 Example Correspondences between ‘MCS’s in the Running Example

We substitute each condition in every MCS of M_a with a corresponding condition in M_b as specified in Table 10. Then the intermediate MCS set (which has the same content of the MCS set of M_a) with transformed equivalent conditions in M_b can be examined against MCSs of M_b. The correspondences of ExCR3 between the intermediate MCS set and the MCSs of M_b are viewed as the correspondences depicting the relationships between MCSs between M_a and M_b.

It is obvious from Table 12 that each ‘MCS in M_a has a corresponding ‘MCS in M_b. We can conclude that no violation of ExCR3 between M_a and M_b is found through the examination of the derived correspondences in Table 12.

We can also perform the violation identification for ExCR3 with tool support. The algorithm for checking ExCR3 with the user-defined correspondences is presented in Table 13.

```
Function CheckExCR3 as Boolean

// MCSa is the set of minimal cutsets of ModelA
// MCSb is the set of minimal cutsets of ModelB
//Rcon is a set of relations between members of Ca and members of Cb
//assuming that no one-to-n relations from a member of Ca to a member of Cb in Rcon
//a n-to-one relation from a member of Ca to a member of Cb is allowed

violation= ∅
derivedR=∅

For each mcsa in MCSa
        //get a member mcs from a set of MCSs
        mcsab= ∅
        for each c in mcsa
                //get each member condition of a mcs and substitute it
                //with the corresponding member in Rcon
                tempR= GetRcon(c,y)
                if tempR.Gety ∉ mcsab then mcsab = mcsab ∪ { tempR.Gety }
        next c

        if mcsab ∉ MCSb
                violation =violation ∪ {mcsa}
        else
                derivedR= derivedR ∪ {(mcsa,mcsab)}
        endif
next mcsa

if violation=∅
        return true //no ExCR3 violation found
else
        return false // ExCR3 violation situations identified
endif

End function
```

Table 13 Example Algorithm for Checking ExCR3

After defining correspondences and implementing the checking (with algorithms), we can find that both ExCR2 and ExCR3 have not been violated.

If we got another fault tree ‘Tree4’ as M_b instead of ‘Tree1’. The events of ‘Tree1’ are the same set as ‘Tree1’, but the MCSs are {{Tree4.E3}, {Tree4.E4}, {Tree4.E5}}, then ‘violations’ would be reported for two MCSs of M_a.

In this example, we have the same number of system elements and conditions in M_a and M_b. But the Phase 3 of the inconsistency analysis method is generic in terms of considering (in)consistent models with different numbers of model elements. In that case, the user may exclude irrelevant model elements for model comparison in the Scope Description of Phase 1, or the user may need to define and derive more sophisticated correspondences between models elements of different details. For example, a system element in M_a corresponds to two system elements in M_b due to more detailed design; or a condition in M_a corresponds to two or more conditions in M_b due to more detailed description of failure causes. The inconsistency analysis method does not cover the details in potentially more sophisticated correspondences definition and inference, which are planned as future work in Section 8.2.2.

5.4.6 Phase 4: Interpreting Comparison Results

This phase of the analysis is intended to query whether there is a reasonable explanation for the violation situations identified in the comparison process and to understand whether the identified violation situations imply flaws in the models under comparison. The result from a comparison activity for consistency checking is not only a simple Boolean value of a proposition – e.g. ‘the two models are consistent’ or ‘the two models are inconsistent’. The comparison result should include a proposition on consistency and the associated consistency relationships and, moreover, the indication of which elements in model instances violates the anticipated consistency relationships, if any. Without the details of the consistency relationships and the violation scenarios, the comparison results may not be considered in a proper context. The violation situations identified in the inconsistency analysis might be discharged by considering the claimed metadata or the claimed validity context of model instances. Or the inconsistencies identified may require further examination and update of the system safety analyses.

Figure 44 shows the interpretation process of Phase 4. The reason provided for the existence of a violation situation can be recorded using the *Explanation* shown in Figure 34.

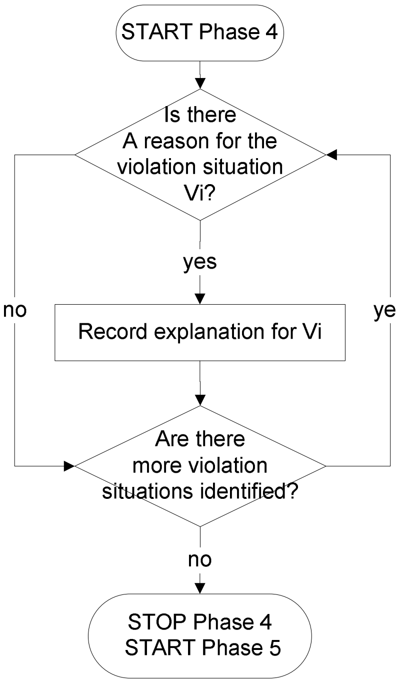

START Phase 4
Is there A reason for the violation situation Vi?
yes
no
yes
Record explanation for Vi
Are there more violation situations identified?
no
STOP Phase 4 START Phase 5

Figure 44 Phase Four Flowchart

Running Example of Phase 4:

At this point, we need to claim consistency or inconsistency according to the findings from the previous phase. In our example, no violation has been reported. We can claim that the two example models are consistent in terms of consistency relationships ExCR2 and ExCR3. Otherwise, we need to go through each of the violation situations and examine what is the reason of ‘violating’ the consistency relationships. This must be carried out through reviewing and investigating the two models with consideration of the system description.

For example, in Tree1 (M_b), if the ‘AND’ gate that connects Tree1.E4 and Tree1.E5 with Tree1.E2 was misrepresented with an ‘OR’ gate (which is a common mistake in tool-supported traditional FTA), the MCSs of Tree1 will change into {{Tree1.E3}, {Tree1.E4}, {Tree1.E5}}. Two MCSs in Tree3 (M_a), {Tree3.E3, Tree3.E4} and {Tree3.E4, Tree3.E5}, would be reported as violations due to no corresponding MCS in the misrepresented fault tree results. In this case, the reported violations reflect that inconsistent logical relationships have been presented in M_a and M_b. We would suggest revising the incorrect model and it should not be used as evidence for any safety claims.

Another example, in Tree1 (M_b), if there was another basic event Tree1.E6 (relating to a human error that leads to Tree1.E1) presented next to Tree1.E3 under the same ‘OR’ gate below the top event, then we would find that there was no corresponding condition of Tree1.E6 in M_a. In this case, Tree1.E6 in M_b would be reported as a violation situation. However, we may find a declared assumption of not considering any human errors in M_a. If so, we can tolerate the reported violation, provide an explanation, and treat it as an acceptable violation situation rather than an inconsistency.

5.4.7 Phase 5: Evaluating the Inconsistency Analysis

This phase of analysis aims to have a comprehensive review of the outputs of previous steps in order to determine whether the comparison activity has been carried out properly. Each step of the process has some domain inputs. If these inputs are improper or incorrect, the steps followed will be founded on a flawed basis and the comparison results will be naturally defective. Basically, the comparability of model instances, the coverage and reasonableness of the consistency relationships, the reasonableness of the extra intermediate information introduced in the implementation phase are re-examined. This step serves as a review that double-checks the integrity of the inconsistency analysis process and its products. This step can be informally performed by expert reviews, but the activity should be appropriately documented.

Running Example of Phase 5:

The task of this phase is to review the data generated from each of the previous phases and ensure that the illustrated inconsistency analysis has been carried out in a proper manner. The factors we have reviewed include:

- Whether two models chosen were comparable and whether the consistency relationships are clear and reasonable.
- Whether the user-defined correspondences are correct, e.g. we checked that no n-to-n correspondence relationship exists in the defined correspondences.
- Whether the checking algorithms for violation detection are correct.
- Whether the explanation of the analysis results is reasonable.

5.4.8 Phase 6: Citing and Justifying Analysis Results

The comparison results can provide us with some basis of claiming consistency between safety assessment models. For models serving as evidence for the same claim or the same primary safety argument, the comparison results from the inconsistency analysis can be adopted as items of evidence in the confidence argument associated with that primary safety argument. Model consistency confirmed by comparison activities can be used as supporting evidence for a consistency claim; whereas inconsistency identified can be used as challenges against a model consistency claim (Chapter 6 presents generic argument structures employing inconsistency analysis results as evidence.). For a compelling safety case, the inconsistencies between safety assessment models that are to be used as items of evidence in that safety case should be rigorously identified and systematically justified.

Running Example of Phase 6:

As explained in Section 5.4.7, the inconsistency analysis results can be used as evidence in confidence arguments. In this demonstrating example, the inconsistency analysis results are supporting evidence for the trustworthiness claim of the model that is used as evidence in a primary safety argument. We do not provide exemplar arguments in this section. An example argument related to inconsistency analysis will be constructed basing on the content of Chapter 6. A case study on model adequacy argument construction is presented in Appendix C.5.

5.4.9 Summary of the Inconsistency Analysis Method

The overall inconsistency analysis method is described as a standalone analysis activity. But the method we outlined above may be used in other ways, such as embedded steps as part of another safety review or safety analysis. Additionally, we may repeat all or some steps of the cross-model inconsistency analysis process several times to keep the inconsistency analysis results updated. Safety assessment models are generated periodically and will evolve with system design. They may also be revised due to inconsistencies identified. Therefore, the overall inquiry of inconsistency, in theory, only stops when we are sufficiently confident in the consistency between models based on the rigour of the inconsistency analysis.

5.5 Practicality and Capability Analysis

The idea of comparing models and detecting inconsistency between models is not new. But for a long time, the comparison of safety assessment models has been carried out in practice only as an implicit informal process. The example consistency relationships and corresponding checking presented aim to illustrate the potential power and difficulties encountered during rigorous inconsistency analysis. The method presented is more structured for dealing with large-scale safety assessment models.

Existing consistency checking practice usually does not involve two models explicitly – the content of a model is compared with some knowledge or information we got from other resources (such as a failure mode data base or past operation experience). Unlike the existing practice, the cross-model inconsistency analysis presented in this chapter focuses on the similarity and differences between models, and allows for identifying inconsistency between processed substance results such as MCS if there is not enough domain knowledge to check the reasonability of MCS directly.

The principal benefits obtained from performing consistency analysis based upon user-defined correspondences and consistency relationships are *scalability* and *rigour*. Although some effort may be required to set up the consistency checks, once established they can be applied to models of any scale. Purely manual review can struggle to identify anomalies when safety assessment models become large and complex. In addition, it is easier to examine the explicitly-stated correspondences and rules, than it is to justify an unstructured manual review (see Chapter 6). Also, once established the rules can be dogmatically and systematically applied (for example with tool support) without fear of 'slip' or omission that may occur in expert reviews and can undermine the effectiveness of manual analysis processes.

We need to stress that human decision and knowledge is crucial during inconsistency analysis. Firstly, we need to decide the comparability and consistency relationships between models. Secondly, real world semantics are not understandable by computer unless explicitly specified links were established by human to resolve the differences of language constructs and referential names. Finally, the interpretation of inconsistency analysis findings must be performed manually by human in order to taking into account information on models depicted in natural language.

Any inconsistency analysis cannot disclose all potential flawed points in models. Firstly, if two models have the same erroneous view regarding an aspect of a system, they may still exhibit a 'consistent view' of that aspect of a system according to the representation of the models. Secondly, it is impossible to provide a complete set of consistency relationships between two models. Hence, we cannot claim absolute cross-model consistency on the basis of the results of cross-model inconsistency analysis.

The power of cross-model inconsistency analysis of safety assessment is limited in terms of its support to the identification of the causes of inconsistency. The analysis results may indicate that one of the models could be wrong about a certain model element, but there will not be clues for answering such questions as which one is the flawed one, or which model is inconsistent with the reality, unless we know with certainty that which model can be deemed as a 'correct' model before implementing the comparison.

5.6 Summary

This chapter presents the definition of inconsistency between safety assessment models and a typology of safety assessment model consistency. This provides an expanded view of the consistency issue that can affect the evaluation of the trustworthiness of safety assessment models. This chapter also defines the characteristics of potential user-defined consistency relationships that describe the agreement that is expected between models. These can be used as heuristics for structuring good engineering practice for consistency checking and as a catalyst for more rigorous reviews. More importantly, a structured inconsistency analysis method is proposed and demonstrated with an example walkthrough of the analysis process. The method clarifies and decomposes previously implicit inconsistency investigation activities into six concrete phases. Although there are recognised limitations to the method presented, it provides benefits in terms of stimulating transparent, rigorous and repeatable scrutiny of the consistency checking between safety assessment models.

6 Using and Justifying Safety Assessment Evidence

6.1 Introduction

As the common basis of evidence elaborated the discussion in Section 4.2.3 indicated, the source information of evidence is neutral by itself; it is the argumentation context in which evidence participates that endows the source information with a role of being supporting evidence, counter-evidence or contextual data. In this chapter, we consider how we use evidence in argument construction and how confidence in safety cases can be improved with a more rigorous argumentation process and more structured justification of safety assessment evidence.

By nature, a safety case is *not* born to be strong and compelling and to be accepted without doubts. A typical safety case will contain positive arguments and supporting evidence and it is a developer’s ultimate goal to have a positive safety case for their developed product. However, it does not mean that we can only think positively, or admit only positive things, or present only positive points in arguments. A safety case is not composed by structured positive arguments and associated evidence for demonstrating system safety characteristics alone, especially during the early stages of evolution and when we have a rich source of information during argument construction. Firstly, it is necessary to include extra information to demonstrate our confidence in the arguments and evidence presented. Therefore, a safety case should also contain the justification or backing of existing structured positive arguments and evidence. Secondly, it is also necessary to include extra information to demonstrate why we justifiably disregard negative, or alternative, information for our positive arguments. It is commonplace that one item of source information might differ from or disagree with another item of source information. We should not neglect any relevant items of information that do not meet our positive argumentation need and we should record how those items of information are considered and why they are discharged or not selected over those items of evidence that are presented. Incorporating how negative arguments and evidence are resolved in a safety case is vital for achieving a compelling and defensible safety case. However existing argument construction practice has typically left counter-evidence probing activities until the review stage of a safety case lifecycle. In order to exploit various potential sources of evidence within argument construction, an expanded safety case construction process is presented in this chapter in order to incorporate the active consideration of counter-evidence into the development process of safety case arguments.

It is important to consider systematically how to justify the adequacy of supporting evidence employed within a safety case, especially for a very common form of evidence – namely, safety assessment models. As we have explained in Chapter 2, validating safety assessment models as 'complete' and 'correct' representations of the real world is difficult. Hence, our aim of presenting model adequacy arguments is not to 'validate' a model, or to demonstrate the correctness of the use of a safety assessment model, but instead, to establish confidence in that they are sufficient for their usage in a particular context as evidence. However, the existing standards and guidance do not provide sufficient recommendations explicitly on how to evaluate various safety assessment models and how to justify whether they provide adequate evidential support for domain claims in particular argumentation contexts. During MISSA project meetings, Mr Jean-Pierre Heckman, a safety expert from EADS APSYS, formerly from Airbus (France), detailed that there are some good codes of practice on model validation and review by safety practitioners, but they are often informally implemented and passed on by people. It is necessary to formulate methodical solutions to guide the justification of safety assessment evidence based on engineering practice and the characteristics of various models. An argument pattern for justification of the adequacy of safety assessment models is presented in this chapter.

In addition, in Chapter 5, it is shown that using multiple evidence items can be problematic – an evidence item may disagree with real world or another evidence item. The rigorous inconsistency analysis introduced in Chapter 5 provides a means of identifying further supporting evidence or counter evidence for the justification of adequacy of safety assessment models. The role of inconsistency analysis results is shown in the pattern for the justification of safety assessment evidence. In this chapter, two further argument patterns are also presented for the generic justification in the presence of counter evidence and the generic justification of the adequacy of inconsistency analysis itself.

6.2 Role of Evidence in Safety Cases

As described in Section 4.4, relationships between evidence and argument are not simple and singular. A piece of information can serve as supporting evidence for a claim, as counter evidence against a claim, or as contextual data for goal decomposition. In addition, the elements in an argument that may be linked with an item of evidence are varied, and not limited only to domain safety claims. For example, an item of evidence can support the warrant that authorises the inference between two claims; or an item of evidence can attack the trustworthiness of an item of evidence for a domain claim.

In GSN the source data of evidence is cited or referenced by solutions or contexts; whereas the role of evidence is carried by the asserted relationships between an item of evidence and other argument elements. In this section, the argument links between evidence and argument elements are revisited for a clear explanation of the roles that evidence can play in safety cases.

It is explained in Section 4.5.3 that an item of evidence can associate with other argument elements in ARM via three subtypes of argument relationships: *AssertedEvidence, AssertedCounterEvidence*, *AssertedContext*.

The three roles of evidence in safety cases are *Supporting-Evidence*, *Context* and *Counter-Evidence* (as explained in Figure 45). Each item of evidence has its data part (the content of the source information of evidence) and its proposition part. The data part can be cited as context of argument reasoning elements (e.g. *goal*, *strategy* in GSN terms). The proposition part can be cited as supporting evidence for argument reasoning elements and asserted relationship elements (e.g. *SupportedBy* in GSN terms), or as counter evidence against argument reasoning elements or asserted relationship elements.

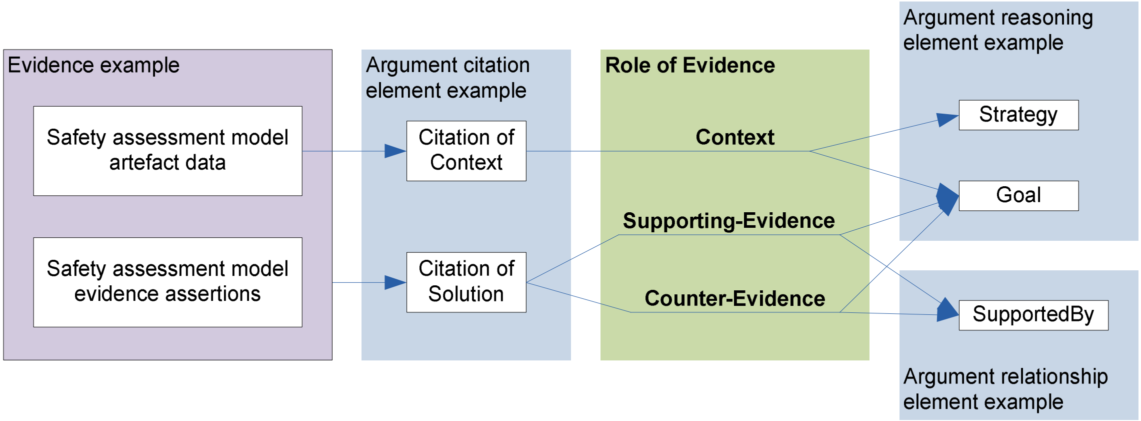

Evidence example
Safety assessment model artefact data
Safety assessment model evidence assertions
Argument citation element example
Citation of Context
Citation of Solution
Role of Evidence
Context
Supporting-Evidence
Counter-Evidence
Argument reasoning element example
Strategy
Goal
SupportedBy
Argument relationship element example

Figure 45 Illustration of Roles of Evidence

The evidential role of a piece of information is not constant:

- The same item of information is capable of serving two different evidential roles in a safety case, e.g. a hazard log can provide the context of a strategy and the supporting evidence for a goal at the same time within the same safety case.

- The same item of information can be used as supporting evidence for a claim in one safety case, but serve as *counter* evidence for another claim in another safety case, even if they both utilise the same evidence result assertion.

- The role of an item of evidence may evolve or change with further justification, further information becoming available, or as the system design changes being made. For example, a potential item of counter evidence for a claim in a primary safety argument, if sufficiently rebutted, may become context to other claims in the confidence argument associated with the primary safety argument.

6.3 Consideration of Counter Evidence

As described in Section 2.7.4, DS 00-56 [149] requires that counter evidence should be searched, documented, analysed and referenced by relevant safety claims. However, existing practice has typically only considered the role of counter evidence during the evaluation or review of safety cases, rather than actively taking it into account at the argument construction stage.

Greenwell et al [86] have defined some typical fallacies in safety cases, as presented in Figure 46. ‘Ignoring Available Counter-Evidence’ is clearly presented as a sub-category of fallacies existing in safety arguments. However, it is improper to put ‘Arguing from Ignorance’ under the type of ‘Unsupported Assertions’ because ignoring counter evidence is not in the category of ‘Unsupported Assertions’ and ‘Ignoring Available Counter-Evidence’ is one way of ‘Arguing from Ignorance’.

To be clearer, it is helpful to differentiate two aspects of using evidence in safety arguments[12]: acknowledging the existence of evidence (either supporting evidence or counter evidence) and addressing evidence properly in relevant arguments. Failures to acknowledge relevant evidence information lead to the fallacy of ‘Arguing from Ignorance’; whereas failures to address relevant supporting evidence or counter evidence lead to the fallacy of ‘Omission of Key Evidence’. To avoid ‘Arguing from Ignorance’, we should perform a sufficiently-rigorous search for both supporting evidence and counter evidence even though the search ultimately returns no results. To avoid ‘Omission of Key Evidence’, we should consider both the negative and positive evidence available for a claim with further justification.

[12] We have not yet considered the contextual role of evidence here. Ignoring or omitting relevant contextual information will make the argument unclear and difficult to evaluate. But that is not the main focus of the thesis.

Circular Reasoning

- Circular Argument
- Circular Definition

Diversionary Arguments

- Irrelevant Premise
- Verbose Argument

Fallacious Appeals

- Appeal to Common Practice
- Appeal to Improper/Anonymous Authority
- Appeal to Money
- Appeal to Novelty
- Association Fallacy
- Genetic Fallacy

Mathematical Fallacies

- Faith in Probability
- Gambler’s Fallacy
- Insufficient Sample Size
- Pseudo-Precision
- Unrepresentative Sample

Unsupported Assertions

- Arguing from Ignorance
- Unjustified Comparison
- Unjustified Distinction

Anecdotal Arguments

- Correlation Implies Causation
- Damning the Alternatives
- Destroying the Exception
- Destroying the Rule
- False Dichotomy

Omission of Key Evidence

- Omission of Key Evidence
- Fallacious Composition
- Fallacious Division
- Ignoring Available Counter-Evidence
- Oversimplification

Linguistic Fallacies

- Ambiguity
- Equivocation
- Suppressed Quantification
- Vacuous Explanation
- Vagueness

Figure 46 The Safety Argument Fallacy Taxonomy (from [86])

Concerning the insufficient attention paid to counter evidence in existing argument development processes, it is necessary to consider how counter evidence should be integrated into the traditional positive argument construction process. We find that counter evidence is associated with a primary safety argument in four ways.

Firstly, the primary argument may be challenged by counter evidence in the inference links between two domain claims, if we have evidence to falsify the rationale or warrant for an inference relation between two claims or to demonstrate the insufficiency of support, even if all lower level domain claims were true.

Secondly, a domain safety claim can be challenged by an item of counter evidence if the evidence result assertion disagrees with the domain claim and demonstrates that the domain claim is a false proposition. In this situation the counter evidence is a challenge to the other items of supporting evidence for that domain safety claim as well, because they are inconsistent regarding the conclusion on the Boolean value of a domain claim.

Thirdly, the items of evidence presented in the primary safety argument may be challenged by counter evidence in terms of their trustworthiness.

Lastly, the appropriateness of the usage of items of evidence may be questioned. This challenge affects the validity of the argument links between items of evidence and domain safety claims being supported. This is different from the challenge towards the inference links purely from the argument structure point of view. The asserted evidence relationship considers both the relevance and fitness of the evidence result assertion towards a domain claim.

The first way of challenging primary safety arguments focuses on the validity of the argument structure. The other three ways of challenging focus on that whether the Boolean value of a domain claim has been demonstrated sufficiently to be true with sufficient confidence. All four kinds of challenges ought to be addressed in a confidence argument rather than in a primary safety argument.

Govier recognises the existence and function of *counterconsiderations* [85], which are the negatively relevant points towards a conclusion within arguments. In a natural language argument, these *counterconsiderations* are usually introduced with signposts such as 'although', 'though', 'even though' or 'despite the fact that'. *Counterconsiderations* depict challenging or weakening effects on conclusions. They are represented by Govier as wavy lines between premises and a conclusion [85] (as shown in Figure 47).

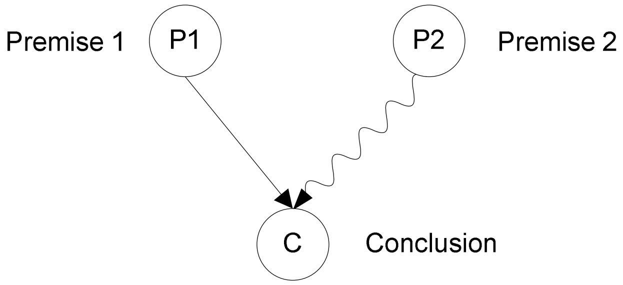


Figure 47 Counter Consideration Representation

During the articulation of an argument, we must acknowledge and present these negative points even though they do not support our claims. Actually, if we could reason that the supporting points for a claim outweigh the negative points or the negative points are reasonably discharged, the presence of *counterconsiderations* in an argument would make the argument more defensible. Considering counter evidence is part of the *counterconsiderations* that we can apply in safety case development to avoid some of the safety argument fallacies that Greenwell identified.

6.4 Presentation of Counter Evidence

Counter evidence has received some consideration in the literature on safety case practice. For example, Spriggs explains how to consider counter evidence in safety case development in [188]. He also states that the *lack or absence of counter evidence* recognised through proactive campaigns can *support* a Goal in safety cases. GSN does not contain specific symbols that describe counter evidence or the potential negative role of the source information of evidence. Some researchers have suggested extensions to managing counter evidence. Cockram [54] proposes a negation symbol on the *Solution* symbol in GSN (as shown in Figure 48 (a)) and Johnson [108] adopts a cross symbol to indicate the refuting role of a *Solution* in GSN instead (as shown in Figure 48 (b)).

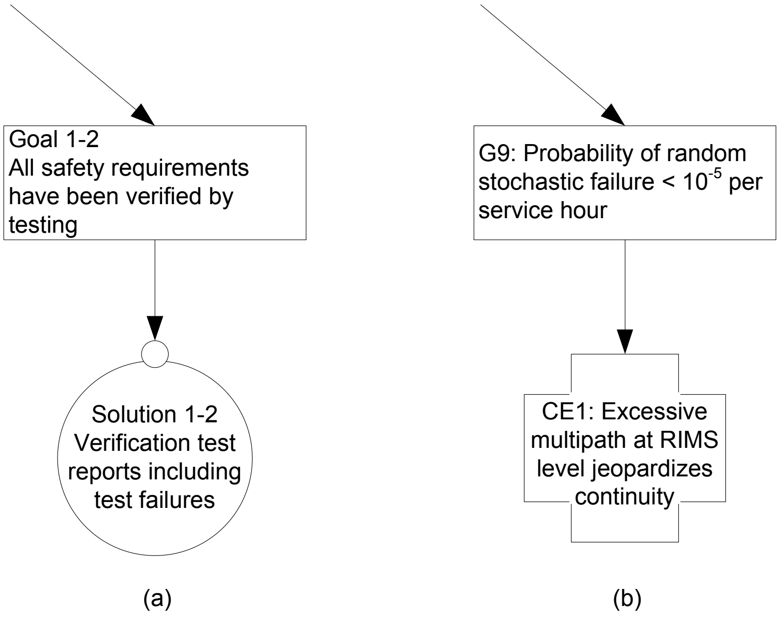
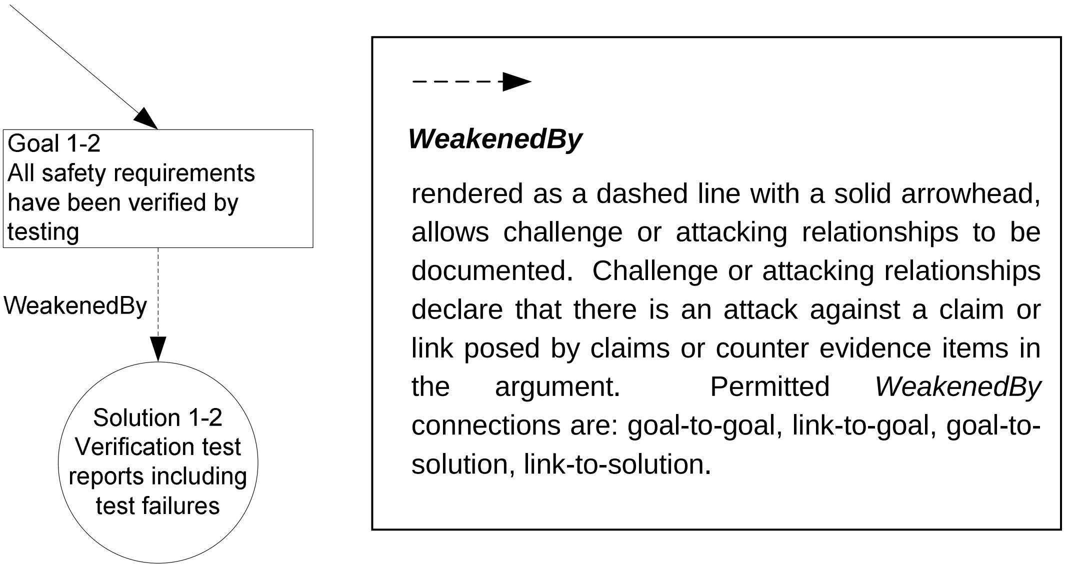

Goal 1-2
All safety requirements have been verified by testing
Solution 1-2
Verification test reports including test failures
(a)
G9: Probability of random stochastic failure < 10-5 per service hour
CE1: Excessive multipath at RIMS level jeopardizes continuity
(b)

Figure 48 Representation of Counter Evidence ((a) from[54] and (b) from [108])

However, as we have stated in Chapter 4, it is the relationship between a claim and the source data being adopted as an item of evidence that carries the evidential force of ‘support’ or ‘challenge’. Therefore, the *Solution* symbol is not the right subject to expand or change to represent counter evidence. Instead, we will indicate items of counter evidence in safety cases by adapting the symbol of *SupportedBy* relationship in GSN. Figure 49 illustrates a dashed line with a filled arrow that is used to present the challenge posed by a piece of counter evidence of a goal in goal structures. This visually denotes the *AssertedChallenge* and *AssertedCounterEvidence* relationships between argument elements, as specified in ARM [156].


Goal 1-2
All safety requirements have been verified by testing
WeakenedBy
Solution 1-2
Verification test reports including test failures

WeakenedBy

rendered as a dashed line with a solid arrowhead, allows challenge or attacking relationships to be documented. Challenge or attacking relationships declare that there is an attack against a claim or link posed by claims or counter evidence items in the argument. Permitted *WeakenedBy* connections are: goal-to-goal, link-to-goal, goal-to-solution, link-to-solution.

Figure 49 Representation of Counter Evidence Relationship

It is worth noting that for a signed-off[13] and accepted safety case, the counter evidence relationships in the safety case, usually, should be sufficiently discharged. *Discharging* an item of counter evidence means that the concerns raised by that item of counter evidence have themselves been rebutted (rejected), undermined (i.e. shown to be irrelevant) or resolved. Identified (potential) counter evidence, has an attacking role during intermediate stages of safety case development, but serves as 'evidence' of rigorous thinking in safety case construction for the safety case presented to an assessor. If it is presented as unresolved counter evidence, the presence of counter evidence relationships in a safety case submitted for review could reflect that there are potentially unsupported safety goals and the confidence in safety demonstrated can be undermined.

6.5 Safety Case Processes

In order to use evidence properly and to its best potential in developing safety cases, we need to first have a general view of the processes associated with existing safety case development. This section presents a review of supporting processes of safety case development and describes a widely-used argument construction method.

6.5.1 Existing Processes

In the 1990s, graphical argumentation notations emerged in the development of safety cases, which bring about more clarity to the presentation of structured arguments. In addition, the

[13] A 'signed-off' safety case means an approved safety case that has passed through scrutiny by experts on behalf of the developer's organisation or certain authorities.

introduction of structured argumentation processes enforces the explicitness of previously implicit information elements in safety case construction, such as hidden context, supporting rationale of inferences, or assumptions. The six-step method [117] of goal structure construction is a typical structured argument development process.

Along with structured safety case construction, there are other processes defined in support of safety case management in a system safety lifecycle, e.g. a process for safety case change management [120], a process for safety case reuse with safety case patterns [116], a process for argument review [112], or a safety case life cycle with safety case submission and acceptance [152].

An enhanced safety-case lifecycle is presented in [87] to revise imperfect safety cases with failure evidence from the system operational phase. Sets of critical questions have been proposed as safety argument schemes for challenging typical safety arguments to help the review and evaluation of safety cases [215]. However, both of these approaches are 'after-the-event' methods, which may lead to late modification and reworking in a project lifecycle.

Although there are many processes related to safety cases, as described above, the construction process is the essential one in the safety case domain, in which the robustness of a safety case is rooted. However, the *rigour* of this process has not been exploited sufficiently in existing practice, partially because the role of *counter-evidence* is not properly addressed as it should be. With regard to processes orientated around evidence employment, the issue of counter evidence is also insufficiently considered. For example, the evidence selection and justification process in [96], pays insufficient attention to the items of evidence that have evidence result assertions that refute the truth of a domain claim, which might bring the 'Ignoring Available Counter-Evidence' fallacy into safety cases. Instead, collecting relevant counter evidence in parallel to the selection of supporting evidence during construction of safety cases provides an opportunity for defending evidence selection in a rigorous manner.

Therefore, it is necessary to have a more active way to take account of *counterconsiderations*, especially *counter evidence* in safety argumentation.

6.5.2 Six-Step GSN Method

Kelly proposes a six-step method [117] of safety case construction in GSN (for a detailed description of the method, the reader is referred to [89]). The method is systematic, especially in providing assistance to constructing arguments positively. The steps of the method [117] are presented in Figure 50.

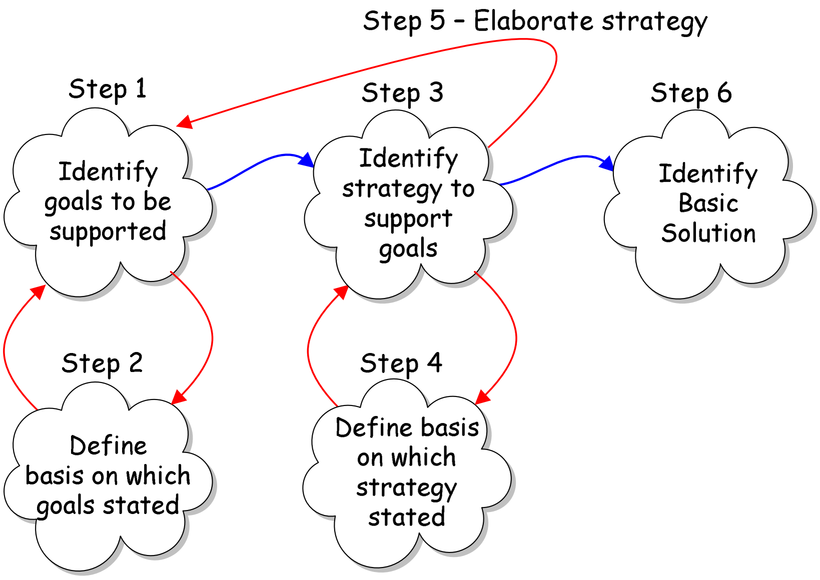

Step 5 – Elaborate strategy
Step 1
Identify goals to be supported
Step 3
Identify strategy to support goals
Step 6
Identify Basic Solution
Step 2
Define basis on which goals stated
Step 4
Define basis on which strategy stated

Figure 50 Six Steps of Argument Construction in GSN (from [117])

Step 1 - *Identify the goals to be supported;*

Step 2 - *Define the basis on which goals are stated;*

Step 3 - *Identify the strategy used to support the goals;*

Step 4 - *Define the basis on which the strategy is stated;*

Step 5 - *Elaborate the strategy (and proceed to identify new goals – back to Step 1);*

Step 6 - *Identify the basic solution.*

These steps can be repeated as many times as is necessary to produce a complete argument where all of the goals in the argument have been supported. This method supports structured safety case construction and enforces elicitation of unstated rationales or context for a clear and understandable argument. However, assurance of the strength of the arguments produced is not considered. Weaver expanded the six-step method through including the use of SALs (Safety Assurance Levels) in Step 2, Step 4 and Step 6 [204] to take into account assurance of structured argument elements. Hawkins et al have performed HAZOP-style deviational analysis on the six steps of the GSN method to consider assurance deficits in argument construction [98]. Although their work brings more thought during argument construction, no new steps are introduced to the process to increase the rigour of potential argumentation and the issue of counter-evidence is not directly addressed.

Counter evidence may be in existence already (whether we are aware of it or not) at the point of citing evidence in safety cases and the total set of obtainable evidence should be carefully examined and considered while we are choosing items of evidence for a claim and justifying their adequacy of being supporting evidence.

In the next section, an expanded argument construction process based on the six-step method is defined in order to incorporate both assurance of positive argument elements and counter evidence (from various relevant information sources) comprehensively during safety case development.

6.6 Expanded Structured Argument Construction

6.6.1 Overall Expanded Process

From the discussion in Section 6.4, we set two objectives for an expanded GSN argument construction process: a) Clear integration of the construction of confidence arguments with the construction of a primary safety argument; b) Clear integration of the construction of pure positive arguments with the consideration of *negative positive arguments* that put forward positive arguments from an opposite viewpoint with sufficient consideration of opposing points.

The generic six-step method can be applied to either confidence argument construction or primary safety argument construction. The nature of both types of arguments is the same, but the role or function of the arguments differs. The function of confidence arguments is to assure the soundness of a primary safety argument.

The *negative positive arguments* intend to argue from an opposite viewpoint. In informal argumentation, e.g. essay writing or tribunal presentation, arguing from two sides is common practice. When an issue is identified as a topic to argue about, both reasoning sides of the issue, supporting and challenging, should be explored and considered. This entails more transparency of why one side outweighs another and makes the argument more balanced, convincing and defensible. The six-step process does not contradict the performance of two-sided argumentation, but it had not highlighted the importance of arguing from the negative side explicitly.

Figure 51 outlines the expanded process of structured argument construction in GSN. The expanded steps in the development of safety cases in GSN are:

Step E1: Justify the basis of goals

Step E2: Identify the alternative basis on which goals are stated

Step E3: Justify the unsuitability of the alternative basis of goals

Step E4: Justify the basis of the strategy

Step E5: Identify the alternative basis on which the strategy can be stated

Step E6: Justify the unsuitability of the alternative basis of a strategy

Step E7: Justify the adequacy of solutions

Step E8: Identify counter evidence

Step E9: Justify the discharge of counter evidence

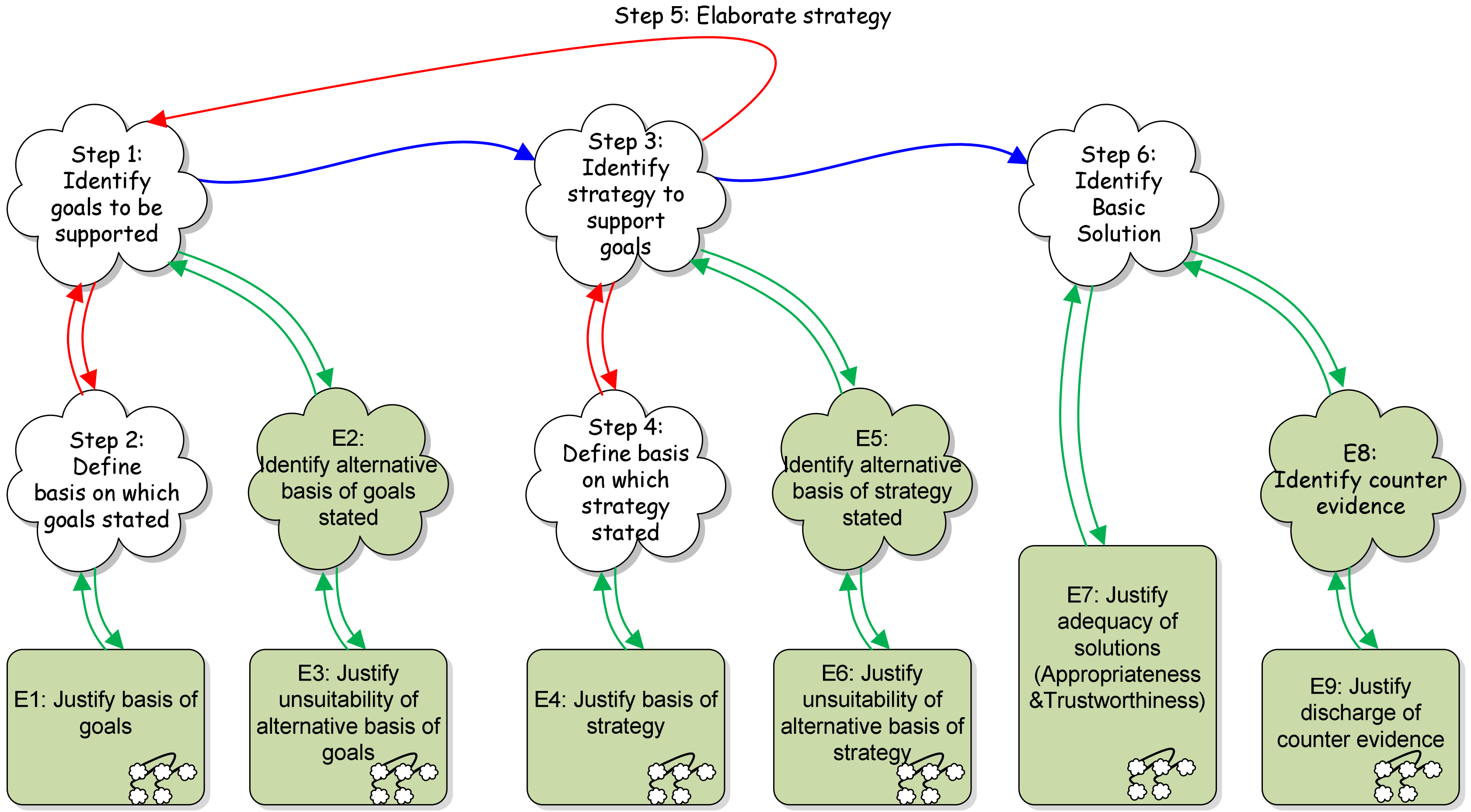

Step 5: Elaborate strategy
Step 1: Identify goals to be supported
Step 3: Identify strategy to support goals
Step 6: Identify Basic Solution
Step 2: Define basis on which goals stated
E2: Identify alternative basis of goals stated
Step 4: Define basis on which strategy stated
E5: Identify alternative basis of strategy stated
E8: Identify counter evidence
E7: Justify adequacy of solutions (Appropriateness &Trustworthiness)
E1: Justify basis of goals
E3: Justify unsuitability of alternative basis of goals
E4: Justify basis of strategy
E6: Justify unsuitability of alternative basis of strategy
E9: Justify discharge of counter evidence

Figure 51 Overall Expanded Process of Structured Argument Construction in GSN

As Figure 51 indicates, the six core steps still serve as the main frame of argument construction, but are enriched with more explicit consideration of assurance of the strength of a primary safety argument. The expanded steps are presented in shaded blocks in Figure 51. For example, while identifying solutions (Step 6), we should consider the justification of the adequacy of selected solutions (E7); in the meantime, we should also search for potential counter evidence (E8) and consider its effects on the goal to be supported (E9).

Due to the common nature of argumentation, some expanded steps are, in fact, embedded whole and additional argumentation processes in their own right. Those steps -Step E1, E3, E4, E6, E7 and E9- are depicted with rounded-corner rectangles rather than clouds in Figure 50.

The overall expanded steps can be grouped into three categories:

- For providing backing of defined argument elements related to positive arguments **– E1, E4** and **E7;**
- For recognising existence of alternative information source or counter evidence that are relevant to defined argument elements **– E2, E5** and **E8;**
- For presenting how the alternative information, usually negative, is discharged from affecting the positive arguments **– E3, E6** and **E9**.

In the following sections, each of the expanded steps is described with more details.

6.6.2 Step E1: Justify Basis of Goals

This step expands Step 2 of the six-step GSN method. The objective of this step is to provide explicit explanation of the basis of identified goals if necessary. As described in the GSN community standard, Step 2 is used for ensuring the reader has an “adequate and correct understanding of the context” [89] surrounding the goals identified. Most items of contextual information are obtained from various sources in system lifecycle as artefacts of safety activities. In some cases, we may need to provide further reasons for their suitability.

Figure 52 from the GSN standard [89] depicts an example of contextual information of a safety claim. In this example, we may need to justify that the system implementation activities described in Ref Y (C1 in Figure 52) are the right versions in use for the system under study. We may also need to justify that the safety principles from Ref Z (C3 in Figure

52) are fit for use for the system under study due to historical good practice. We may not worry about C2 in Figure 52 because it can be viewed as a factual description.

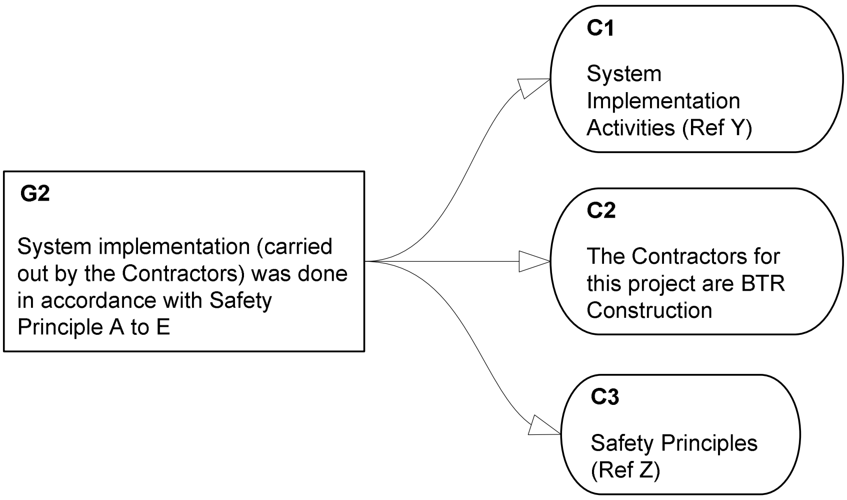

G2
System implementation (carried out by the Contractors) was done in accordance with Safety Principle A to E
C1
System Implementation Activities (Ref Y)
C2
The Contractors for this project are BTR Construction
C3
Safety Principles (Ref Z)

Figure 52 Example of Basis of a Goal (from [89])

In reality, it may be impractical to address every subtle detail in the goal definition and contextualization process. But it is valuable to explicitly present justification when we think further explanation is helpful for a clearer understanding of the basis of the identified goal.

6.6.3 Step E2: Identify Alternative Basis of Goals Stated

This step expands Step 1 of the GSN method and operates in parallel to Step 2 of the GSN method. This step aims to encourage the user to explore alternative basis that may contextualize identified goals. Different from Step 2, Step E2 focuses on active search from potential contextual information and critical thinking, especially if there exist two or more information items that are all capable of providing the same kind of necessary context for an identified goal.

For example, returning to the example presented in Figure 52, if there is another reference guideline or document (let us call this Ref AX) in the same industry domain (or in similar system development lifecycle) that defines more safety principles than Ref Z does, we need to acknowledge the existence of this document and record it for further analysis in Step E3.

6.6.4 Step E3: Justify Unsuitability of Alternative Basis of Goals

This step expands Step E2 and prompts explicit explanation of why the alternative basis of identified goals is not suitable for the argument. This step, as a result, justifies the suitability of the defined basis (obtained in Step 2) of goals from another perspective, which is in contrast to the justification presented in Step E1. Step E2 and Step E3 are not exactly thinking from a negative perspective, but they are a form of active thinking which contributes to confidence establishment in our decision making process. Following the example for Step E2, the reason for unsuitability of the safety principles presented in Ref AX as the basis of *Goal* G2 should be elicited.

Sometimes, the reasoning of Step E1 and Step E3 occurs simultaneously (with E2 providing necessary contextual information for E3). It is feasible to merge E1, E2 and E3 if the user would like to have a more succinct view of the confidence argument associated with the three steps. However, for a transparent and clear incorporation of two-sided arguments, we insist on depicting them separately in the expanded process. The same thought is applied to the separation of E4, E5 and E6 and the separation of E7, E8 and E9 respectively.

6.6.5 Step E4: Justify Basis of Strategy

This step expands Step 4 of the GSN method. The objective of this step is to prompt explicit explanation of the basis of an identified strategy if necessary. Step 4 is used for ensuring a reader understand the basis of an identified strategy and helping the assessment of the reasonableness of the strategy. A flawed basis of a strategy may result in decomposing arguments into flawed sub-goals which may make the inferences between sub-goals and a higher-level goal invalid or insufficient.

The basis of a strategy may include three types of contextual information: a) *Context* elements which provide the details of a term used in the strategy description (e.g. the identified hazards); b) *Assumptions* made while adopting a strategy (e.g. the divide-and-conquer strategy works with the assumption that the safety of sub-system components considered individually can fully represent the safety of a composed system if the interaction between subsystem elements is not shown as a sub-goal); c) *Justifications* needed for adopting a strategy (e.g. product X is suitable to be developed according to standard X, the customization or selection of requirements from standard X is adequate). They all link with a *Strategy* by an asserted *InContextOf* relationship.

Figure 53 presents an example of contextual information used as a basis of a safety strategy (adapted from [89]). In the example, the basis of *Strategy* S1 in Figure 53 is *Context* C4. We may need to justify that the 'Hazard Log HL_x' (shown as cited document in *Context* C4 in Figure 53) is a sufficiently comprehensive and adequate documentation of system operational hazards. In addition, we may be uncertain about whether we have defined the proper basis of S1. We may ask ourselves that "Has HL_x been reviewed for its completeness?" or "Has a specific critical hazard been addressed by HL_x?". The answers to these questions can help establish greater confidence in the appropriateness of the defined basis of a strategy.

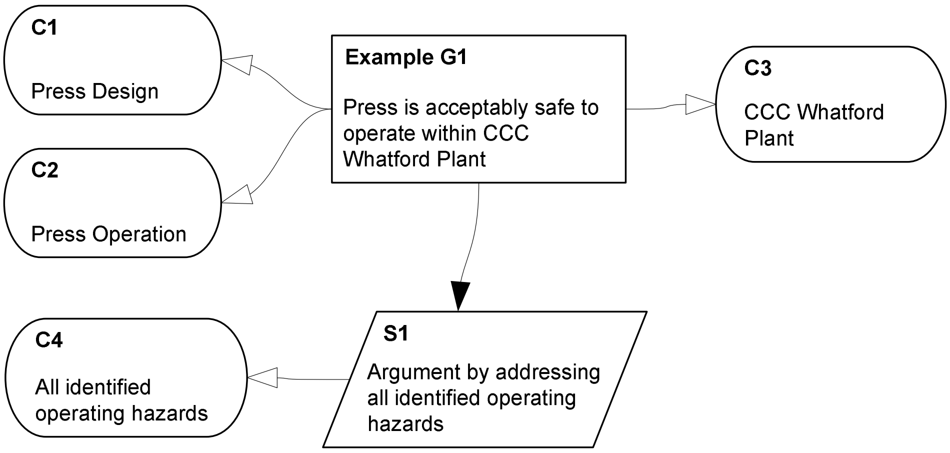

C1
Press Design
C2
Press Operation
Example G1
Press is acceptably safe to operate within CCC Whatford Plant
C3
CCC Whatford Plant
C4
All identified operating hazards
S1
Argument by addressing all identified operating hazards

Figure 53 Example of Basis of a Strategy (adapted from [89])

Similar to Step E1, it may be impractical to deal with all of the subtle detail in our strategy definition and contextualization process in real practice. In some cases, the strategy or the contextual basis of a strategy is evident by itself, and then this step can be skipped. In some other cases, especially when a complicated item of evidence is cited in the defined contextual basis of a strategy, this step is important and should not be neglected.

In the original GSN method, the justification of the strategies can be considered as a part of Step 4 [117]. However, presenting it as a separate step will make it clearer that there might be significant further argumentation needed after Step 3 and Step 4.

6.6.6 Step E5: Identify Alternative Basis of Strategy Stated

This step expands Step 3 of the GSN method and operates in parallel with Step 4 of the GSN method. The objective of this step is to stimulate the user to explore possible different basis that may contextualize an identified strategy. Different from Step 4, Step E5 places emphasis on active inquiry of potential contextual information, especially if there exist two or more information items that are all capable of providing the same kind of necessary context for an identified strategy, but maybe with different or even inconsistent concrete content.

For example, in the example presented in Figure 53, if there is another hazard log HL_y that documents various hazards of CCC Whatford Plant, we need to recognise the existence of this document and record it for further analysis in Step E6.

It is *not* our intention to identify alternative strategies in this step. The issues related to strategies in argument construction are discussed further in Section 6.6.11. Typical strategies for goal decomposition from good practice may have been documented in safety case patterns. However, the instantiation of the basis of some strategies may be problematic due to the contextual data being used. The basis of a strategy could have considerable impact on the adequacy of specific sub-goals generated.

6.6.7 Step E6: Justify Unsuitability of Alternative Basis of Strategy

This step expands Step E5 and prompts an explicit explanation of why the alternative basis of an identified strategy is not suitable for the argument. In this step, the suitability of the defined basis (obtained in Step 4) of a strategy is justified from another perspective, which is in contrast to the justification presented in Step E4. Step E5 and Step E6 are not exactly thinking in an opposing way, but they represent the efforts we have invested in having a more cautious consideration of possibilities in order to obtain more confidence in the subsequent elaboration of the defined strategy. Following the example for Step E5, the reason for unsuitability of the operational hazards presented in HL_y as the basis of a strategy should be provided.

6.6.8 Step E7: Justify Adequacy of Solutions

Safety arguments without evidence are ungrounded; but evidence without justification is unconvincing. Therefore, it is demanding and challenging to justify the adequacy of safety evidence in safety cases.

This step expands Step 6 of the GSN method, which defines a direct reference to evidence data sources and indicates that the data is asserted evidence for an identified goal. It is concerned with providing sufficient confidence in the appropriateness and trustworthiness of the identified solutions. Previously, the justification of the adequacy of evidence was not explicitly separated as an argumentation step and the associated goals and evidence were presented together with argument and evidence for domain safety goals. However, as described in Section 2.7.5, it is stated in recent work [97] that the separation of primary safety argument and evidence and confidence arguments enables greater clarity than the traditional single argument structure. Adding this explicit step naturally suits this purpose of distinguishing the two types of arguments. Furthermore, this step signifies the importance of evidence justification in a way that users cannot overlook.

Depending on the complexity of the information that is contained in an item of evidence, the justification of the adequacy of a solution could itself be another complex argumentation process. The content associated with this step can be packaged as an argument module [119] to ease the management of argument elements and for the sake of a clearer view of its role in the overall confidence argument associated with a primary safety argument. The argument information needed in this step might be obtained from the evaluation of items of safety evidence, e.g. human expert reviews, or particular tests or comparison analysis of items of evidence, such as the inconsistency analysis described in Chapter 5. A generic safety case pattern on model adequacy justification is presented in Section 6.9.

As described in Section 6.5.1, some steps are embedded argumentation processes by themselves (all rectangular blocks with rounded corners in Figure 51). We can illustrate a typical example in this step. For example, assume that a fault tree model was adopted as supporting evidence for a goal. While justifying its adequacy of providing support to the goal, we may use the expert review report of that fault tree model, in which the coverage of conditions of the fault tree is checked and claimed as good enough. However, we may also identify some negative evidence, such as an omitted condition through inconsistency analysis between that fault tree and a relevant FMEA. In this case, both items of evidence - the review report and the inconsistency analysis result - should be addressed in the justification of the adequacy of the fault tree.

6.6.9 Step E8: Identify Counter Evidence

This step expands Step 6 of the GSN method. This step aims at leading active and rigorous exploration of potential items of counter evidence that may challenge the fulfilment of the identified goal. In fact, we can carry out this step simultaneously with Step 6 of the GSN method. Both of the steps need to work from the available knowledge and information sources that we have access to. Knowledge and data are always valuable assets in a system project lifecycle. Neglecting or ignoring relevant information for any argument elements (either supporting or challenging items) may lead to a partial or even biased view of the system and its critical features. This is a significant concern in the safety domain, in which any relevant analysis, test, or operational data items are precious and should be considered and exploited sufficiently. Examining carefully various development information and safety artefacts is a critical task both for identifying adequate supporting evidence and for recognising relevant counter evidence.

Step E8 may have a wider scope of searching than the information span that can be used in Step 6. Within Step 6, we primarily explore relevant knowledge and data from the safety artefacts of a system under study; whereas within Step E8, the counter evidence for a solution may come from analysis artefacts of *similar* systems, operational records of similar systems, or accidents reports and recommended remedies and practice in a specific domain. The range of accessible information and, more importantly, the knowledge and experience of the person who carries out the task of counter evidence identification, are critical for the rigour of the outcome of Step E8.

It is possible that we may not identify any counter evidence. In this case, we should still document the efforts of searching for counter evidence, which documentation can itself be used as supporting evidence of the absence of counter evidence for a goal. Evidence of the absence of negative points is necessary; but the absence of relevant evidence, due to negligence or ignorance, is undesirable and should be avoided.

The evidence result assertion of an item of supporting evidence identified in Step 6 must fit the goal to be supported strictly in terms of the relevance of its subject, the supportive force, and the acceptable context. In contrast, while searching for items of counter evidence, we may not be so strict with the evidence result assertion of an information item. An item of evidence must be marked out as counter evidence if any of its evidence assertions (result or descriptive) disagree with the goal identified.

Step E8 is an important step to explicitly bring in the counter considerations into safety arguments, which will entail a two-sided intermediate safety argument. But the ultimate view of the top level goal is dependent on further analysis and justification of both the supporting evidence and the counter evidence, or even further design and implementation changes that are needed to resolve the impact imposed by the identified counter evidence.

As we know that a safety case is a living document. Although the identification of counter evidence is presented in the expanded process only as a step in argument construction, it does not mean that it is merely considered during argument construction or it is performed once-for-all for each safety goals. Similar thinking processes should be embedded in the whole safety case lifecycle and be stressed by each safety management system of safety-critical systems.

6.6.10 Step E9: Justify Discharge of Counter Evidence

Expanding Step E8, this step emphasises the provision of explicit explanation as to why the counter evidence identified does not refute the fulfilment of a goal in the end. In such situation, counter evidence is no longer used as a challenger, but instead, it is presented as contextual information in the justification. The justification is actually also supporting evidence for a rigorous and robust safety argumentation process. This step is an *idealized* resolution of identified counter evidence. The step *does not* imply that it will necessarily be possible to discharge all items of counter evidence.

Counter evidence identified may be discharged for many different reasons.

- If the counter evidence is shown irrelevant to our safety goals
- If the system is improved regarding the issues raised by the counter evidence
- If the counter evidence is flawed and the flaw in it disables its attacking capability
- If the counter evidence is from unreliable sources

It is impossible to list out all potential reasons for discharging an item of counter evidence from its negative role. But, the consensus is that they can only be discharged with sufficient and sound justification and it is not guaranteed that every item of counter evidence identified can be discharged from its negative role. If an item of information remains as counter evidence in a safety case, we need to admit this fact. It denotes that there are residual, unresolved counter considerations in the safety case, which can undermine our confidence in certain aspects of the system safety.

It is important to separate Step E7, Step E8 and Step E9. Because counter-evidence for a domain safety claim is only one way of attacking argument; in E7, the justification may need to consider more kinds of counter evidence for the relationship between a solution and a goal. So the issues addressed by E7 and E9 may overlap, but are not exactly identical, it is better to think with separated steps and, if needed, refer to an argument module for the reused parts of the justifications.

6.6.11 Rethinking of Strategy

There is only one step in the GSN method that has not been expanded – Step 5 Elaborate strategy. In this section, we rethink the function and nature of a strategy and explain what should be further considered in Step 3 and Step 5 of the original GSN method beyond the defined expanded steps.

A strategy “describes the nature of the inference that exists between a goal and its supporting goal(s)” [89]. A strategy “adds further detail to” or “describes the approach adopted” in a goal decomposition [117].

So the nature of a strategy is a kind of narrative description that explains goal decomposition or an inference step. It is not a claim that depicts a True/False statement. It is also not a warrant (in Toulmin’s model) that can ‘authorise’ the inference. But from the description provided by a strategy, a user can understand more easily why a set of sub-goals are presented to support a higher-level goal.

Through adopting a strategy, goals are decomposed into more concrete and tractable ones to be addressed in engineering practice. Actually, a strategy will shape the direction and nature of the supporting goals of a higher-level goal during argument development. But the solution space of a higher-level goal can be cut down due to the use of a strategy as well. Adoption of different strategies for a higher-level goal may lead to different sets of evidence items. If we have two parallel strategies (e.g. Figure 54), sometimes, we can identify that they can lead to different types of evidence items (e.g. G3 may be addressed by an inspection report or a state machine analysis report; G5 may be addressed by the adoption of a specified design measure as the standard required).

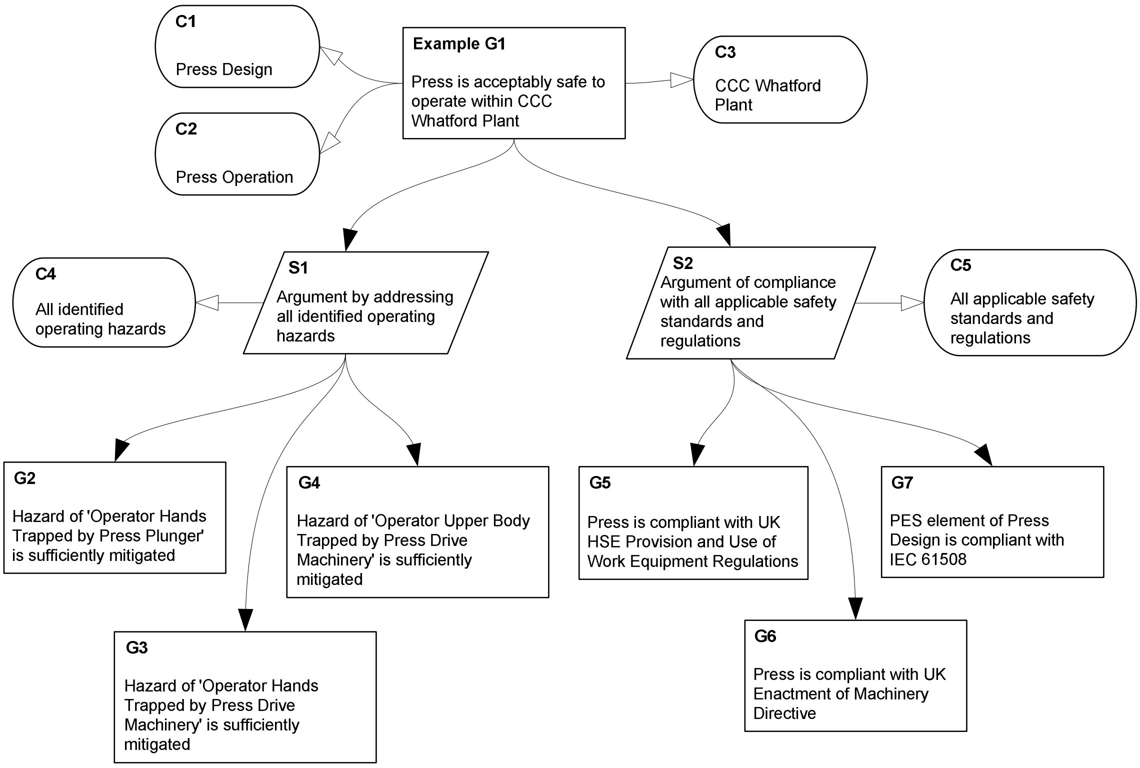

C1
Press Design
C2
Press Operation
Example G1
Press is acceptably safe to operate within CCC Whatford Plant
C3
CCC Whatford Plant
C4
All identified operating hazards
S1
Argument by addressing all identified operating hazards
S2
Argument of compliance with all applicable safety standards and regulations
C5
All applicable safety standards and regulations
G2
Hazard of 'Operator Hands Trapped by Press Plunger' is sufficiently mitigated
G4
Hazard of 'Operator Upper Body Trapped by Press Drive Machinery' is sufficiently mitigated
G3
Hazard of 'Operator Hands Trapped by Press Drive Machinery' is sufficiently mitigated
G5
Press is compliant with UK HSE Provision and Use of Work Equipment Regulations
G7
PES element of Press Design is compliant with IEC 61508
G6
Press is compliant with UK Enactment of Machinery Directive

Figure 54 An Example of Parallel Strategies (from [89])

Therefore, we have more issues to be considered in Step 3 and Step 5 of the original GSN method.

Step 3- identify strategy to support goals[14] . At this step, we may need to consider potential strategies for a goal, rather than simply adopting the first strategy to be identified. It is worth noting that more than one strategy may exist and they can be used individually or in

[14] The strategy does not support goals by itself, but it supports goal decomposition and provides the viewpoint of the decomposition and introduces the contextual basis of goal decomposition.

combination. Some of them may lead to sub-goals that can be supported by the same set of evidence items; whereas some of them may lead to sub-goals to be supported by different types of evidence. The selection of a strategy may be based on common practice, standard requirements, or the features of the system in question (can developed with typical strategies listed). However, we should acknowledge that the solution space associated with one strategy may be different from the space associated with another strategy and be aware of the risk of *omission* of some potential viewpoints of a higher-level safety goal caused by the strategy-adoption decision. (One interesting point is that if there is a case that different strategies lead to sub-goals that contradict each other, it indicates that the definition or understanding of the higher level safety goal is insufficient, ambiguous, or inconsistent.)

Step 5 – Elaborate strategy. Elaborating a strategy involves putting forward lower-level goal statements appropriately according to the contextual basis of a strategy. This step involves the elicitation of all relevant sub-goals of a higher-level goal based on the defined strategy. At this step, we should also consider or trigger the justification of the inference from the collection of sub-goals to a higher-level goal. Due to that the main focus of the thesis is evidence in a safety case; we have not presented an explicit expanded step to Step 5 of the GSN method for the justification of inferences. Moreover, the location of the potential expanded justification step is difficult. The justification of the inference should be done for a small branch of argument (which involves Step 1, 3 and 5) rather than for each single *SupportedBy* relationship, because the asserted inference is in fact a many-to-one relationship formed from the composition of all *SupporteBy* relationships for a given goal.

6.6.12 Practicality Analysis

Two primary factors inherent in the expanded process may make the application of the process challenging. First, the user may be unfamiliar with alternative information sources or consideration of counter evidence. Without sufficient knowledge, experience and available relevant information sources, it would be difficult to implement those expanded steps for assurance. For example, the relevant source information of potential evidence (for and against a claim) may scatter around the system development process, which makes the implementation of Step E8 difficult and inconvenient. This, in turn, presents us with a demand for a well-organized evidence inventory or repository. In a well-informed evidence inventory, the evidential content of an item of source information, such as its evidence result assertions (as defined in the model of evidence in Chapter 4), should be clarified as clearly as possible in order to help the identification of relevant counter evidence.

Secondly, the user of the expanded process may be ‘frightened’ by the demands of constructing two-sided arguments and supplying substantial justification simultaneously. However, as long as the two-sided thinking is performed in the construction process, we can appeal to ‘modular arguments’ [119] for addressing some lines of arguments in a light way first with a placeholder (which is an empty argument module) and to have them developed in detail later if necessary.

6.6.13 Relations with Confidence Argument

As described in Section 2.7.5, Hawkins et al [97] define a confidence argument as the argument that justifies the sufficiency of a safety argument that documents argument and evidence adopted for establishing a domain safety claim. However, the confidence argument in [97] has not covered negative positive arguments for confidence introduced in the expanded process. It should be noticed that the overall confidence in a safety case can be established on two bases: the mitigated and controlled uncertainty of a positive argument (as the focus of [97]) and the mitigated and controlled uncertainty of potential attacks to the positive safety argument.

Taking into account the opposing side of the argument construction in the expanded process, we can refine the product-branch of our framework of confidence in safety cases (see Section 4.6.1).

As illustrated by Figure 55, confidence based on justified argument elements can be divided into two types – confidence established on the adequacy of the adopted arguments and confidence established on justified unsuitability of alternatives. Similarly, confidence based on justified evidence can also be divided into two types - confidence established on the adequacy of supporting evidence and confidence established on justified unsuitability or discharge of counter evidence. Confidence based on the adequacy of supporting evidence can be further broken down according to two important evidential properties of argument elements – trustworthiness and appropriateness (according to the evidential properties presented within EviM in Chapter 4).

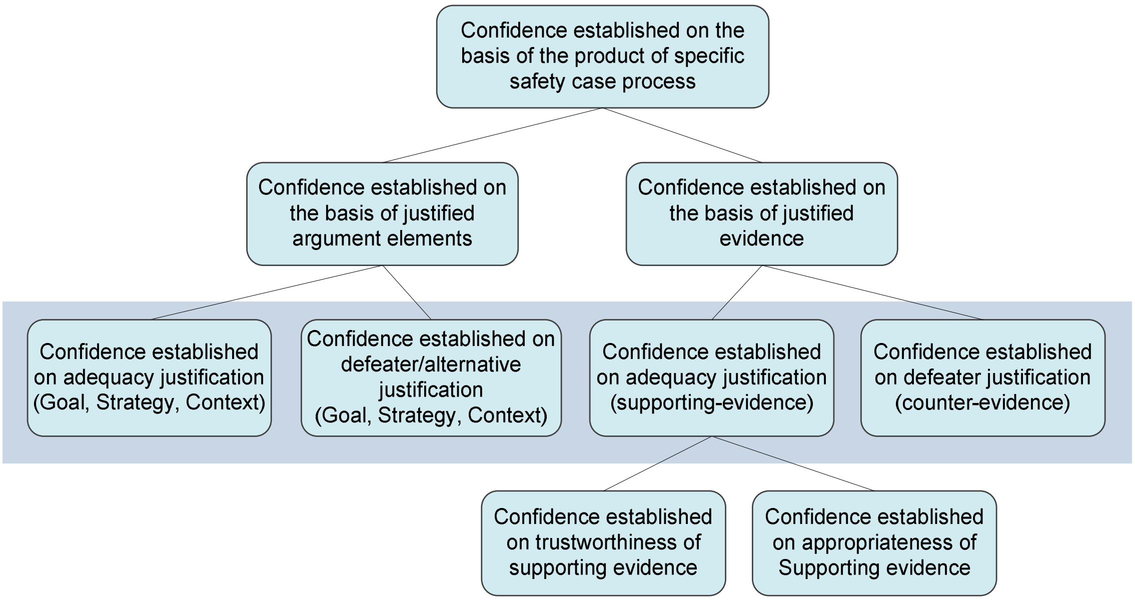

Confidence established on the basis of the product of specific safety case process
Confidence established on the basis of justified argument elements
Confidence established on the basis of justified evidence
Confidence established on adequacy justification (Goal, Strategy, Context)
Confidence established on defeater/alternative justification (Goal, Strategy, Context)
Confidence established on adequacy justification (supporting-evidence)
Confidence established on defeater justification (counter-evidence)
Confidence established on trustworthiness of supporting evidence
Confidence established on appropriateness of Supporting evidence

Figure 55 Refined Product-Branch of the Framework of Confidence in Safety Cases

In this way, we have incorporated both information contributing to confidence establishment and threatening confidence establishment into one unified structure. The overall confidence framework, which integrates Figure 55 and Figure 31, is presented in Figure 56.

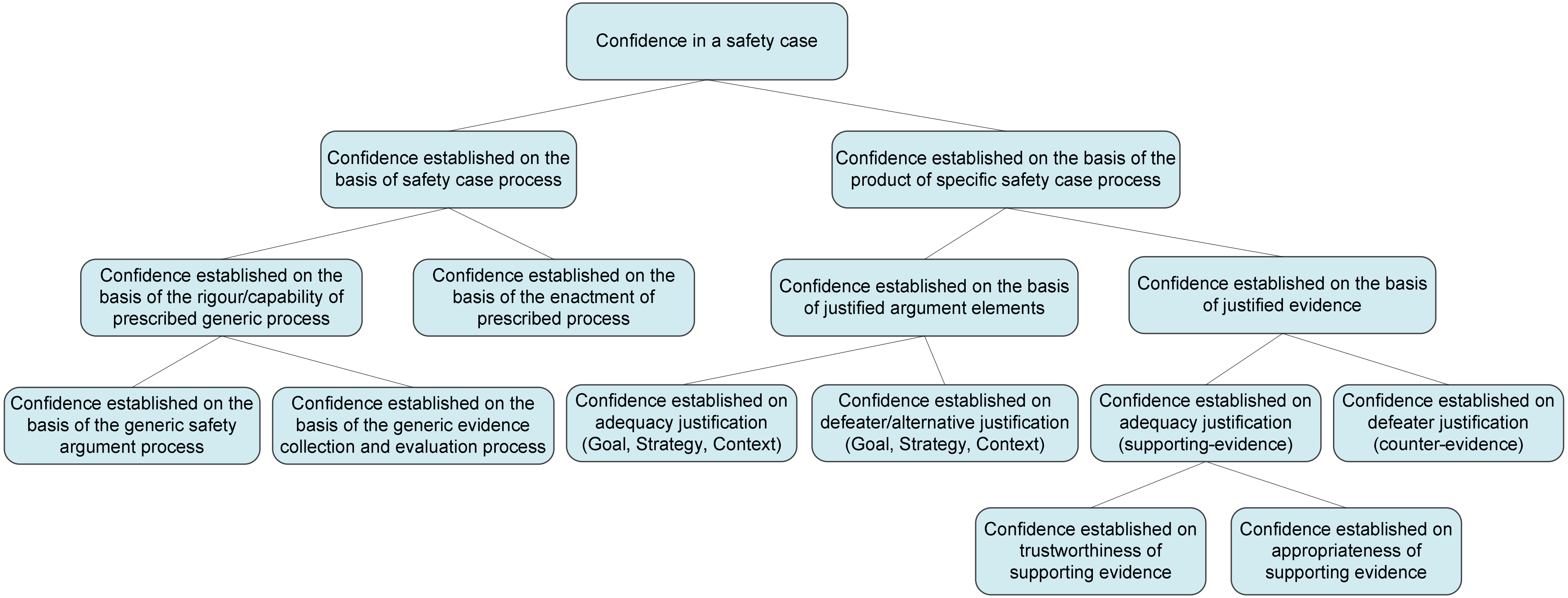

Confidence in a safety case
Confidence established on the basis of safety case process
Confidence established on the basis of the product of specific safety case process
Confidence established on the basis of the rigour/capability of prescribed generic process
Confidence established on the basis of the enactment of prescribed process
Confidence established on the basis of justified argument elements
Confidence established on the basis of justified evidence
Confidence established on the basis of the generic safety argument process
Confidence established on the basis of the generic evidence collection and evaluation process
Confidence established on adequacy justification (Goal, Strategy, Context)
Confidence established on defeater/alternative justification (Goal, Strategy, Context)
Confidence established on adequacy justification (supporting-evidence)
Confidence established on defeater justification (counter-evidence)
Confidence established on trustworthiness of supporting evidence
Confidence established on appropriateness of supporting evidence

Figure 56 Overall Framework of Confidence in Safety Cases

As discussed in Section 4.4.1, Toulmin introduced ‘backing’ as an important element in arguments. The importance of backing is prominent - “the soundness of our claims to knowledge turns on the adequacy of the arguments by which we back them” [198]. The confidence framework we propose, in nature is the foundation of backing of the strength of a primary safety argument. But the general backing function of the confidence framework should not be confused with the ‘backing’ element in Toulmin’s argument model. The confidence framework covers richer ‘ingredients’ than Toulmin’s ‘backing’. In particular, it includes three parts:

- positive backing for structured safety case elements (corresponding to Toulmin’s ‘Data’ and ‘Warrant’ elements)
- negative positive backing of safety case elements
- backing of backing (the recursive feature of backing[15])

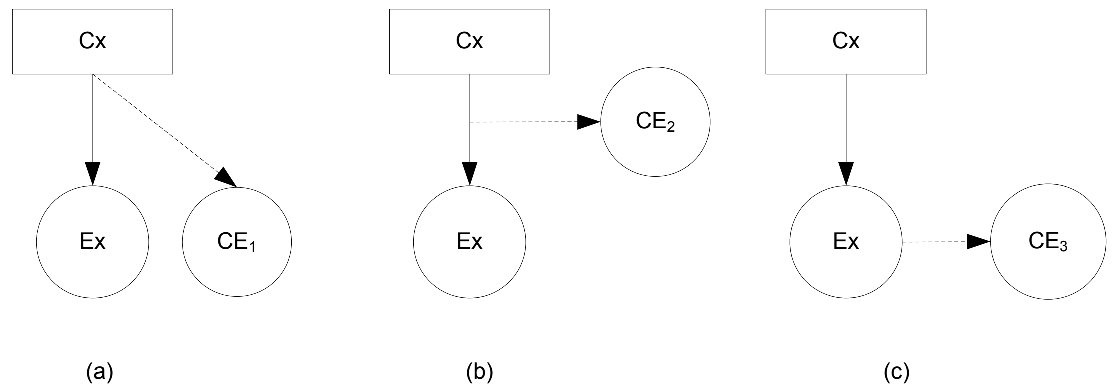
6.6.14 Relationship to Assurance Claim Points

Assurance Claim Points (ACPs) were originally introduced for referencing associated confidence argument within a graphically presented argument [97]. However, in fact, the concept of ACP does not need to be constrained within the graphical view of a safety argument. The limitation of thinking ACP within notations lies in two situations. ACPs are attached to what has been presented, and it is difficult to use them as a point of reference as to what *has not* been presented. Secondly, if all the asserted relationships for one claim need to be considered as a whole, e.g. coverage, sufficiency, where no strategy is used there is no proper place in the graphical argument for us to attach an ACP symbol.

This section clarifies and expands the ACP concept by analyzing the asserted evidence relationships and asserted counterevidence relationships associated with a safety claim. Otherwise, the negative-side of confidence arguments could not be addressed properly.

As depicted in Section 6.3, there are three ways in which counter evidence is associated with the argument-evidence interface. Figure 57 illustrates the three situations.

[15] We should be cautious about the use of backing arguments. They should not be asked without end that will stop the primary argument from progressing.



Cx – a domain safety claim

Ex - an item of supporting evidence for Cx

CE_1 - an item of counter evidence for Cx

CE_2 - an item of counter evidence for the appropriateness of Ex

CE_3 - an item of counter evidence for the trustworthiness of Ex

Figure 57 Counter Evidence for Argument-Evidence Interface

If we still use ACP to address the evidential relationships associated with C_x, the above three situations can all be addressed in one argument structure, as illustrated by Figure 58. In the figure, CE_1 is addressed at the same point as CE_3 is addressed during justification of the trustworthiness of E_x. The reason for this combination is that CE_1, actually, is a member of the set of *potential* CE_3. CE_1, as an item of counter evidence to C_x, should have had an inconsistent evidence result assertion from the assertions of E_x. CE_1, by itself, can challenge the trustworthiness of E_x. Certainly, if CE_1 could not be discharged with proper reasons, it should be kept in the primary argument as presented in Figure 57 (a).

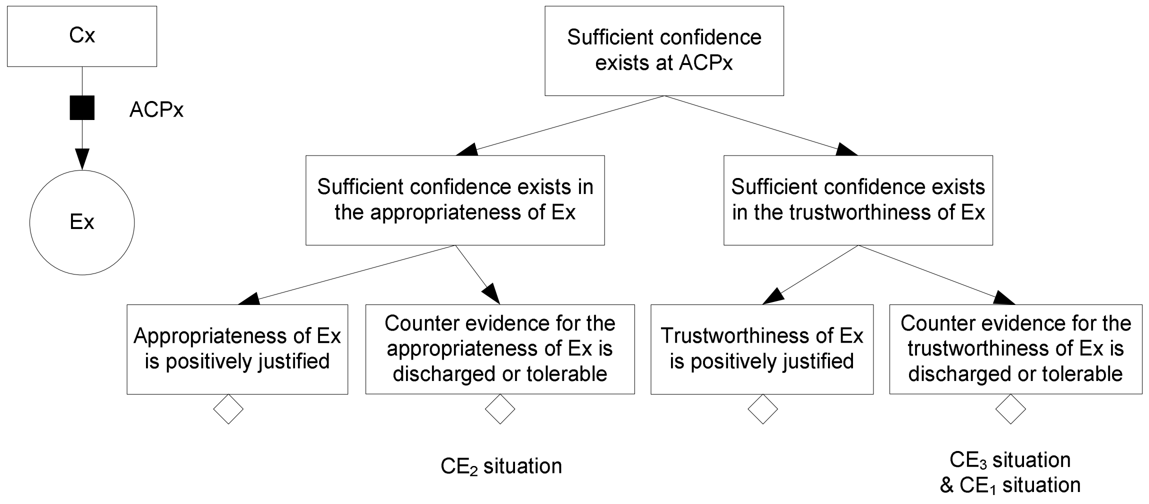

Cx
ACPx
Ex
Sufficient confidence exists at ACPx
Sufficient confidence exists in the appropriateness of Ex
Sufficient confidence exists in the trustworthiness of Ex
Appropriateness of Ex is positively justified
Counter evidence for the appropriateness of Ex is discharged or tolerable
Trustworthiness of Ex is positively justified
Counter evidence for the trustworthiness of Ex is discharged or tolerable
CE₂ situation
CE₃ situation & CE₁ situation

Figure 58 Expansion of ACP Concept

Therefore, with the expanded safety case construction process, we can still use ACP to annotate the evidence relationships to be assured, but more factors that may affect our confidence in the evidence relationships will need to be addressed than shown in [97].

6.6.15 Confirmation Bias in Arguments

Confirmation bias is "the tendency to test one's beliefs or conjectures by seeking evidence that might confirm or verify them and to ignore evidence that might disconfirm or refute them" [3]. It may exist in many situations, e.g. research studies, daily decision making, or system safety analysis.

Govier addresses potential confirmation bias in argumentation [84]. As she stated, we should not make judgements with double standards for things we agree with and things we do not agree with. It is valuable to *acknowledge* the existence of confirmation and be *aware* of its effects. More importantly, we need to find way to combat or alleviate any potential bias. Taking into account counter considerations is one of the ways to alleviate confirmation bias.

The explicit negative thinking points in the expanded process, e.g. Step E8, drive the user to search for opposing information during argument construction. The safety case generated from the process should include both reasoned support for claims and reasoned refutation for attacks (or alternative basis) to claims, which should make it more compelling and defensible. One-sided safety arguments are more likely to be vulnerable and open to attack.

It is well understood that safety cases are not intended to *prove* safety [190] but to communicate and encourage active and critical thinking [91, 114], which is also the essential requirements for performing any safety activities for the development of safety-critical systems. Thinking from the opposing perspective is one way to alleviate potential confirmation bias in safety cases.

6.7 Argument Pattern Essentials

Prior to presenting the argument patterns that have been developed to accompany the expanded process presented in Section 6.6, this section presents a brief overview of the concept and approach of using Safety Case Patterns to express generic argument structures.

6.7.1 Pattern Overview

A pattern describes a recurrent problem and the core of a reusable solution to that problem [20]. The core of a pattern is the expression of a relation between a certain context, a problem and a solution [20]. Safety Case Patterns were first introduced in [116]. They are abstract representations of the structure of a generalised safety argument associated with just one aspect of the overall argument structure. Safety case patterns, sometimes referred as argument templates [39] or generic arguments [188], provide a mechanism for capturing and reusing common arguments within safety cases. Extensions to the Goal Structuring Notation (GSN) support the representation of safety case patterns [89]. Using safety case patterns can have the following advantages [115]:

- to provide inspiration or a starting point for new safety argument developments
- to help in planning and scoping safety cases
- to help those with little safety case experience
- to help improve argument completeness
- to help speed up safety case development
- to provide a benchmark when reviewing a safety case.

However, it is worth noting that safety case patterns are not silver bullets. They are only *partial* generic solutions and they are not intended to provide a reusable model of a safety argument for a complete safety case.

A series of safety case patterns have been developed for generic construction of safety cases, such as the ALARP pattern, Diverse Argument Pattern and Safety Margin Pattern in [117]. In addition, there are also collections of interrelated safety case patterns for specific topics, such as patterns for the use of COTS (Commercial-Off-The-Shelf) components in safety applications [214], or patterns for arguing software safety in a system lifecycle [204]. Recently, safety case patterns have also been developed for Model-based development approach [27] or safety assessment justification [196] and have been practiced with more case studies [95].

6.7.2 Pattern Documentation and Generation

A typical safety case pattern is documented with the following headings [117]:

Pattern Name — *a label by which people will identify this pattern; it communicates the key principle or central argument being presented by a safety argument pattern.*

Intent — *a statement that explains what this pattern is trying to achieve.*

Motivation — *a description about why the pattern was constructed.*

Applicability (Necessary Context) — *a section to record the necessary application context of a pattern, including assumptions and principles that help avoiding misuse of the pattern.*

Structure — *a graphical representation of the structure of a pattern with clear labels using GSN pattern extensions.*

Participants — *a description of each of the elements (contextual information, strategies, goals)of the goal structure pattern presented in the '**Structure'** Section.*

Collaborations — *descriptions on how the different elements of a pattern work together to achieve the desired effect of the pattern.*

Consequences — *a declaration of work remaining after having applied or carried out an argument pattern with references to the elements of the pattern.*

Implementation — *a section that mainly communicates how to instantiate a pattern and potential traps and supports of applying a specific pattern.*

Related Patterns — *brief references of other patterns that are interrelated with a defined pattern.*

This documentation style is adopted for recording the argument patterns proposed in this thesis.

Patterns are commonly observed and extracted from good practice. It is always desirable to mine existing practice or analysis to improve and upgrade existing patterns. However, it is not possible if we don't have enough experience or the experiences are not well-documented.

Inspired by the Make/Buy/Mine/Commission Analysis [53] for helping organisations to make conscious choice on how to introduce software assets, we recognize that safety argument patterns can be developed in one of three ways: 1) they can be *mined* from existing examples of mature (reviewed and accepted) safety case practice; 2) they can be *bought* in from standards and guidance; 3) new patterns can be *made* from systematic and critical thinking and evaluation of various safety assessment processes and products (required, desired and currently practiced).

In our case, there is insufficient detailed guidance from existing standards to 'buy' a solution and, unfortunately, insufficient experience (of explicit model justification within safety cases) to ‘mine’ a pattern. Therefore, our approach has been primarily to ‘make’ patterns and iterate for improvements, following community use and feedback.

The following sections (Section6.8, Section 6.9 Section 6.10) will introduce three new argument patterns that address the structural issues during the argumentation according to the expanded process.

- Two-Sidedness Argument Pattern – this pattern shows how counter considerations can be represented; it is applicable to in both primary safety arguments and confidence arguments; it may be adopted in Step E1, E3, E4, E6, E7, or E9 of the expanded process.
- Safety Assessment Model Adequacy Argument pattern – this pattern shows how models are justified; it specifically corresponds to Step E7, for justifying the adequacy of safety assessment models as supporting evidence.
- Cross-Model Inconsistency Analysis Adequacy Argument Pattern – this pattern shows how we justify the adequacy of the results of inconsistency analysis (as described in Chapter 5) between safety assessment models (part of the embedded argumentation inside Step E7 for backing up the use of inconsistency analysis results).

6.8 Two-Sidedness Argument Pattern

The argument having counter evidence considered may appear in different ways. For example, as shown in Figure 59, Spriggs [188] presents a generic argument structure that considers the absence of counter evidence for claims. In this example, the definition of argument decomposition strategies is influenced by the fact of absence of counter evidence. On the other hand, if counter evidence is found and discharged, the associated may appear in a different shape. Figure 60 (also from [188]) is another example that depicts no adverse impact from counter evidence.

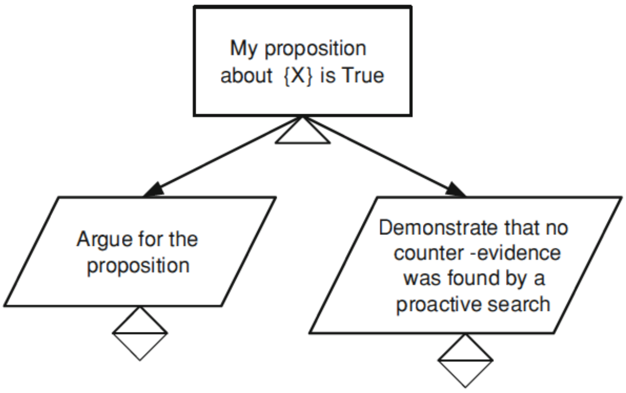

My proposition about {X} is True
Argue for the proposition
Demonstrate that no counter -evidence was found by a proactive search

Figure 59 A Generic Argument with Absence of Counter Evidence (from [188])

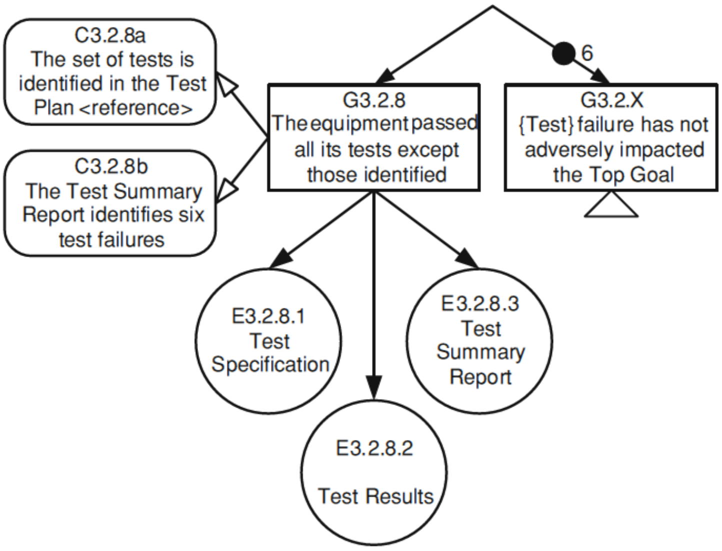
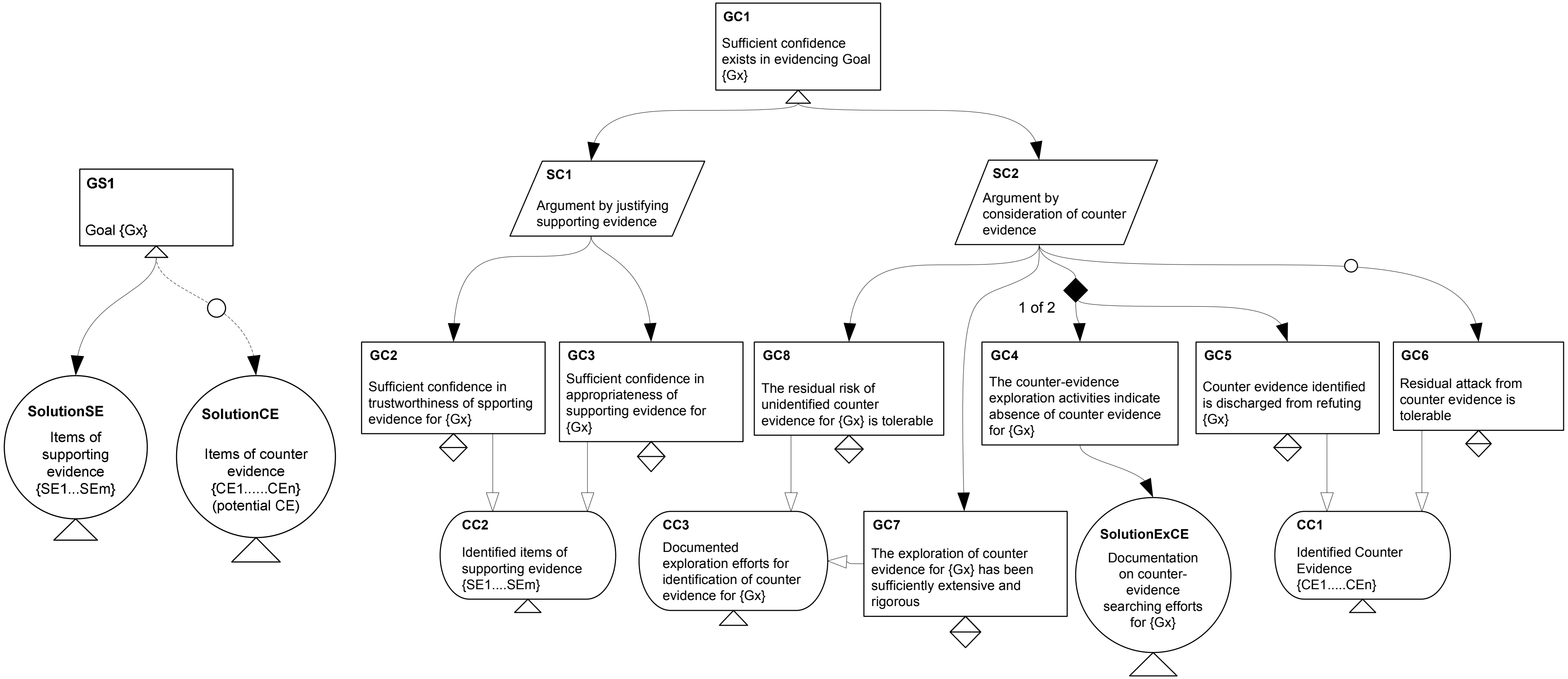

C3.2.8a
The set of tests is identified in the Test Plan <reference>
C3.2.8b
The Test Summary Report identifies six test failures
G3.2.8
The equipment passed all its tests except those identified
6
G3.2.X
{Test} failure has not adversely impacted the Top Goal
E3.2.8.1
Test Specification
E3.2.8.3
Test Summary Report
E3.2.8.2
Test Results

Figure 60 An Argument with Presence of Counter Evidence (from [188])

We generalize the consideration of counter evidence in safety case construction as a generic two-sidedness argument structure. This pattern is intended for a wider use of counter considerations in our thinking and arguing practice, and even in other safety activities. The pattern is designed for presenting supporting and opposing points in both primary safety arguments and confidence arguments. The pattern should be considered for all the expanded steps presented in Section 6.6.1.

The two-sidedness argument pattern is depicted in Figure 61. For a more detailed description, readers are suggested to refer Appendix A.1.


GS1
Goal {Gx}
SolutionSE
Items of supporting evidence {SE1...SEm}
SolutionCE
Items of counter evidence {CE1......CEn} (potential CE)
GC1
Sufficient confidence exists in evidencing Goal {Gx}
SC1
Argument by justifying supporting evidence
SC2
Argument by consideration of counter evidence
1 of 2
GC2
Sufficient confidence in trustworthiness of spporting evidence for {Gx}
GC3
Sufficient confidence in appropriateness of supporting evidence for {Gx}
GC8
The residual risk of unidentified counter evidence for {Gx} is tolerable
GC4
The counter-evidence exploration activities indicate absence of counter evidence for {Gx}
GC5
Counter evidence identified is discharged from refuting {Gx}
GC6
Residual attack from counter evidence is tolerable
CC2
Identified items of supporting evidence {SE1....SEm}
CC3
Documented exploration efforts for identification of counter evidence for {Gx}
GC7
The exploration of counter evidence for {Gx} has been sufficiently extensive and rigorous
SolutionExCE
Documentation on counter-evidence searching efforts for {Gx}
CC1
Identified Counter Evidence {CE1.....CEn}

(a)Two-sided safety argument

(b) Two-sided confidence argument

Figure 61 Two-Sidedness Argument Pattern

As Figure 61 illustrates, counter evidence {CE1…..CEn}[16], if identified for a domain safety claim, should be presented in the primary safety argument structure (a). If one or more items of information have been identified as counter evidence for {Gx}, they should be presented with the *WeakenedBy* symbol introduced in Section 6.4. While both 'support' and 'challenge' are provided for a goal, the strength of the primary argument is *uncertain* until we carefully examine these asserted relationships both as a whole and as individuals. It is valuable to cultivate a habit of considering both evidence *for* and *against* a claim.

On the other hand, the confidence in all the evidential relationships presented in Figure 61 (a) can be separately presented in Figure 61 (b). The overall confidence needs to be established on the justification of all asserted relationships – every *SupportedBy* and every *WeakenedBy* together. Each item of supporting evidence should be justified for its trustworthiness and appropriateness of providing support. If there is no counter evidence presented in a primary safety argument, the GC4 should be chosen rather than GC5 or GC6. If there are any items of counter evidence presented, they should be addressed in CC1. Furthermore, the ideal result is that GC5 – 'Counter evidence identified is discharged from refuting {Gx}' – is developed for all items of counter evidence; whereas if there were unresolved items of counter evidence, GC6 – 'Residual attack from counter evidence is tolerable' – should be developed.

In Figure 61 (b), what we want to highlight is to consider the overall adequacy of presented evidential relationships in combination. Certainly, each asserted relationship plays a role in it. But a local view of each asserted relationship individually is deficient for examining the sufficiency attribute of overall evidencing efforts and results. It is difficult to graphically represent the adequacy of a collection of asserted relationships, but it must be considered carefully in the instantiation of the pattern.

This pattern is applicable to a wider situation, rather than to a fragmented primary safety argument only. It can be adopted and reused if there is a need to take into account of counter or alternative considerations into an argument construction process.

[16] In all patterns, texts in { } are placeholders for objects to be instantiated while applying the argument patterns. For example, {CE1….CEn} means a set of counter evidence items from 1 to n; {SE1…..SEm} means a set of supporting evidence items from 1 to m; {Gx} means a desired safety goal to be justified.

6.9 Model Adequacy Argument Pattern

6.9.1 Synopsis View

Safety assessment models can play all three types of roles in safety cases - *Supporting-Evidence*, *Counter-Evidence*, and *Context*, as described in Section 6.2. In this section, we assume that a safety assessment model SAM_x is adopted as an item of *supporting* evidence in a primary safety argument. Starting with this assumption, we aim to justify that SAM_x is adequate for its usage as supporting evidence, specifically related to the Step E7 in Section 6.6.8. Nevertheless, the factors considered in the model adequacy argument pattern can also be adopted for justifying the adequacy of safety assessment models when they are in the other two roles.

Through communication with researchers, industrial safety analysts, reviewers, and certification professionals, it is acknowledged that many factors should be considered in the justification of safety assessment models. However, the factors are not always explicitly and systematically presented, considered, and documented. It is also acknowledged that, before commencing modelling, there should be pre-justification of the choice of modelling techniques. Some factors to be addressed in justification of a safety assessment model include [196]:

- The modelling technique adopted (and/or the tool adopted) is fit for the safety assessment task.
- The representation is adequate (e.g. safety requirement representation, failure mode representation, failure logic representation).
- The assumptions made during modelling are acceptable.
- The input data used is appropriate for its usage.
- Accidental modelling errors are sufficiently identified and eliminated.
- The modeller is competent to do the modelling.
- There is sufficient understanding of the entity being modelled.
- The modelling tool is properly configured for the model.

The model adequacy argument pattern proposed in this section integrates these factors in a structured way in support of the implementation of Step E7 in the expanded argument construction process.

A typical use of safety assessment models as evidence in a primary safety argument is presented in Figure 62. The evidence result assertion of SAM$_x$ is omitted in this figure. But as described in Figure 29 in Section 4.5.5, the connection between the source data of an item of evidence with an argument lies in evidence result assertions, the propositional content of the evidence item. The solution node in the graphical view of an argument provides only the reference to the original artefact of SAM$_x$.

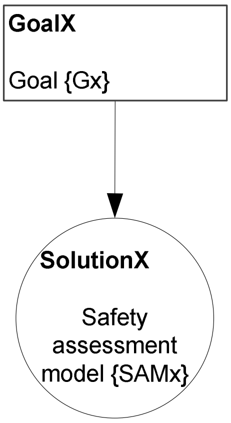


Figure 62 Use of Model SAM$_x$ as Supporting-Evidence

As engineering experience indicates, model adequacy is influenced and exhibited by a variety of factors. To depict these factors in a clear and organized way, we have divided the overall pattern into three parts. Figure 63 provides us a synopsis view of the context and relationships of the different parts of the argument pattern, which are associated with the adequacy of safety assessment models.

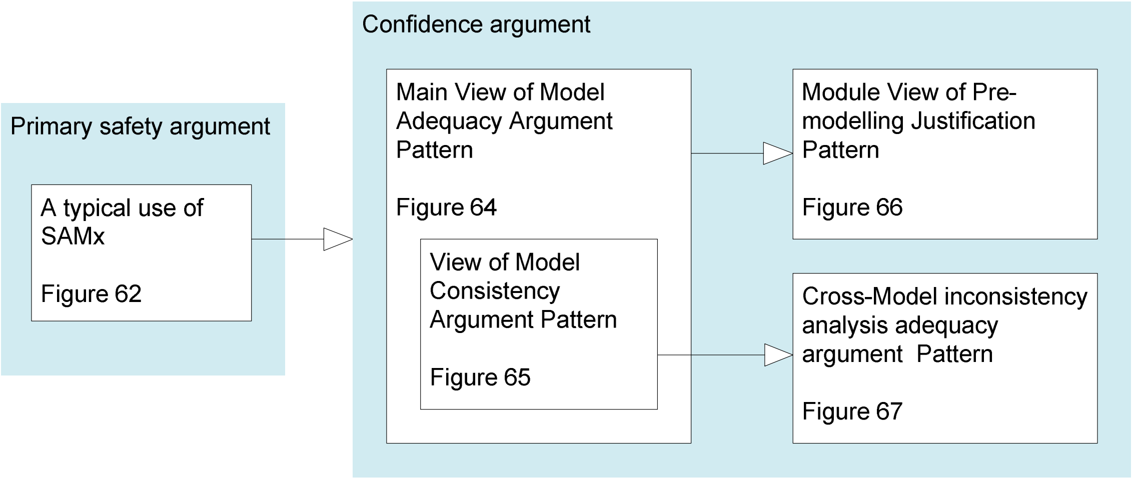

Confidence argument
Primary safety argument
A typical use of SAMx
Figure 62
Main View of Model Adequacy Argument Pattern
Figure 64
View of Model Consistency Argument Pattern
Figure 65
Module View of Pre-modelling Justification Pattern
Figure 66
Cross-Model inconsistency analysis adequacy argument Pattern
Figure 67

Figure 63 Synopsis View of Model Adequacy Argument Pattern

6.9.2 Main View - Post Modelling Justification

The justification and use of a safety model is achieved in two stages – the pre-modelling justification of the methodology selected (and tool adopted if there is one) and the post-modelling justification of the adequate use of the model as evidence.

Figure 64 illustrates the main view of the model adequacy argument pattern. It is developed as a generic confidence argument linked with the exemplar primary safety argument presented in Figure 62. Appendix A.2 Part (a) presents more detailed description of the Post-Modelling Justification of Model Adequacy Argument Pattern.

The post-modelling justification blocks shown in Figure 64 are concerned with justification from the perspective of both the enactment (i.e. the specific execution instance of the modelling process) and the outcome of this modelling process (a specific safety assessment model that has been produced for a target system with the selected method and tool). Four different types of model-relevant information should be considered – the model building blocks or modelling construction elements, the assumed context of safety assessment models, the substantial results generated from safety assessment models, and the absence of unjustified inconsistency between comparable models. The lower level blocks in Figure 64 are not orthogonal. The justification of model building blocks can be used to support the justification of safety modelling substance results. The block of inconsistency justification can be used to support the other three depending on the nature of the consistency relationships applied. Furthermore, although we do not describe justification *during* modelling, the in-line annotation of the modelling decisions and rationale during analysis is essential for the post-modelling justification.

There are two parts of the adequacy argument pattern that are simplified in the main view:

- the pre-modelling justification as a modular context block (it is presented as a separate argument module in Figure 66);
- the decomposition of GM11 in - a sub-goal that is related to the model consistency of {SAM_x} (it is refined with further details in a separate view in Figure 65).

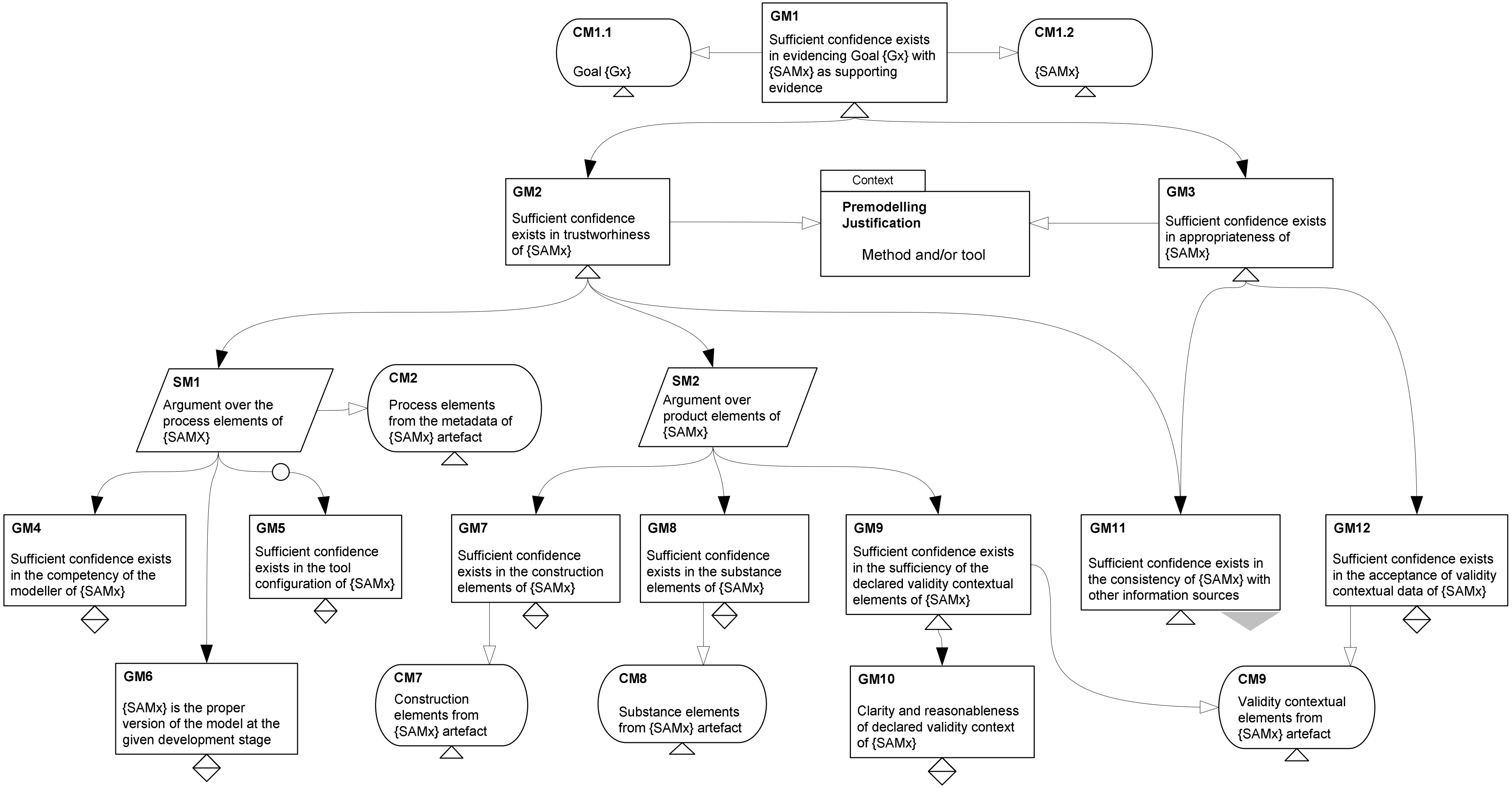

CM1.1
Goal {Gx}
GM1
Sufficient confidence exists in evidencing Goal {Gx} with {SAMx} as supporting evidence
CM1.2
{SAMx}
GM2
Sufficient confidence exists in trustworhiness of {SAMx}
Context
Premodelling Justification
Method and/or tool
GM3
Sufficient confidence exists in appropriateness of {SAMx}
SM1
Argument over the process elements of {SAMX}
CM2
Process elements from the metadata of {SAMx} artefact
SM2
Argument over product elements of {SAMx}
GM4
Sufficient confidence exists in the competency of the modeller of {SAMx}
GM5
Sufficient confidence exists in the tool configuration of {SAMx}
GM7
Sufficient confidence exists in the construction elements of {SAMx}
GM8
Sufficient confidence exists in the substance elements of {SAMx}
GM9
Sufficient confidence exists in the sufficiency of the declared validity contextual elements of {SAMx}
GM11
Sufficient confidence exists in the consistency of {SAMx} with other information sources
GM12
Sufficient confidence exists in the acceptance of validity contextual data of {SAMx}
GM6
{SAMx} is the proper version of the model at the given development stage
CM7
Construction elements from {SAMx} artefact
CM8
Substance elements from {SAMx} artefact
GM10
Clarity and reasonableness of declared validity context of {SAMx}
CM9
Validity contextual elements from {SAMx} artefact

Figure 64 Main View - Model Adequacy Argument Pattern

6.9.3 Branch View - Model Consistency Justification

Figure 65 illustrates the generic argument structure of model adequacy justification based on the model consistency claimed, which is a refined part of the goal G11 in Figure 64. For a detailed description of the Model Consistency Justification of Model Adequacy Argument Pattern, readers should refer to Appendix A.2 Part (b). The goal G11 in the main view of the adequacy argument pattern is decomposed according to the typology of model consistency defined in Section 5.2.2. The cross-model inconsistency analysis suggested in Chapter 5 is addressed as a generic solution node in this part of the adequacy argument pattern.

Similarly, since we present a reference to cross-model inconsistency analysis results by Solution 11.S1, we may need to provide further backing for the adequacy of inconsistency analysis. This is the topic of Section 6.10.

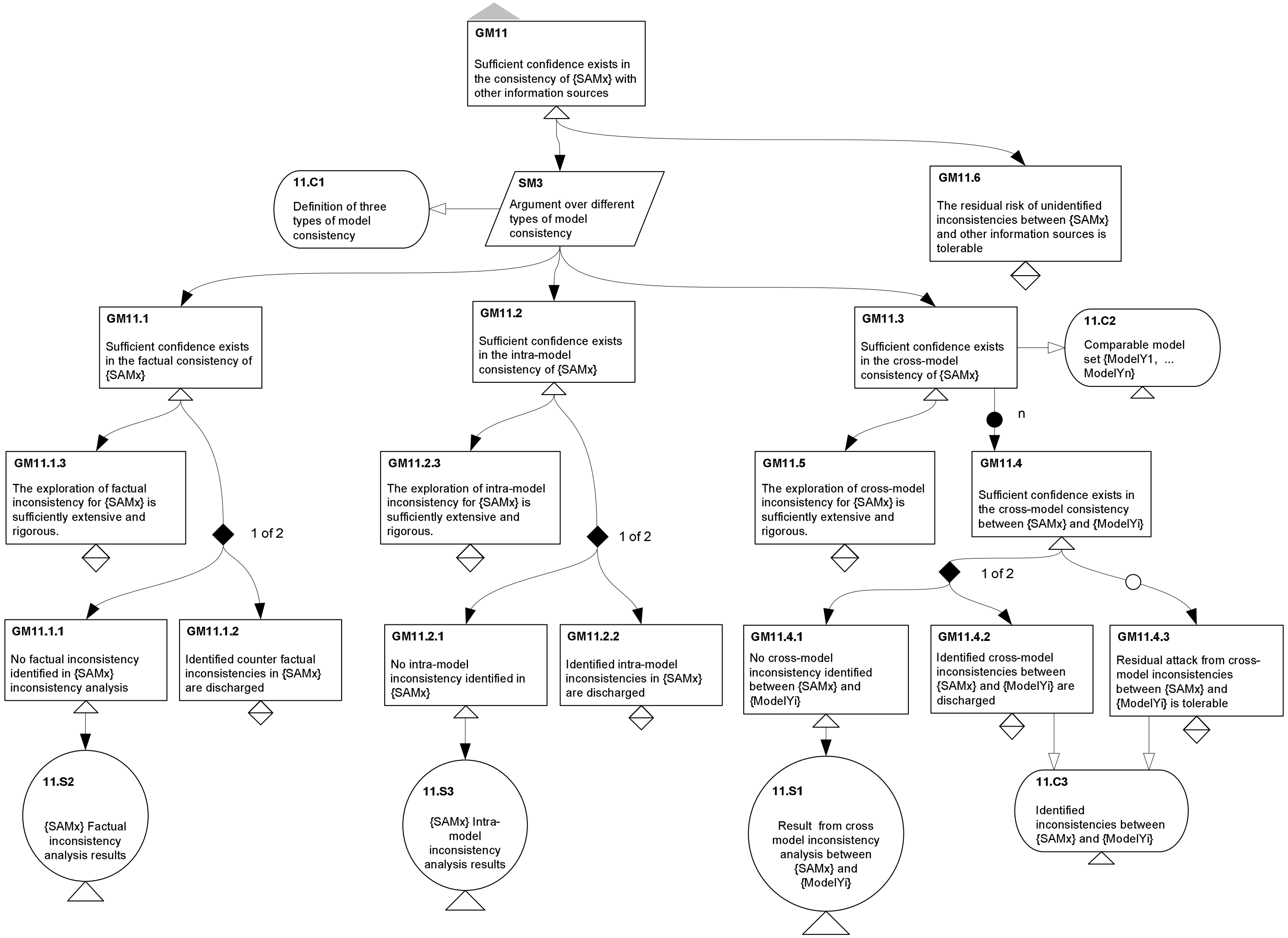

GM11
Sufficient confidence exists in the consistency of {SAMx} with other information sources
11.C1
Definition of three types of model consistency
SM3
Argument over different types of model consistency
GM11.6
The residual risk of unidentified inconsistencies between {SAMx} and other information sources is tolerable
GM11.1
Sufficient confidence exists in the factual consistency of {SAMx}
GM11.2
Sufficient confidence exists in the intra-model consistency of {SAMx}
GM11.3
Sufficient confidence exists in the cross-model consistency of {SAMx}
11.C2
Comparable model set {ModelY1, ... ModelYn}
n
GM11.1.3
The exploration of factual inconsistency for {SAMx} is sufficiently extensive and rigorous.
1 of 2
GM11.2.3
The exploration of intra-model inconsistency for {SAMx} is sufficiently extensive and rigorous.
1 of 2
GM11.5
The exploration of cross-model inconsistency for {SAMx} is sufficiently extensive and rigorous.
GM11.4
Sufficient confidence exists in the cross-model consistency between {SAMx} and {ModelYi}
1 of 2
GM11.1.1
No factual inconsistency identified in {SAMx} inconsistency analysis
GM11.1.2
Identified counter factual inconsistencies in {SAMx} are discharged
GM11.2.1
No intra-model inconsistency identified in {SAMx}
GM11.2.2
Identified intra-model inconsistencies in {SAMx} are discharged
GM11.4.1
No cross-model inconsistency identified between {SAMx} and {ModelYi}
GM11.4.2
Identified cross-model inconsistencies between {SAMx} and {ModelYi} are discharged
GM11.4.3
Residual attack from cross-model inconsistencies between {SAMx} and {ModelYi} is tolerable
11.S2
{SAMx} Factual inconsistency analysis results
11.S3
{SAMx} Intra-model inconsistency analysis results
11.S1
Result from cross model inconsistency analysis between {SAMx} and {ModelYi}
11.C3
Identified inconsistencies between {SAMx} and {ModelYi}

Figure 65 Branch View - Model Consistency Argument Pattern

6.9.4 Contextual Module of Pre-Modelling Justification

Figure 66 illustrates the generic argument structure of pre-modelling justification. The pre-modelling justification needs to consider the capability and feasibility of the selected technological elements in modelling – commonly the modelling method and the modelling tool. For example, it is necessary to consider if the expressive power of the modelling methodology is sufficient for the modelling intent (e.g. the important aspects of components behaviour can be represented; the important classes of inter-component interactions and dependencies can be represented). It is also necessary to check whether the analytic capability of the modelling methodology is sufficient for the modelling intent (e.g. resolution level of the calculation of failure rates, inference of failure effects), and whether the underlying assumptions of the methodology are acceptable for the system being modelled in terms of its modelling purpose. The resources needed by the modelling methodology need to be considered (e.g. assessing as to whether required data is available; or the tool support available is appropriate for the modelling requirement; or whether there exist sufficient modellers with the required knowledge and experience for the adoption of the methodology). Documenting the reasoning process behind the selection of modelling methodology (and or tool) can provide useful backing to the adequacy of safety assessment evidence adopted in a primary safety argument.

Appendix A.2 Part (c) presents more detailed description of the Pre-Modelling Justification Module of Model Adequacy Argument Pattern.

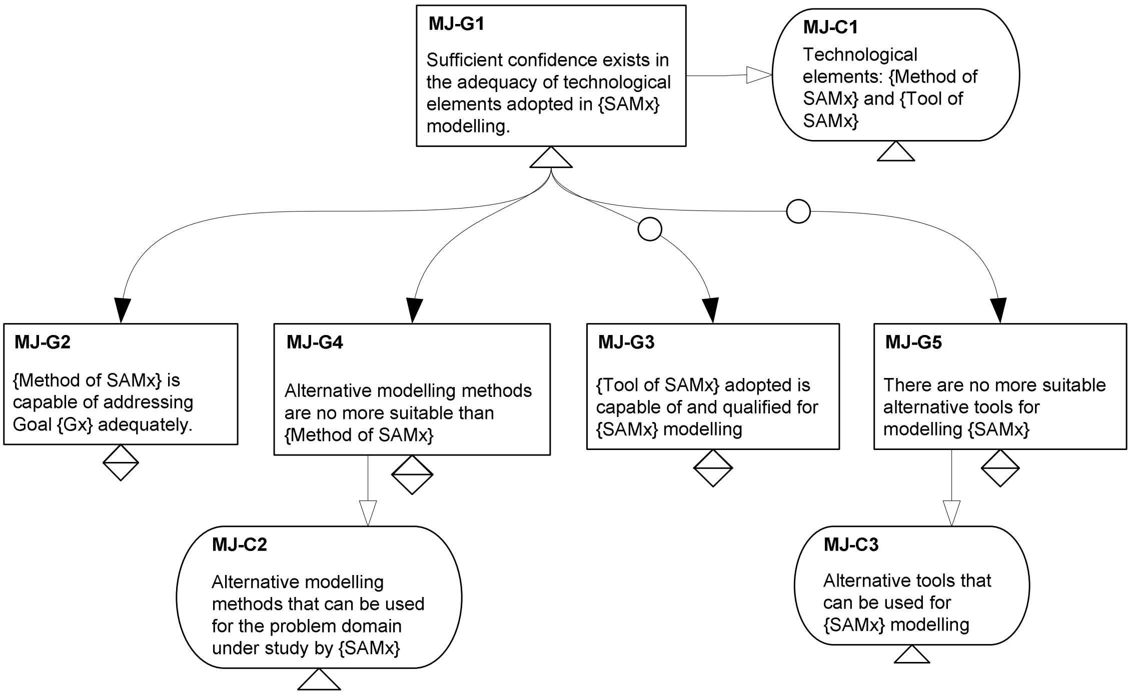

MJ-G1
Sufficient confidence exists in the adequacy of technological elements adopted in {SAMx} modelling.
MJ-C1
Technological elements: {Method of SAMx} and {Tool of SAMx}
MJ-G2
{Method of SAMx} is capable of addressing Goal {Gx} adequately.
MJ-G4
Alternative modelling methods are no more suitable than {Method of SAMx}
MJ-G3
{Tool of SAMx} adopted is capable of and qualified for {SAMx} modelling
MJ-G5
There are no more suitable alternative tools for modelling {SAMx}
MJ-C2
Alternative modelling methods that can be used for the problem domain under study by {SAMx}
MJ-C3
Alternative tools that can be used for {SAMx} modelling

Figure 66 Argument Pattern of Pre-Modelling Justification

6.9.5 Pattern Features

Some significant features of the model adequacy argument pattern (as presented in Section 6.9.1~ Section 6.9.4) include:

- It is in line with the confidence framework presented in Section 6.6.12.
    - It is developed particularly for confidence argument construction;
    - It makes clear the distinction between confidence arguments and a primary safety argument;
    - It addresses justification from both the modelling process perspective and the modelling product perspective.
- Counter consideration has been accounted for and included in the pattern, e.g. the decomposition of different types of model consistency in Figure 65.
- The source data of key contexts is in line with the safety assessment CoreDMM (as depicted in Chapter 3), e.g. CM2, CM7, CM8, CM9 can be obtained from SAM artefacts, as required by CoreDMM.
- The dual effects of inconsistency analysis are shown in Figure 65. It has potential impact on both the trustworthiness of evidence information and the appropriateness of the

evidence relation. Whether both of these aspects (GM2 or GM3) are addressed depends on the specific consistency relationships defined and the inconsistencies identified. Each individual inconsistency identified should be considered comprehensively for both GM2 and GM3.

- It demonstrates how the results of the inconsistency analysis described in Chapter 5 can and should be integrated into a confidence argument, e.g. the Solution 11-S1 in Figure 65.

6.9.6 Role of Evidence Assertion

The relationship of evidence assertions with the model adequacy argument pattern is depicted in this section. Even though no generic evidence assertions are presented in the model adequacy argument pattern, the pattern does indicate the importance of eliciting evidence assertions in order to provide assistance to using evidence properly and understanding the adequacy of evidence items more easily.

From discussion with domain experts, we acknowledge that there are various types of model review or 'validation' activities that are planned and performed on different types of models with varied scales in real practice in a system lifecycle. It seems that the top goal GM1 in the model adequacy argument pattern can be decomposed according to these model evaluation activities, which may bring about a much simplified generic argument structure. However, thinking of the diverse model evaluation scenarios in reality, it is found that the generic sub-goal decomposition based on model evaluation activities is of little use for clarifying the specific points that should be consider in the evaluation of safety assessment models, but only a requirement as a repetitive summarization of existing evaluation work.

For example, if a human expert review report of SAM_x (in short, ReportX) was used for supporting the confidence in the adequacy of SAM_x directly, we could not see through the solution referenced in an argument of why the review report is an adequate item of supporting evidence. We may ask questions, such as:

- What has been reviewed as recorded in ReportX?

- The competency of the modeller, or each construction elements of SAM_x?

- Does the conclusion recorded in ReportX totally agree with and support the goal of 'sufficient confidence in the adequacy of SAM_x'?

- Is the review rigorous enough? Does it cover multiple efforts in analyzing and examining SAM_x in different ways and from different perspective?

If evidence assertions are unstated and un-clarified, the evidential power of ReportX cannot be fully displayed. In the worse case, a weak and insufficient review may provide us an unaware partial view or illusion that we have sufficient confidence. However, the real case may be that the review covers only the competency of modellers and makes our confidence fully based on this belief. That is far from what is needed for the justification of complicated safety assessment models. To avoid such potential hazards to our confidence argument, explicit evidence result assertions can enable us link an item of evidence item, e.g. ReportX, with one or more sub-goals in the model adequacy argument pattern, which is significantly helpful for clarifying what kind of confidence sources have been considered and whether they have been addressed and demonstrated by an item of evidence.

In addition, many of the sub-goals in the model adequacy argument pattern may need evidence descriptive assertions as supporting evidence. For example, to address the goal concerning modeller competency, descriptive assertions regarding the modeller and modelling technique can be used as support. It is also impossible to be clear and explicit without evidence descriptive assertions, especially when an item of evidence is rich in content and/or complicated in nature.

As shown in Section 4.6.1, Section 6.6.12 and the model adequacy argument pattern, confidence in safety cases originates from a variety of sources. Unless we have had a good understanding of the factors influencing our confidence in the use of a model, the confidence in the model results cannot be systematically established. Linking a top or higher-level goal in an argument directly with the reference to an item of evidence, cannot provide us clear view and understanding of the argument, if in absence of the help from evidence assertions. Obviously, reliance on digging out evidence assertions buried in the original evidence artefacts until safety case review stage is undesirable. Evidence assertions should be elicited during argument construction or even before the argument construction if the typical usage of items of evidence is known from experience.

6.9.7 Undeveloped Goals

There are still many undeveloped goals in the model adequacy argument pattern. This section provides instructions for some of them, and is intended to shed some light on how to address these goals in further supporting arguments.

GM4: a goal regarding the competency of modellers. This is a goal that worth more emphasis and weight than other process-related goals, e.g. GM5, GM6. In context of model justification, the knowledge and experience of the modeller about the system in question and the modelling technique are key factors to be evaluated for this goal. There are also good reference materials that may assist the development of this goal. For example, a generic argument structure is presented in [188], which depicts how '{Staff} has pertinent Knowledge' can be further decomposed into six sub-goals. There is a systematic 'competence scheme' defined in [104], which helps the break-down of personnel competency into sub-aspects.

GM7: a goal addressing many details of construction elements in a model. The development can be based on the meta-model of a specific modelling technique. The various decisions made during a modelling process should be considered carefully. An example of partial detailed development of this goal is provided in Appendix C.5.

GM8: a goal addressing details of substance elements of a model. It may be supported by the evaluation and analysis of the reasonableness of substantial modelling outputs that to be used in evidence result assertions. For example, the MCSs of a fault tree model should be evaluated for their reasonableness, as suggested by [137, 150].

GM10: a goal addressing contextual model elements that are often neglected. The main contextual data, such as assumptions of a model, limitations of a model, data sources of a model, are defined in CoreDMM in Chapter 3. The validity context of a model should be as explicit as possible. We need to examine the sufficiency of declared context to ensure that we have sufficiently clear understanding of the validity envelope of a model.

GM12: a goal addressing that the declared validity context remains acceptable in the argumentation context. That is to say, for example, the assumptions made by a model are still considered acceptable when the model is used to support a domain safety goal. It is necessary to check that the context of the model is compatible with the context assumed of the domain claim being supported by that model.

6.10 Cross-Model Inconsistency Analysis Adequacy Pattern

In this section, the justification of the adequacy of cross-model inconsistency analysis is presented in a structured way. As illustrated in Figure 67, four aspects need to be addressed in the justification process.

- The coverage and reasonableness of defined consistency relationships
- The correctness of user-defined correspondences between models elements (if they exist)
- The correctness and implementation of algorithms associated with defined consistency relationships
- The reasonableness of the explanation of identified violations of consistency relationships.

This pattern can be used in the Phase 6 of the cross-model inconsistency analysis method (as depicted in Section 5.4.7). It can also be adopted in presenting 'backing of backing' in Step E7 (as depicted in Section 6.6.8) for justifying the adequacy of evidence for the adequacy of an item evidence for a domain claim.

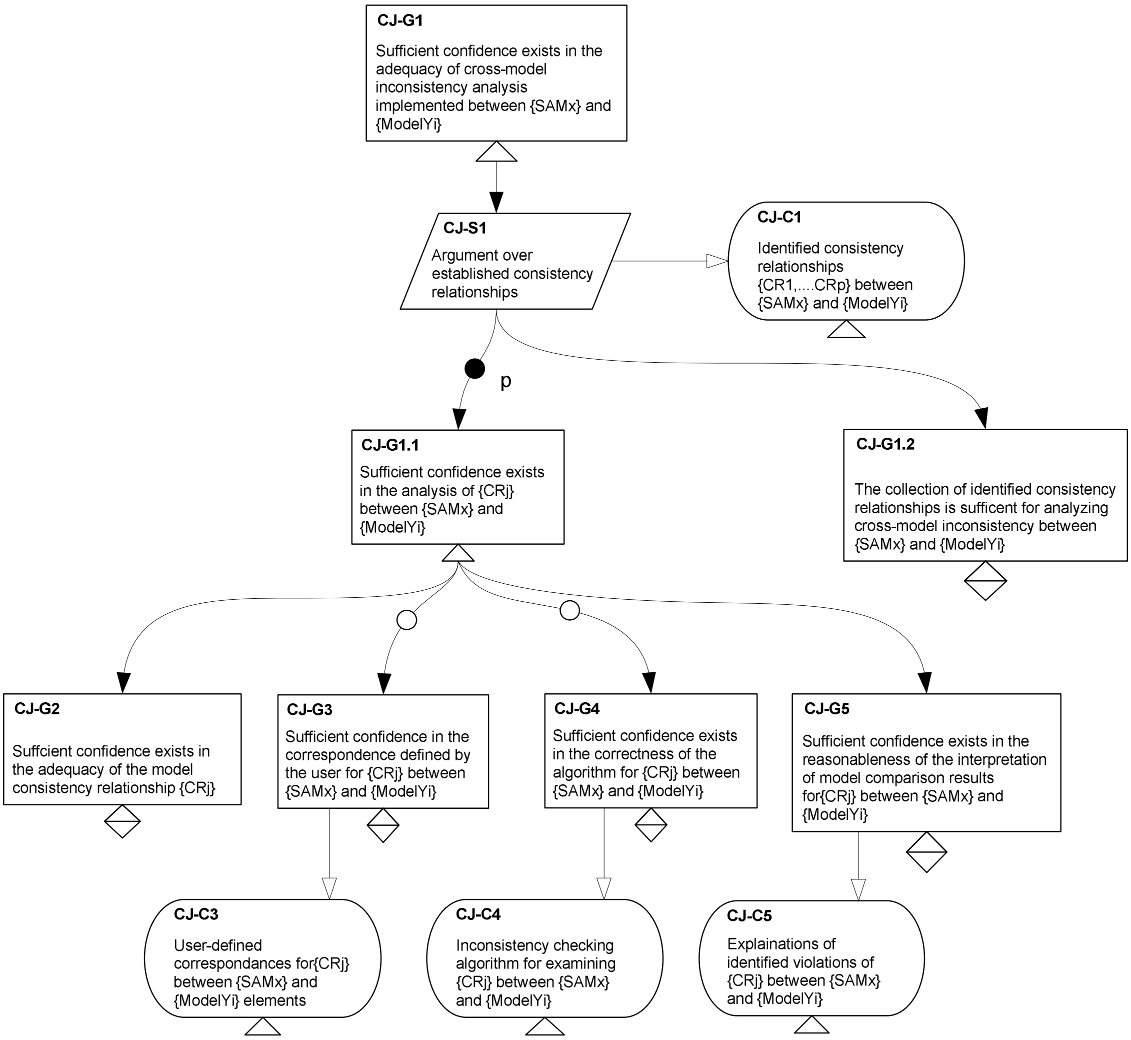

CJ-G1
Sufficient confidence exists in the adequacy of cross-model inconsistency analysis implemented between {SAMx} and {ModelYi}
CJ-S1
Argument over established consistency relationships
CJ-C1
Identified consistency relationships {CR1,....CRp} between {SAMx} and {ModelYi}
p
CJ-G1.1
Sufficient confidence exists in the analysis of {CRj} between {SAMx} and {ModelYi}
CJ-G1.2
The collection of identified consistency relationships is sufficent for analyzing cross-model inconsistency between {SAMx} and {ModelYi}
CJ-G2
Suffcient confidence exists in the adequacy of the model consistency relationship {CRj}
CJ-G3
Sufficient confidence in the correspondence defined by the user for {CRj} between {SAMx} and {ModelYi}
CJ-G4
Sufficient confidence exists in the correctness of the algorithm for {CRj} between {SAMx} and {ModelYi}
CJ-G5
Sufficient confidence exists in the reasonableness of the interpretation of model comparison results for{CRj} between {SAMx} and {ModelYi}
CJ-C3
User-defined correspondances for{CRj} between {SAMx} and {ModelYi} elements
CJ-C4
Inconsistency checking algorithm for examining {CRj} between {SAMx} and {ModelYi}
CJ-C5
Explainations of identified violations of {CRj} between {SAMx} and {ModelYi}

Figure 67 Cross-Model Inconsistency Analysis Adequacy Argument Pattern

The context data needed in this pattern, e.g. CJ-C2, CJ-C3, CJ-C4, CJ-C5, is in line with the data recorded on the basis of the information model of cross-model inconsistency analysis presented in Chapter 5.

6.11 Special Concerns

The concerns and the patterns presented are intended to be inspirational and informative. According to the study in [196], we also need to be careful with the formulation and adoption of safety case patterns:

- Patterns should not be inappropriately selected and dogmatically applied.
- There is no exemption from critical thinking and thinking about the 'lower-level' details of safety arguments.
- Rigorous safety case review should not be overlooked because of the adoption of patterns.

Nevertheless, we believe that the high level structure of the justification of the validity of safety assessment models presented in previous sections will help shape a holistic view of factors influencing model validity. It can also influence the evidence acquisition requirement from the beginning of modelling, the data management and collection in the modelling process, and the subsequent evaluation of modelling artefacts in terms of their fitness for purposes.

6.12 Summary

In this chapter, we present an expanded safety case construction process and three safety case patterns. Through the expanded process, we are able to deal with positive argument and negative positive argument separately with distinctive steps; meanwhile, they are integrated/ organised within one coherent process, allowing clear elaboration of the relationships between elements in a primary safety argument and elements in associated confidence arguments. The three argument patterns have addressed comprehensively the confidence issues associated safety assessment models in an argumentation context. Importantly, the model adequacy argument pattern has used the outputs of previous chapters are as a basis of argument structuring: data elements within CoreDMM in Chapter 3 as *Context* and *Solution* in safety argument and confidence argument; data elements (evidential properties) within EviM in Chapter 4 for decomposition of goals in confidence argument; data elements from cross-model inconsistency analysis in Chapter 5 in confidence argument. In addition, the application of *EvidenceAssertion* of EviM has been demonstrated in the case study of model adequacy argument construction in Appendix C.5.

7 Evaluation

7.1 Introduction

In Chapter 1, the following thesis proposition was stated:

> *The use of a **structured** approach to the integration and justification of safety assessment evidence within safety cases facilitates the identification and potential resolution of issues which may otherwise reduce confidence in safety justification practice.*

The proposition is supported in the subsequent chapters through:

- **Structured information** based on clarified concepts of safety assessment evidence and model consistency
    - The development of a safety assessment core data meta-model, a model of evidence that interfaces evidence items and safety arguments, and an information model of cross-model inconsistency analysis
- **Structured processes** driven by active thinking and rigorous exploration
    - The elaboration of a cross-model inconsistency analysis method
    - The formulation of an expanded argument construction process
- **Structured guidance** synthesized from dialectical argumentation and model evaluation practice
    - The development of argument patterns that presents reusable argument structures

Within the time-span of the doctoral program, the evaluation of the thesis proposition has considered two main concerns, namely, the *efficacy* of the overall approach and the *practicality* of the approach. The efficacy part intends to demonstrate the declared capability of the approach to achieve the intended better integration and justification of safety assessment models; the practicality part aims to identify potential difficulties engaged in the application of the approach for establishing increased confidence in safety cases.

Section 7.2 describes the means of evaluation that have been applied in this thesis. Section 7.3 explains how each research output is evaluated. Section 7.4 discusses further evaluation to be conducted.

7.2 Means of Evaluation

Different means of evaluation were employed during different phases of this doctoral study, with consideration of the features of the research outputs being assessed and the available resources. The means of evaluation adopted in the thesis include the following:

- Simple running examples
- Comparison with existing work
- Formalisation with tool support
- Case studies
- Peer review

The following subsections describe each of these forms of evaluation. Section7.3 presents the results of these forms of evaluation.

7.2.1 Running Examples

This is a simple form of evaluation at an early stage of a study that provides inline explanation and illustration of the ideas, concepts, models, or methods newly introduced in the thesis to facilitate understanding and to quickly demonstrate how they are used in typical application contexts. This is a weak form of evaluation, but it is useful for providing initial thoughts on the efficacy of a new concept or process. Running examples are presented in the thesis as much as possible in support of quick understanding and succinct illustration of ideas.

7.2.2 Comparison with Existing Work

Comparison with existing research and practice provides some context to judge the progress or improvements achieved through the research. It is a fundamental research step to acknowledging the existing work in academia and in industry that is relevant to the problem domain under study and identifying their strengths, disadvantages and deficiencies. This form of evaluation is adopted for demonstrating the reasonableness of conceptual outputs and the benefits of methodical outputs. It can also explicate the intended application areas of research outputs and how they fit into the current domain practice.

7.2.3 Formalisation with Tool Support

The evaluation of conceptual outputs of research work, such as meta-models that capture concepts or information elements and relationships between them, is difficult. The formalisation of meta-models in a tool environment is one initial step of examining the coverage, the expressive-sufficiency and the self-consistency of the model definitions. Widely-used model editors, such as Eclipse Modelling Framework (EMF) [189], can support the examination of the expression sufficiency of defined meta-models in context of a meta-meta-modelling language (e.g. Ecore [189]) and provides tool support for future attempts of more complicated model manipulation and potential mechanisation of some model analysis steps. EMF has been adopted for the three information models presented in the thesis.

7.2.4 Case Studies

A case study is an application of the defined approach using more detailed examples with a relatively real context. Compared to the simple running examples, case studies are a stronger means of evaluation for investigating the efficacy because of the real and concrete application context employed. Besides that, more issues potentially related to the practicality of a proposed approach may be identified during more extensive exercises in case studies with an application setting. However, it is not always feasible to obtain a significant sample of case studies for each kind of research outputs. In the thesis, a case study is adopted for the evaluation of structured processes (cross-model inconsistency analysis and expanded argument construction) and structured argument patterns.

A Braking System Control Unit (BSCU) of an aircraft Wheel Braking System (WBS) taken from ARP 4761 [181] is adopted as the example system under study. The system architecture is depicted in Figure 68. The BSCU is a computer used for handling braking commands and producing control signals to the brakes at wheels in order to decelerate the aircraft on the ground.

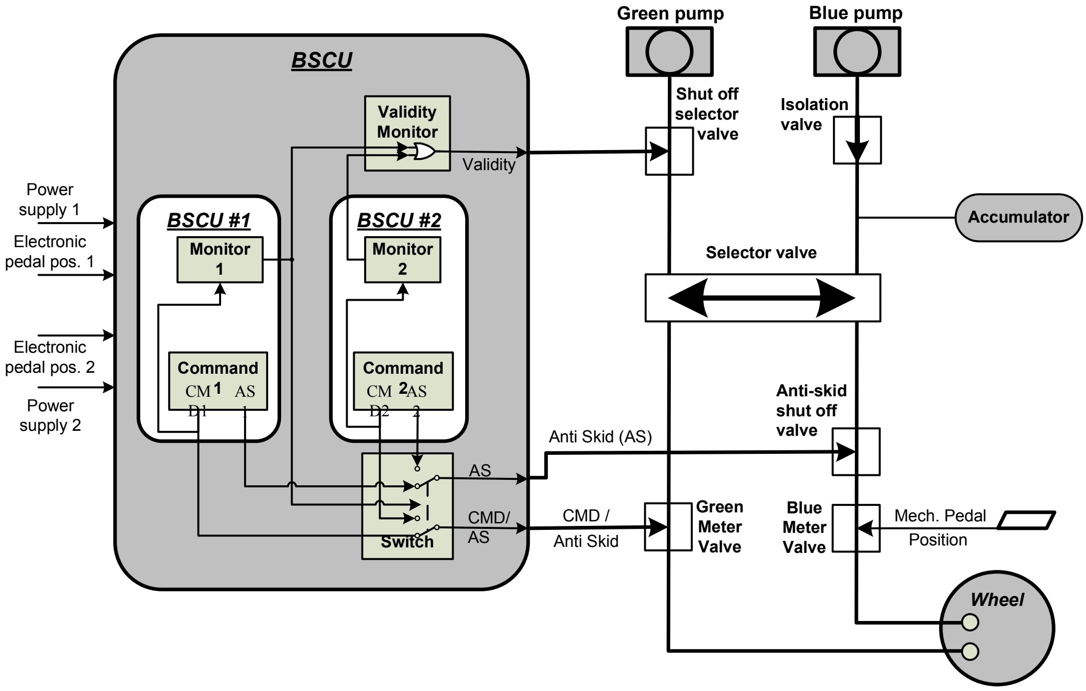


Figure 68 Example System Architecture

We have two safety assessment models associated with this system at the PSSA stage (as described in the safety assessment process shown in Section 2.5.1).

Model I: Wheel Brake System taken from MISSA FLM Handbook [131]

Coded as **OCAS.4761.WBS**.

Model II: Brake System Control Unit FTA taken from ARP 4761Appendix L [181]

Coded as **FT.4761.IB**.

Model I is a failure logic model expressed in AltaRica that is developed on the basis of the prototype wheel braking system as described in [181]. Comprehensive model descriptions of the ARP Wheel Braking System Failure Logic Modelling example are presented in [132]. Model II is a manually-constructed fault tree model taken from SAE ARP 4761 Appendix L [181].

In the case study, we apply the inconsistency analysis method defined in Chapter 5 to investigate the inconsistency between the two exemplar safety assessment models. We also evaluate the expanded argument construction process and the proposed argument patterns through the development of example arguments on the basis of the example models and the inconsistency analysis performed.

7.2.5 Peer Review

Peer review is a common means of evaluation of research work. The exposition of research output to experts and practitioners in the safety domain can provide useful feedback to the efficacy and practicality of an approach. Some of the research outputs in the thesis have been communicated with and reviewed by the research and engineering community in the following ways:

- Presentation of the research results at regular project meetings for the EC funded MISSA (More Integrated Systems Safety Assessment) project and feedback on work presented in project deliverables.

- MISSA dissemination workshops to industry with presentation of materials by the author.

- Peer-reviewed papers published at international conferences in the safety domain.

7.3 Evaluation of Research

7.3.1 Evaluation of Elaborated Concepts

The three core concepts underlying the research presented in this thesis are *Evidence*, *Inconsistency* and *Confidence*. Despite the wide use of these concepts in safety engineering practice, they are not clearly defined and elaborated in the context of safety cases. The elaboration of these concepts is evaluated primarily by providing running examples and comparison with existing work.

Evidence is the core subject under study in this thesis. In particular, our focus is on a common type of evidence – safety assessment models. Running examples of the representation of evidence and the elicitation of evidence assertions are provided in Chapter 4. The establishment of the definition of evidence in a safety case is grounded on the comparison and analysis of existing conceptualisation of evidence in philosophy, law, evidence-based medicine and safety case guidance, as described in Section 4.2.

Model inconsistency is one of the issues that may damage our confidence in the trustworthiness of evidence and the appropriateness of asserted evidence relationships. Running examples of different types of model inconsistency are provided in Section 5.2. The inconsistency defined in this thesis is formulated on the basis of discussions of the concepts of inconsistency in other domains and the diverse use of the term of inconsistency in system safety, as shown in Section 5.2.1. The typology of model inconsistency is also established partially on the basis of a peer-reviewed conference paper [194].

Increased confidence is the ultimate objective to be achieved through the application of the approach presented in this thesis. The conceptual framework of confidence in safety cases is established on the basis of the analysis of existing work on confidence associated with safety cases (as described in Section 2.7.5). An exemplar use of the product-branch of the framework is depicted in the model adequacy argument pattern in Section 6.9. Further extensive peer review is expected for the evaluation of the overall confidence framework.

7.3.2 Evaluation of Information Models

Three information models are developed for capturing the core data and features of the elaborated concepts: a core data meta-model of safety assessment artefacts, a model of evidence interfacing arguments and the source data of evidence items, and an information model organising the interrelated data that is needed and emerges during cross-model inconsistency analysis. Running examples and comparison with existing safety meta-models are also employed in the evaluation of the core data meta-model.

The diagrammatic views and the specification scripts of the three information models are presented in Appendix B. The classes and associations defined in EMF conform to the information structures presented in the main thesis chapters. The Ecore version of the information models form the basis of structured documentation of the elaborated concepts in a unified format, which not only facilitates communication of the concepts in implementation but also indicates the feasibility of future tool support.

The expressive power of CoreDMM is illustrated with running examples in Section 3.4.2. The core data meta-model is compared with three typical existing domain meta-models in Section 3.4.3, for the illustration of its designed features in support of model evaluation and justification. This model has also been reviewed and presented at an international conference [193].

In addition, as described in Section 6.9 and Section 6.10, the contextual information required by the model adequacy argument pattern and the inconsistency analysis justification matches the information elements that can be provided by the three information models. It also provides a partial indication that the conceptual coverage of the three information models satisfies the need of intended active evaluation and rigorous justification.

The model of evidence can be evaluated in comparison with the OMG SAEM. Due to the fact that SAEM and SACM are evolving work in progress (as Beta versions), we have briefly discussed the insufficiency of SAEM as our application context in Section 2.7.4. Future evaluation will be implemented with the update of SAEM and SACM for the distinction of our model of evidence from that presented in SAEM. The information model of cross-model inconsistency analysis is specially designed in support of the inconsistency analysis method in the thesis. The comparison with existing work is not suitable for evaluating this model.

7.3.3 Evaluation of Structured Processes

Two structured processes are defined in the thesis, one for guiding cross-model inconsistency analysis, another for constructing two-sided safety arguments. The two processes are relatively novel in the safety domain as there are no similar methods or processes in existing safety practice or public academic work in safety. Hence, running examples and case study are adopted as the major means of evaluation for these two processes. In addition, the inconsistency analysis method has been presented at two MISSA open workshops in 2010.

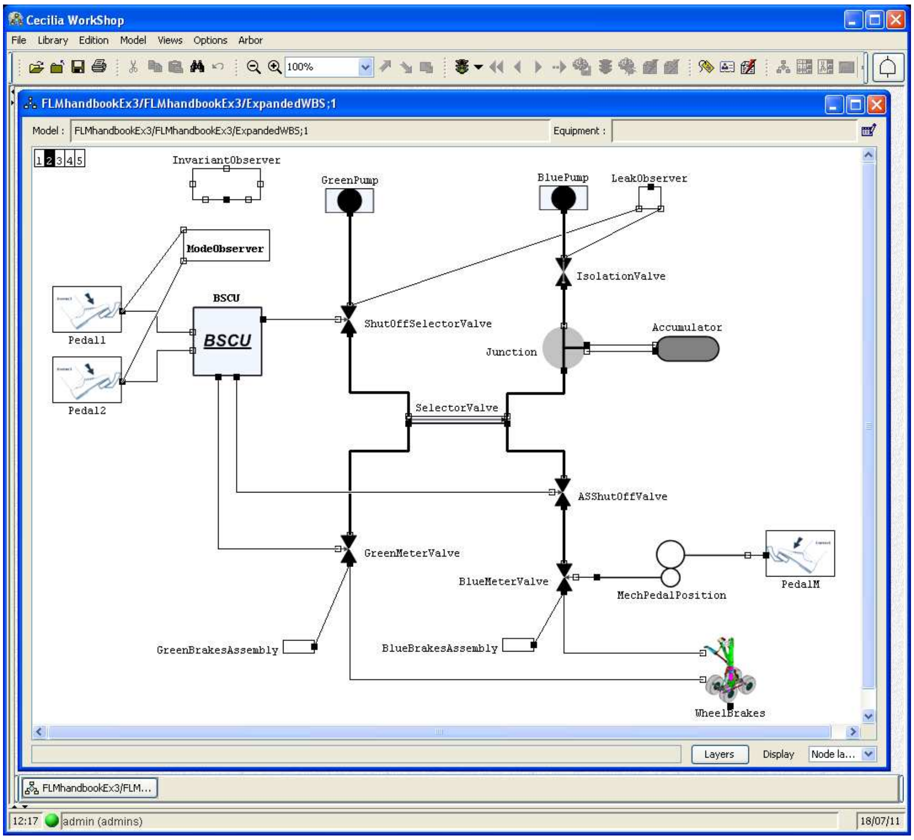
A demonstration example for the defined cross-model inconsistency analysis method is presented in Section 5.5. Inline examples of each of the expanded steps in the expanded argument construction process are provided in Section 6.6 as an illustration of how to consider alternative and negative argument elements during the development of two-sided safety cases. Even though the running examples are simple, they indicate the initial efficacy of the processes in a brief way.

The case study is a contiguous example that integrates both the evaluation of the inconsistency analysis method and the evaluation of the expanded argument construction. Appendix C provides in detail the description of the models under study and the key results obtained from each phases of the inconsistency analysis and the example arguments constructed for justification of the adequacy of the example models.

Case Study Details

Model I of the case study is coded in AltaRica within OCAS Cecelia WorkShop[17]. The main graphical view of **OCAS.4761.WBS** is presented in Figure 69. The failure condition in this model under MCS analysis is '*FC-WBS2: Inadvertent application of brakes*'.

[17] OCAS Cecelia WorkShop is a tool environment for modelling with AltaRica developed by Dassault Aviation.



Model : FLMhandbookEx3/FLMhandbookEx3/ExpandedWBS;1

Equipment :

12:17 admin (admins) 18/07/11

Figure 69 OCAS.4761.WBS in OCAS Cecilia

Model II is a traditional fault tree. The top event is '*BSCU commands braking in absence of braking inputs and causes inadvertent braking*'. FT.4761.IB is presented graphically using OpenFTA [11], an open source tool for fault tree analysis. Figure 70 is the screenshot of the tree in OpenFTA. The detailed fault tree structure and the description of events of Model II are presented in Appendix C.3.

The overlapping concern of safety analysis (represented by two models) is '*Inadvertent braking caused by BSCU in absence of brake inputs*' (coded as **FC.casestudy**).

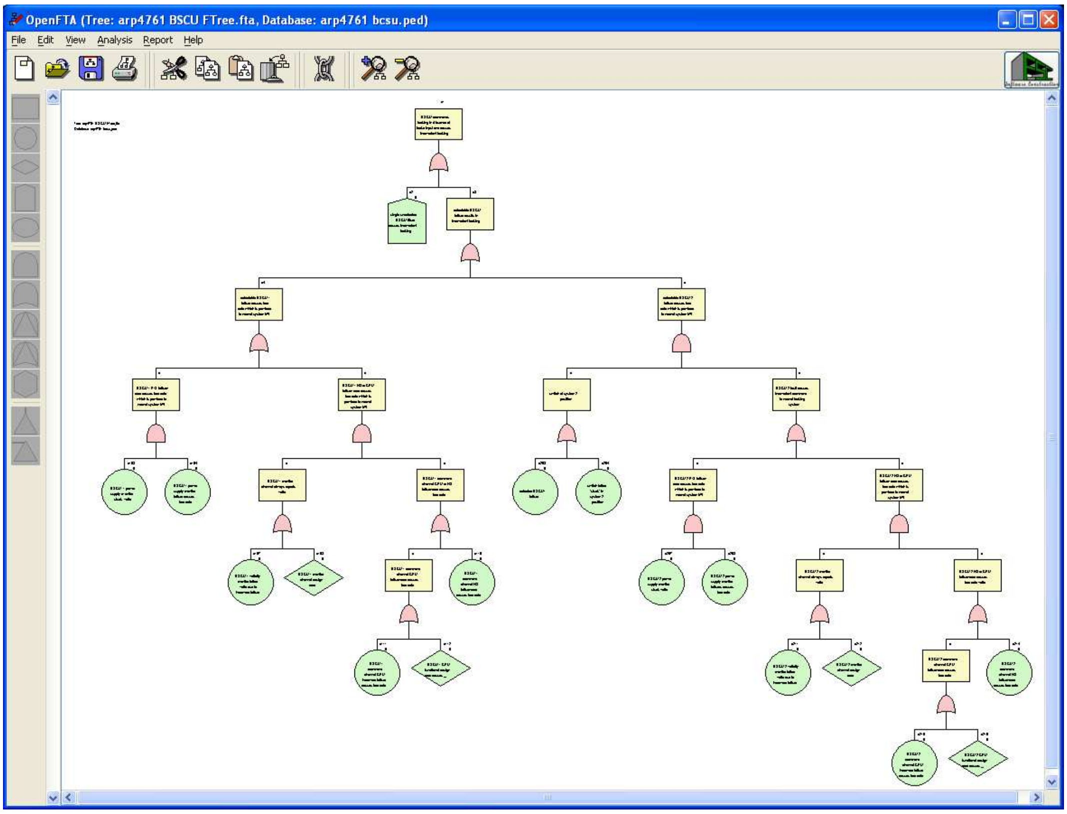

OpenFTA (Tree: arp4761 BSCU FTree.fta, Database: arp4761 bcsu.ped)
File
Edit
View
Analysis
Report
Help

Figure 70 FT.4761.IB in OpenFTA

A synopsis view of the case study on the inconsistency analysis is presented here. More details concerning the implementation are provided in Appendix C.4. The consistency relationships elaborated during the inconsistency analysis are presented in Table 14.

CaCR1: Each ‘condition’ shown in M_b has a corresponding ‘condition’ in M_a.

CaCR2: Each MCS in M_b associated with the top event has a corresponding MCS in M_a that contributes to the overlapping concern (**FC.casestudy**).

Table 14 Consistency Relationships of the Case Study

We set up correspondences between elements of the two models under study and examine whether CaCR1 and CaCR2 have been violated. After comparing the conditions and MCSs within the scope of inconsistency analysis, we have identified the following violations of defined consistency relationships:

- **six** violations for CaCR1 (e.g. **V1-1**: The e104 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS);

- **six** violations for CaCR2 (e.g. **V2-1**: {e204, e211, e214} does not have approximate corresponding MCS in OCAS.4761.WBS analysis results)

Six out of the 12 violations have been identified as inconsistencies against defined consistency relationship CaCR1 and CaCR2, while others have been dismissed due to the reasons provided. For example, **V1-1** is not identified as an inconsistency scenario, because e104 are not modelled within BSCU1 in OCAS.4761.WBS with the model assumption that "the power supply monitor of BSCU1 does not fail in mission"; **V2-1** is identified as an inconsistency scenario, because the Omission of e211 in the relevant MCSs of OCAS.4761.WBS is not allowed.

Based on the models and the inconsistency analysis results in this case study, we have illustrated how to take into account counter evidence and alternative information during safety case construction. The primary safety argument is developed on the basis of the safety goals extracted from ARP 4761 system description and safety analysis. The two models used in the case study are all presented as supporting evidence for a lower-level domain safety goal –'*No single failure of the BSCU shall lead to inadvertent braking*'. Figure 71 presents the primary safety argument used in the case study.

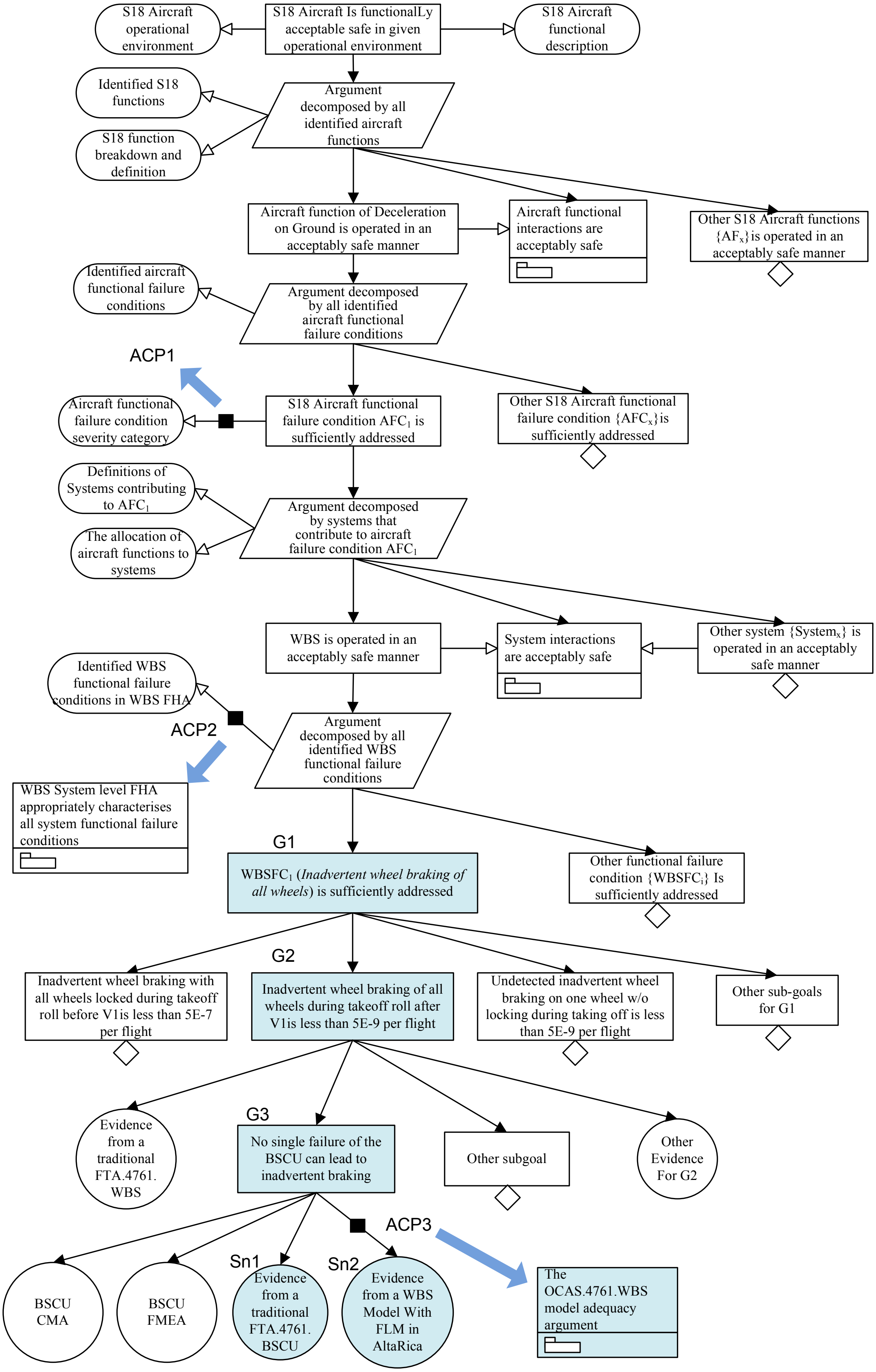

S18 Aircraft operational environment
S18 Aircraft Is functionalLy acceptable safe in given operational environment
S18 Aircraft functional description
Identified S18 functions
S18 function breakdown and definition
Argument decomposed by all identified aircraft functions
Aircraft function of Deceleration on Ground is operated in an acceptably safe manner
Aircraft functional interactions are acceptably safe
Other S18 Aircraft functions {AF_x}is operated in an acceptably safe manner
Identified aircraft functional failure conditions
Argument decomposed by all identified aircraft functional failure conditions
ACP1
Aircraft functional failure condition severity category
S18 Aircraft functional failure condition AFC_1 is sufficiently addressed
Other S18 Aircraft functional failure condition {AFC_x}is sufficiently addressed
Definitions of Systems contributing to AFC_1
The allocation of aircraft functions to systems
Argument decomposed by systems that contribute to aircraft failure condition AFC_1
WBS is operated in an acceptably safe manner
System interactions are acceptably safe
Other system {$System_x$} is operated in an acceptably safe manner
Identified WBS functional failure conditions in WBS FHA
ACP2
Argument decomposed by all identified WBS functional failure conditions
WBS System level FHA appropriately characterises all system functional failure conditions
G1
$WBSFC_1$ (*Inadvertent wheel braking of all wheels*) is sufficiently addressed
Other functional failure condition {$WBSFC_i$} Is sufficiently addressed
G2
Inadvertent wheel braking with all wheels locked during takeoff roll before V1is less than 5E-7 per flight
Inadvertent wheel braking of all wheels during takeoff roll after V1is less than 5E-9 per flight
Undetected inadvertent wheel braking on one wheel w/o locking during taking off is less than 5E-9 per flight
Other sub-goals for G1
Evidence from a traditional FTA.4761. WBS
G3
No single failure of the BSCU can lead to inadvertent braking
Other subgoal
Other Evidence For G2
ACP3
BSCU CMA
BSCU FMEA
Sn1
Evidence from a traditional FTA.4761. BSCU
Sn2
Evidence from a WBS Model With FLM in AltaRica
The OCAS.4761.WBS model adequacy argument

Figure 71 Example Primary Safety Argument in the Case Study

The model adequacy argument for OCAS.4761.WBS has been constructed on the basis of the model adequacy pattern presented in Chapter 6. It is also the implementation result of the Step E7 of the expanded argument construction process. Figure 72 presents an overview of the model adequacy argument in ASCE environment [16].

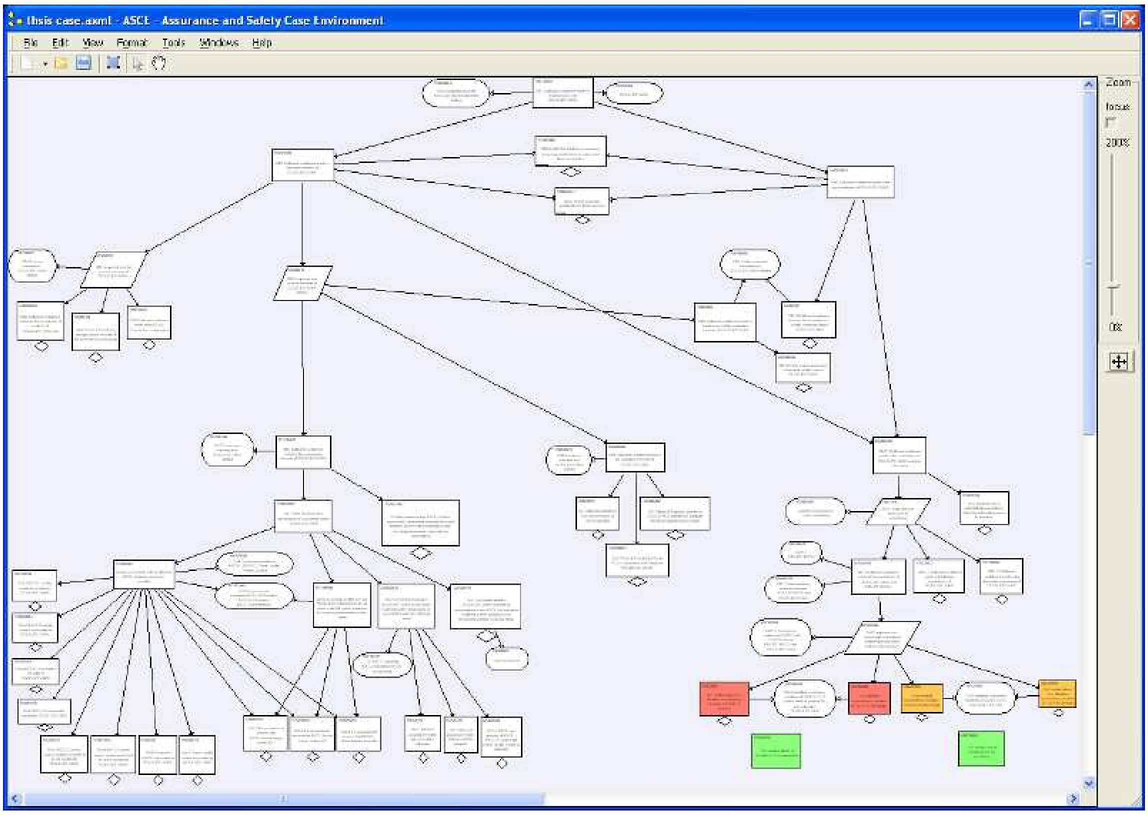

thsis case.axml - ASCE - Assurance and Safety Case Environment
File
Edit
View
Format
Tools
Windows
Help
Zoom
focus
200%
0%

Figure 72 Example Model Adequacy Argument in ASCE

Case Study Conclusions Regarding Inconsistency Analysis

The case study illustrates the feasibility of applying of the processes on a realistic industrial example. However, it also highlights some difficulties that emerged through the application of the cross-model inconsistency analysis method and the expanded argument construction process.

Besides the practical issues discussed in Section 5.6, the difficulties which emerged in application of the cross-model inconsistency analysis in the case study include:

1) A fundamental requirement of performing inconsistency analysis between two safety assessment models is the sufficient understanding of the models, including the modelling techniques adopted and the various construction elements, substance elements, contextual elements of the models under analysis. Otherwise, it is neither possible to determine the comparison scope and the consistency relationships, nor to set up correspondences between model elements.

2) The cross-check between two models based on different modelling techniques is necessarily grounded on the existence of reasonable ‘correspondences’ between the varied modelling constructs of two modelling techniques. If no such correspondences exist between the concepts used in two different modelling techniques, the models based on different techniques are not comparable. For example, the basic events described in a fault tree can be associated with three different modelling concepts (flow, state, event) [132] in the failure logic modelling (FLM) with AltaRica language. However, for the comparison purpose needed in the inconsistency analysis, we only map the ‘basic event’ concept in a fault tree with the ‘failure event’ concept in FLM with AltaRica.

3) The identification and definition of the scope of model elements to be examined through inconsistency analysis can be difficult. A common scenario in reality is that we rarely have two models with exactly the same modelling scope. Usually, the scope of the system elements being modelled or the scope of the safety concerns covered in two models is different. We need to rely upon the analysts to identify the overlapping model elements that represent similar or identical subjects in the real world. Machine support for this task is almost impossible, because it is impractical to expect the same naming convention in two separate modelling processes or to expect the selection of identical names for instances of model constructs in advance.

4) Acknowledging the different abstraction levels of model elements in two models is important during the inconsistency analysis. Insufficient understanding of the differences exhibited by model elements caused by different abstraction levels will hinder the specification of correspondences between two models. For example, two failure modes in a model may be expressed as one failure mode in another model; one system element in one model may be modelled as two model elements in another model. Therefore, the correspondence relationships between elements from two models are not limited to the ‘equivalence’ relationship (which means two elements from two models respectively can substitute each other without adding or losing any information or function associated with the original model elements). In many cases, ‘loose’ correspondences (a relationship that may enable one model element to work as a substitute of another model element approximately in one direction) are used for consistency checking purposes. For example, we assume FMa in Model A corresponds to FMb1 and FMb2 in Model B. Then we may use FMa to replace FMb1 and FMb2 for evaluating the MCS correspondences between Model A and Model B. But we cannot use FMb1 or FMb2 to replace FMa to support the analysis of MCS correspondences.

5) Definition of consistency relationships is crucial for the meaningfulness of the inconsistency analysis. Although it is defined as Phase 2 of the inconsistency analysis process, it interplays with Phase 1 in practice. In some cases, we identify models for inconsistency analysis and clarify the inconsistency analysis scope before the definition or selection of consistency relationships. In some other cases, we have prior knowledge of some generic consistency relationships, thus models can be selected and the analysis scope can be defined in accordance with the expected consistency relationships.

6) It is impractical to have a complete set of potential consistency relationships between two models. Therefore, the evidence result assertion based on the inconsistency analysis results cannot support a claim of absolute consistency between two models, but only one of consistency within the limits of the consistency relationships identified and examined in the analysis (as shown in the model adequacy argument pattern in Section 6.9).

7) In terms of concrete consistency relationships, those addressing condition coverage usually can be examined exhaustively in cross-model inconsistency analysis; whereas those addressing the logical combinations of conditions (such as MCS) are impractical to examine exhaustively in many cases. One reason for that is that it takes a long time to walk through all high-order MCS (e.g. all MCSs of a complicated model whose cardinality is higher than 4). Another reason for it is that if the correspondences between conditions are complicated (e.g. in other types such as ‘is the component of’ or ‘is the cause of’ rather than simple ‘equivalence’ only), the comparison of high-order MCSs can be difficult without tool support.

8) Defining correspondences between elements from two models can be time-consuming and error-prone. It requires careful work by an analyst, even if he knows the details of both models very well. In most cases, the modellers of the two models under study and the inconsistency analyst need to work together to ensure the ‘correctness’ of the correspondences defined. Otherwise, there can be little confidence in the results of the inconsistency analysis.

9) The implementation phase of the analysis method can be labour-intensive and needs precision if the models are complicated. However, we can never expect full machine support for the inconsistency analysis method. As we see from the case study, each phase of the inconsistency analysis involves human judgement. Due to the diversity of data formats and conceptual frames used by different safety assessment models, it is also not possible to have a fully automatic inconsistency analysis in reality. But to explore the potential of tool support for some of the checking steps is very necessary in order to

release human experts from repetitive search and match work to focusing on the more intellectual part of the inconsistency analysis.

Interestingly, the case study of the application of the inconsistency analysis method also reveals some issues concerning FLM in AltaRica and FTA. For example, it is convenient to retrieve the associations between a ‘condition’ and a ‘system element’ in the FLM in AltaRica, but time-consuming for the user to define such associations manually in a traditional fault tree (errors may be introduced in the manual association-setting process). It is inappropriate to consider the correspondences between ‘failure events’ in an AltaRica model and the ‘basic events’ in a traditional fault tree alone; the correspondence cannot be done without analyzing the relationships of the ‘state’ and ‘flow’ elements in an AltaRica model and the ‘basic events’ of a fault tree.

Case Study Conclusions Regarding Expanded Argument Construction

The case study results of argument construction are presented in Appendix C.5. The results of the case study of inconsistency analysis are employed in the demonstration example of argument construction. The case study illustrates the applicability of the expanded process and the generic structure of argument patterns. The function of evidence assertions are also illustrated with examples on the basis of the analysis of lower-level supporting goals in the example argument in Appendix C.5.

Besides the practical issues discussed in Section 6.6.12, the difficulties which emerged during the application of the expanded argument construction process in the case study include:

- **The work load**. Incorporating counter evidence in a safety case involves significant efforts and is experience-based. The progress of an argument may be slow down due to the consideration of counter evidence.

- **The scope of search** for counter evidence or alternative information. There is no clear boundary of potential information sources needed to be considered in those expanded steps. The efforts on counter considerations and rigorous justification should be proportionate to the risk associated with an argument element stated. However, this can be difficult to judge in practice.

7.3.4 Evaluation of Argument Patterns

Three argument patterns addressing different level of detail in argument structures are defined in Chapter 6. As described in Section 6.7.2, the argument patterns developed during the study have primarily been constructed on the basis of analysis of the problem domain and inspirations from informal arguments. Extensive application in practical context and designed peer reviews are significantly desired in the evaluation of these argument patterns.

Within the time-scale of the study, the following evaluation is performed for the argument patterns. The two-sidedness argument pattern was applied in the development of the model adequacy argument pattern. The model adequacy argument pattern is primarily evaluated through peer review during the MISSA project. The inconsistency analysis adequacy pattern is generated on the basis of the analyses of the inconsistency analysis method. The model adequacy argument pattern and the inconsistency analysis adequacy pattern have been exercised in a case study based on the models presented in Appendix C.2 and the results of inconsistency analysis presented in Appendix C.4.

The model adequacy argument pattern has experienced more extensive peer review than have the other two patterns. Early forms of the model adequacy argument pattern were supplied to project partners in the MISSA project. The model adequacy pattern was peer reviewed by both research and industrial project partners in later phases of the project. There were also trial uses of two patterns developed in MISSA project by Mr Keval Mehta, an intern at Airbus (UK) for supporting the construction of primary safety arguments and model adequacy arguments at the SSA (System Safety Assessment) stage. The model adequacy pattern in the thesis is a refined version of the project work, incorporating informal feedbacks primarily from Airbus (UK), EADS APSYS, ONERA, and ALENIA). The key goals in the model adequacy pattern has also been presented as model justification concerns in a peer-reviewed conference paper [196].

7.4 Conducting Further Evaluation

The research outputs in the thesis are evaluated through some forms of evaluation for their efficacy and practicality during the study. However, we are aware that further evaluation is desired, as outlined below, in order to achieve a more informative assessment of the effectiveness and practicality of the overall structured approach if time and resources are not constrained:

- Exposure of the structured approach to a wider community of researchers and practitioners for peer reviews, especially the applicability of the inconsistency analysis, the expanded argument construction, and the argument patterns;
- Implementation of the structured approach with more case studies carried out in an industrial context, especially the inconsistency analysis method and the expanded argumentation process;
- Further evaluation of the confidence conceptual framework through questionnaire-based peer review for examining the adequacy and sufficiency of the confidence factors that are presented in the framework;
- Comparison with the first release of the OMG SACM when it becomes publicly available for investigating the applicability of EviM.

7.5 Summary

This chapter reports on the evaluation of the efficacy and practicality of a structured approach of integrating and justifying safety assessment models within safety cases. The structured approach, which is based on three types of research outputs, has been exposed to a number of modes of evaluation, according to the features of different types of outputs and the time and resources available.

It is demonstrated by the evaluation activities that the conceptual models defined in Chapter 3 and Chapter 4 are fit for the purpose of concept clarification and the need of integrating argument and evidence; the inconsistency analysis method defined in Chapter 5 is structured and feasible; the expanded argument construction process and the argument patterns are practical. In addition, we acknowledge the need for further evaluation of the effectiveness, benefits and practical issues of the approach in an industrial application context.

8 Conclusions and Future Work

8.1 Thesis Contributions

This thesis has defined and demonstrated a structured approach to establishing confidence in safety assessment evidence within safety case development. The contribution of the thesis lies in five principal areas:

- Definition of three meta-models to support the definition and integration of safety assessment evidence with arguments
    - A safety assessment core data meta-model (CoreDMM) (presented in Chapter 3)
    - A model of evidence (EviM) (presented in Chapter 4)
    - An information model of cross-model inconsistency analysis (presented in Chapter 5)
- Definition of a conceptual framework of confidence in a safety case that captures key issues to be justified in structured confidence arguments (presented in Chapter 4 and Chapter 6)
- Definition and evaluation of a structured cross-model inconsistency analysis method for rigorous scrutiny of safety assessment models (presented in Chapter 5)
- Definition and evaluation of an expanded safety argument construction process that incorporates alternative and opposing information needed by a balanced and defensible two-sided argument (presented in Chapter 6)
- Development and evaluation of three argument patterns in support of more systematic and structured justification of the adequacy of safety assessment models, which utilises the data elements organised by the three meta-models (presented in Chapter 6).

8.1.1 Conclusions on Definition of Meta-Models

As distinct from other existing meta-models in the safety domain (e.g. meta-models in [46, 79, 212]), the safety assessment CoreDMM defined in Chapter 3 captures a comprehensive set of data elements that are necessary for cross-model inconsistency analysis and confidence argument construction. As described in Chapter 5 and Chapter 6, the data elements of CoreDMM have been referenced in the inconsistency analysis process and the model adequacy argument pattern.

The model of evidence (EviM) defined in Chapter 4 is founded on the analysis of existing definitions of evidence. Rather than emphasising the diversity of multiple forms of evidence source data, EviM has stressed the connotation of evidence and encompassed the two types of evidence assertions (namely evidence result assertion and evidence descriptive assertion) explicitly. In addition, EviM has distinguished an item of evidence from a set of evidence items and has differentiated the evidential properties of *EvidenceItem* and the evidential properties of *EvidenceSet*, both of which should be considered in assured safety arguments.

The information model that supports cross-model inconsistency analysis defined in Chapter 5 has captured the information elements associated with a cross-model inconsistency analysis. It is functionally linked with the safety assessment CoreDMM and the model of evidence. The safety assessment models under cross-model inconsistency analysis can refer to *SafetyAssessmentArtefact* in CoreDMM. The data elements in the information model of inconsistency analysis can be used as evidence or context in confidence arguments.

The three meta-models, working altogether, establish a foundation of managing the structured data elements associated with safety assessment models for their usage within safety cases. No previous published models or meta-models specifically tackle, or can be used for, this purpose in their own right.

8.1.2 Conclusions on Conceptual Framework of Confidence

Confidence in safety cases is a subjective issue. However, the conceptual framework of confidence in a safety case defined in Chapter 4 and refined in Chapter 6 explicitly presents a deconstructed justification of confidence within safety cases in general, and highlights more concrete and specific confidence issues that can be used for assisting comprehensive and systematic justification of confidence in a safety case.

Unlike the existing research on confidence in safety arguments or safety cases [34, 41, 97, 205] and quantified confidence in arguments [40, 60, 63, 134], the conceptual framework of confidence in a safety case presented in this thesis has encompassed both product elements and process elements in safety cases as contributing factors to confidence in practice and has incorporated both the supportive and opposing sides of argumentation. Existing models on confidence in safety cases have not taken into account the residual risks that are brought about by missing or omitted argument elements (potential arguments and evidence not presented but which can support or challenge claims in a safety case). Through addressing issues that arise from the explicit consideration of counter evidence in our confidence framework, the view of confidence has been broadened significantly. In fact, if two-sided argumentation is rigorously adopted in safety cases, the sufficiency and adequacy of the structures of some of the belief models (e.g. the Bayesian belief model in [63]) for safety arguments may be challenged fundamentally.

8.1.3 Conclusions on Structured Inconsistency Analysis Method

The cross-model inconsistency analysis defined in Chapter 5 has captured the key phases in a cross-model inconsistency analysis process. Each phase of the method has been elaborated with clear aims, inputs and outputs. The difficulties and pitfalls that may arise during the application of this method have been explored and discussed on the basis of a running example and a case study. The method provides a means of facilitating rigorous identification of cross-model inconsistency, motivating explicit structured documentation of inconsistency analyses and supporting justification of the adequacy of safety assessment models as evidence.

From the experience we obtained from MISSA project partners, we know that inconsistency analysis performed on different safety assessment models is usually an *informal* engineering activity. Compared with existing practice, the structured inconsistency analysis method helps conduct cross-model inconsistency analysis between safety assessment models in a more systematic manner and highlights the confidence and justification concerns associated with the inconsistency analysis itself.

The method can be adopted within, or outside of, the context of a safety case. The consistency of diverse evidence items (in particular, qualitative safety assessment models presented as evidence or context in a safety case) can be examined more rigorously with this method. Existing literature on evidence in safety cases has not addressed the inconsistency issue between evidence items with a structured method before. This structured method can prompt a more active consideration and exploration of inconsistency between evidence items used within a safety case.

8.1.4 Conclusions on Expanded Safety Argument Construction Process

The expanded safety argument construction process is an improvement of the existing Six-Step GSN method [117]. Common safety case development practice typically focuses on positive safety arguments and supporting evidence, as shown by many publicly accessible safety cases and safety case patterns (e.g. [73, 117, 126]). Counter evidence or counter considerations, as discussed in Section 6.3 and Section 6.5, *have* been considered but usually at the safety case review stage (late in a safety case lifecycle), rather than been addressed as a necessary issue of explicit concern during safety case development. In this thesis, the original Six-Step method has been expanded with nine extra steps. Three of them (Step E2, Step E5 and Step E8 in Figure 51 in Chapter 6) specifically consider counter or alternative information relevant to a safety argument; the other six (Step E1, Step E3, Step E4, Step E6, Step E7 and Step E9 in Figure 51 in Chapter 6) focus on providing justification of argument elements addressed in a safety case. In addition, two steps of the original Six-Step method have been enforced along with the expanded process (namely, consideration of alternative strategies at Step 3 and justification of inferences at Step 5). As a result, the expanded process has significantly extended the scope of issues being explicitly addressed in the development safety cases.

With the expanded steps, the two-sided argumentation and the confidence argument construction are coherently and methodically integrated within traditional safety case development practice. The expanded argument construction process also helps us to avoid potential fallacies (e.g. omission of key evidence) or potential biases (one-sided argument) in safety arguments. In short, the expanded argument construction process instils more rigour into to existing guidance on structured argument construction and can help the development of more balanced and defensible safety cases than those established through existing safety case practice.

8.1.5 Conclusions on Argument Patterns

Three argument patterns accompanying the expanded argument construction have been presented in Chapter 6. The Two-Sidedness Argument Pattern consists of two parts (as presented in Section 6.8). The two-sided safety argument structure can be used for citing both supporting evidence and counter evidence in a primary safety argument. The two-sided confidence argument structure can be used for justifying the confidence in the primary safety argument; it is established on the presence of supporting evidence and the resolution of counter evidence (either absence, or discharge, or tolerance, of items of counter evidence). This is a *novel* pattern that stresses the consideration and justification of counter evidence in a safety case and provides a way of describing positive claims associated with items of counter evidence in a confidence argument.

The Model Adequacy Argument Pattern comprises three parts. The post-modelling part (as presented in Section 6.9.2) has addressed the details of both the enactment and the output of a safety assessment modelling process in the model adequacy argument structure. The model consistency part (as presented in Section 6.9.3) addresses the decomposition of confidence in model consistency on the basis of inconsistency classification and cross-model inconsistency analysis as defined in Chapter 5. The structure of this part itself is an extension and exemplar use of the two-sidedness argument pattern. The pre-modelling part addresses the justification of the modelling method and the modelling tool adopted in safety assessment as the context of the post-modelling part. Importantly, justification concerns regarding the confidence associated with the use of safety assessment models (qualitative models only) as evidence within a safety case has been addressed in more detail than ever before in this three-part argument pattern. The more concrete but still generic sub-goals presented in this argument pattern can facilitate active thinking during model construction and justification and drive more rigorous model annotation and documentation during safety assessment modelling.

The Cross-Model Inconsistency Adequacy Argument Pattern addresses the key concerns that affect the confidence in adopting inconsistency analysis results as evidence for or against a model adequacy claim. It provides substantial guidance and support to the last phase of the inconsistency analysis method defined in Chapter 5. It also deepens our understanding of the potential weakness and limitations of the cross-model inconsistency analysis performed.

The three argument patterns directly utilise the structured data elements defined in the three meta-models and can be instantiated during the expanded argument construction process and the cross-model inconsistency analysis process.

Through the evaluation of the research outputs presented in Chapter 7, it is demonstrated that the overall structured approach (the integration of structured information, structured processes and structured guidance defined in this thesis) is rigorous and practical in providing support for the systematic integration and justification of safety assessment models in order to establishing better confidence in safety assessment practice, as stated by the thesis proposition initially presented in Chapter 1.

8.2 Areas of Future work

In addition to the contributions that the thesis has made, there are five interesting and promising areas that have emerged as a result of the research outputs presented in this thesis:

- Structured expression of evidence assertions
- More sophisticated and informative relationships of correspondence specification in inconsistency analysis
- Potential (partial) mechanised tool support for inconsistency analysis
- Analysis and justification of quantitative safety assessment models
- Quantification of confidence establishment

8.2.1 Structured Expression of Evidence Assertions

In Chapter 4, evidence assertions are defined and elaborated with examples. However, both types of evidence assertions (evidence result assertions and evidence descriptive assertions) have been described by natural language in the thesis. In order to facilitate structured and unambiguous communication of the meaning of evidence assertions and integration of EviM with a domain model and to expand the applicability of evidence assertions, we could explore using SBVR (Semantics of Business Vocabulary and Business Rules) [157] to define potential vocabularies and rules that could be used in evidence assertions, similar to the recent efforts by the OMG on using SBVR in SAEM [158]. Evidence assertion templates in structured English for different evidence types may be provided and applied for better integration of evidence and argument. This would also be useful supplement work to the research in formalism in safety cases (e.g.[142, 176, 192]).

8.2.2 Sophisticated Correspondence Specification

The consistency relationships between two models under inconsistency analysis are formulated on the basis of domain knowledge. It is identified through the case study of cross-model inconsistency analysis that the types of correspondences (a pair of data elements of two safety assessment models that are linked with specified relationships) that are associated with the defined consistency relationships can be more complicated than the simple 'equivalent to' relationships. For example, a 'condition' may need to be associated with another in the corresponding model that is described as its 'cause' or 'effect' or 'failure mode' of its parent system element. Some sophisticated correspondence relationships between model elements are very interesting, e.g. unidirectional correspondences, bidirectional correspondences, synchronized correspondences, cause-effect correspondences, or parent-component correspondences. Further investigation of these potential types of correspondences and their impact on reasoning using consistency relationships, would be beneficial to expand the scope of inconsistencies that can be inferred for examination. These sophisticated correspondences are at the model instance level, which are different from, but may be able to benefit from, the research on mapping languages [50] and model comparison [125] in the MDE domain.

8.2.3 Mechanised Tool Support for Inconsistency Analysis

In the thesis, the walkthrough of the inconsistency analysis method with examples is conducted manually. However, as reflected in the case study, it can be difficult for analysts to handle some of the tasks in Phase 3 (the implementation of model comparison) of the method if the scale of the models under analysis is large. Manually inferring and examining the correspondence pairs for consistency violations can be error-prone. Mechanised model comparison and inconsistency checking may be established on the basis of exchangeable data format and unified naming conventions and may relieve analysts from repetitive and stereotyped tasks. A mapping tool and a comparison tool in support of inconsistency analysis partially have been developed in MISSA project [133]. However, the tools are specialised for two models specified in AltaRica and a specific model consistency relationship. To examine inconsistency between models that are based on varied modelling constructs and documented according to different data formats, tool support is also very necessary, but much more difficult. It is worthwhile to explore the potential of introducing modern MDE techniques to aid some analysis steps if more work is undertaken in safety concept mapping and data format unification. Although it is acknowledged that completely automated model comparison is unachievable [166] due to the semantic issues in models, further work in this area may significantly improve the efficiency of inconsistency analysis and help to formulate and refine a Data Interchange Format of common safety assessment models at the implementation level of model comparison.

8.2.4 Analysis and Justification of Quantitative Models

The subject under study in this thesis is qualitative safety assessment models. However, the research outputs of the thesis can serve as the qualitative foundation of the analysis and justification of quantitative safety assessment models that are also important and common in safety assessment practice. It would be worthwhile to investigate and extend the work on structured inconsistency analysis to those quantitative models. For example, it would be interesting to explore the quantitative consistency relationships (with numerical features) that may exist between two quantitative safety assessment models (e.g. the expected value range, gap, or change of corresponding failure rates in two models). It would also be practical to adapt and extend the work on structured model justification for quantitative models. For example, integrating existing model validation research on quantitative models and simulation [28, 151], we can develop an argument pattern for justifying the adequacy of quantitative safety assessment evidence based on the model adequacy argument pattern presented in this thesis.

8.2.5 Quantification of Confidence Structures

It is known from experience gained through our partners on the MISSA project that engineers can feel uncomfortable with phrases of ‘sufficient confidence’ associated with the argument patterns and the confidence framework defined in this thesis. Quantified confidence is highly desirable for wider industrial application of the work presented. Through the quantification of the qualitative structure of confidence, the practicality of the qualitative structure may be testified or improved (e.g. identification of the (types of) parameters that should be measured). Furthermore, the formation of quantitative confidence models (e.g. a Bayesian belief model associated with model inconsistency analysis) can help us perform machine-supported reasoning of confidence. It can also shed light on the mechanisms of confidence propagation between factors that can affect (positively or negatively) the confidence in an argument.

In addition, as described in Section 7.4, it is also desirable to carry out further evaluation in a real industrial context, in order to demonstrate the potential benefits, effectiveness and practical issues associated with the approach defined in the thesis.

8.3 Finale

A little knowledge of safety-critical systems and the associated safety assessment models is a dangerous thing. A little acknowledgement of our limited knowledge of them is another dangerous thing. Confidence in safety assessment models as evidence in safety cases can be established explicitly and reasonably using the structured approach presented in the thesis. However, the establishment and demonstration of confidence in safety arguments and evidence may only be implemented as rigorously as reasonably practicable (ARARP), as permitted by time and resource constraints and in a way that is commensurate with risk associated with a safety case. In addition, the approach defined in this thesis can help engineers better manage confidence issues, but cannot totally take away the issue of subjectivity (an issue at the heart of the nature of confidence in safety cases). There is approximately fifty years of experience in both safety analysis and safety case development (more in some domains than in others) and it will take a considerable time to determine if these established practices can usefully be shaped by the contribution of the thesis.

Appendix A Argument Patterns

A.1 Two-Sidedness Argument Pattern

Two-Sidedness Argument Pattern			
Author	Linling Sun, Tim Kelly		
Created	08/07/2011 11:41:00	**Last Modified**	10/12/2012 23:06:00

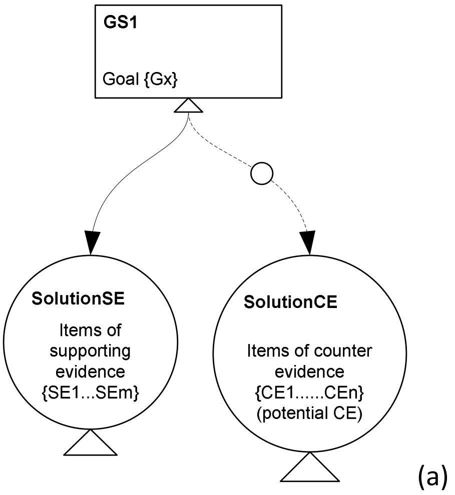
Intent	This pattern provides a generic argument structure for presenting how counter evidence is considered and represented in a primary safety argument and the associated confidence argument.
Motivation	This pattern was developed: • To present the relationship between potential items of evidence (either supporting evidence or counter evidence) and a domain safety claim • To clarify the goals to be justified for the backing of sufficient support from supporting evidence • To clarify the goals to be justified for the backing of the survival of the domain safety claim with consideration of counter evidence

Structure

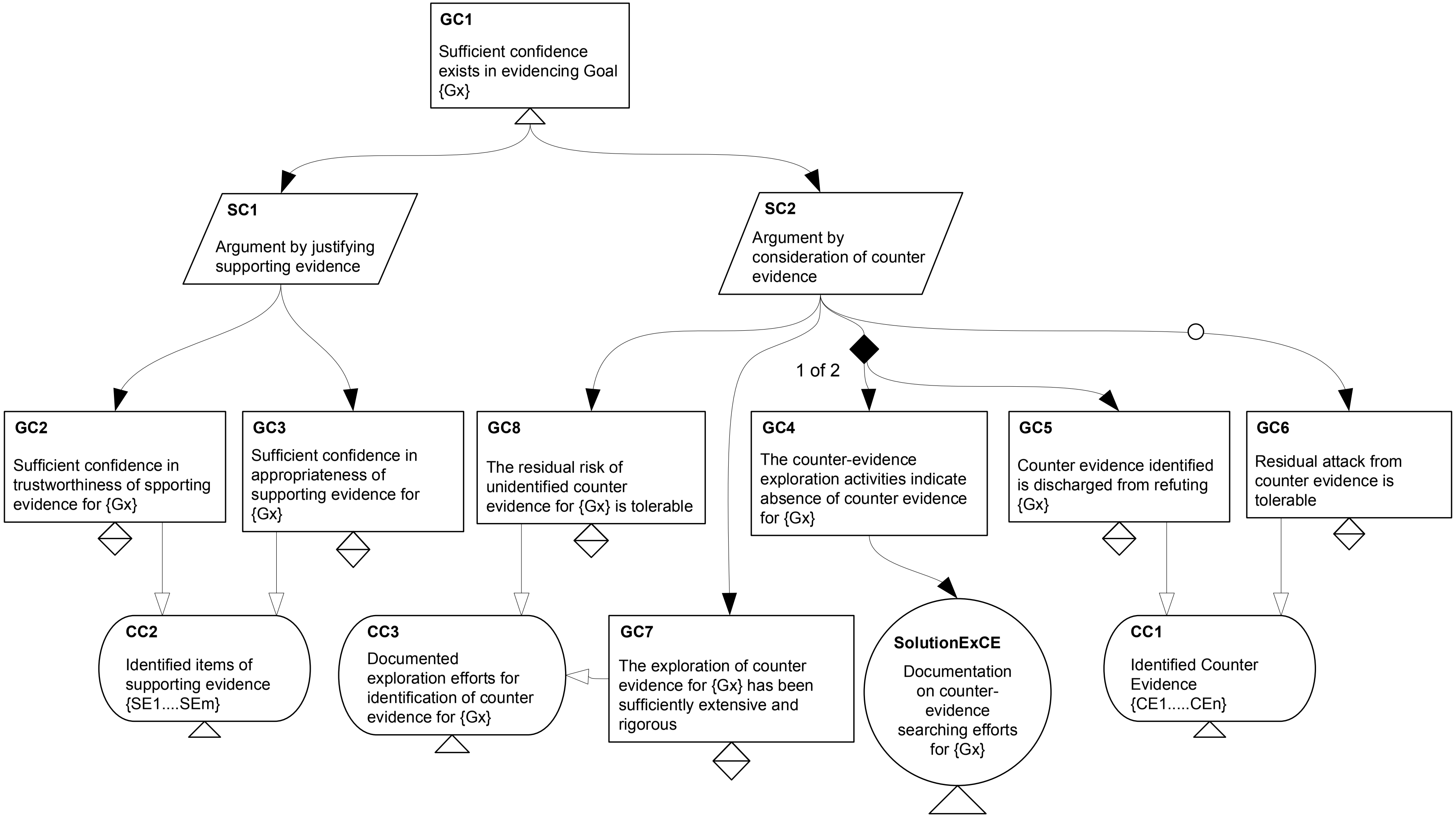
The structure of this pattern consists of two parts:

(a) the primary safety argument part

(b) the confidence argument part


GS1
Goal {Gx}
SolutionSE
Items of supporting evidence {SE1...SEm}
SolutionCE
Items of counter evidence {CE1......CEn} (potential CE)

(a)


GC1
Sufficient confidence exists in evidencing Goal {Gx}
SC1
Argument by justifying supporting evidence
SC2
Argument by consideration of counter evidence
1 of 2
GC2
Sufficient confidence in trustworthiness of spporting evidence for {Gx}
GC3
Sufficient confidence in appropriateness of supporting evidence for {Gx}
GC8
The residual risk of unidentified counter evidence for {Gx} is tolerable
GC4
The counter-evidence exploration activities indicate absence of counter evidence for {Gx}
GC5
Counter evidence identified is discharged from refuting {Gx}
GC6
Residual attack from counter evidence is tolerable
CC2
Identified items of supporting evidence {SE1....SEm}
CC3
Documented exploration efforts for identification of counter evidence for {Gx}
GC7
The exploration of counter evidence for {Gx} has been sufficiently extensive and rigorous
SolutionExCE
Documentation on counter-evidence searching efforts for {Gx}
CC1
Identified Counter Evidence {CE1.....CEn}

(b)

Participants	GS1	GS1 states a domain safety claim {G_x} to be assessed, which is at a level that can be supported directly by items of evidence. GS1 may be addressed both by supporting evidence and by counter evidence.
	SolutionSE	Items of supporting evidence {SE1…SEm} for {G_x}. The evidence result assertion of {SEx} should be capable of indicating that GS1 is true. There may be more than one items of supporting evidence.
	SolutionCE	Items of counter evidence {CE1…CEn} for {G_x}. If no counter evidence is identified, this node does not appear. The evidence result assertion of an item of counter evidence should be capable of challenging the truth value of GS1. There may be more than one item of counter evidence.
	GC1	GC1 defines the overall justification objective that is required for establishing sufficient confidence that GS1 is true. It takes into account both asserted evidence relationships and asserted counter evidence relationships that are associated with GS1.
	SC1	Provided at this point to support the decomposition of claim GC1 on the basis of the feature of asserted evidence relationships and the feature of supporting evidence (SolutionSE).
	SC2	Provided at this point to support the decomposition of claim GC1 on the basis of the consideration of counter evidence, e.g. whether there are no attacks from items of counter evidence for GS1, whether GS1 survives being true against the challenges from items of counter evidence (SolutionCE).
	GC2	A sub-goal that supports GC1 that is focused on the trustworthiness of items of evidence. This goal can be further decomposed for each item of supporting evidence. It can also be addressed by considering various factors that affect the trustworthiness of a specific item of supporting evidence. For a typical form of supporting evidence, safety assessment models, the lower-level generic argument structure of the development of this goal is presented in **GM2** of Part (a) of the model adequacy argument pattern.

	GC3	A sub-goal that supports GC1 that is focused on the appropriateness of the asserted relationships between GS1 and SolutionSE. This goal can be further decomposed for each asserted supporting evidence relationship. It can also be addressed by considering various factors that affect the appropriateness of individual asserted evidence relationships or the appropriateness of asserted evidence relationships as a collection. For a typical form of supporting evidence, safety assessment models, the lower-level generic argument structure of the development of this goal is presented in **GM3** of Part (a) of the model adequacy argument pattern.
	GC4	A sub-goal that supports GC1 from the opposing side of the argument. GC4 and GC5 are mutually exclusive. Even if no SolutionCE is shown in the primary safety argument, we still need evidence of absence of counter evidence to support Goal GC4.
	GC5	A sub-goal that supports GC1 by discharging items of counter evidence with acceptable reasons.
	GC6	This goal exists if there were identified items of counter evidence that were not discharged. The residual attacks towards $\{G_x\}$ from undischarged items of counter evidence should be assessed. Residual attacks might be allowed due to trade-off decisions based on the cost, efforts, new issues brought by resolving some items of counter evidence. Tolerable attacks from counter evidence towards $\{G_x\}$ are still attacks, but they undermine confidence in the overall evidencing result of $\{G_x\}$ only in a tolerable degree.
	GC7	A goal that states the sufficiency of efforts on counter evidence identification. The coverage of potential sources of counter evidence and the rigour of the counter evidence inquiry efforts should be justified to support this goal.
	GC8	A goal that indicates the consideration and desired state of the epistemic limitation of our knowledge of counter evidence for $\{G_x\}$. It is a complementary goal generated from the opposing viewpoint of GC7.
	CC1	A list of items of counter evidence for $\{G_x\}$ identified, as presented in SolutionCE. SolutionCE and CC1 may refer to the same document. But the roles of the same source information are different in the two situations.

	CC2	A list of items of supporting evidence for {G_x} identified, as presented in SolutionSE. SolutionSE and CC2 may refer to the same document. But the roles of the same source information are different in the two situations.
	CC3	Context of documented exploration efforts for identification of counter evidence for {G_x}. An implemented search activity should be documented even if no counter evidence is identified by the search activity.
	SolutioExCE	Solution reference to documented exploration aimed at the identification of counter evidence for {G_x}. CC3 and SolutionExCE may refer to the same document, but for different purposes.
Applicability	This pattern is applicable to the presentation evidence for a domain claim and to the justification of confidence in there being sufficient evidential supports for the domain claim, especially when counter evidence is considered during the argument construction process. Contextual information and evidence needed in the pattern include: • A list of items of supporting evidence presented for {G_x} • A list of items of counter evidence identified for {G_x} • Documentation of counter evidence identification efforts	
Collaborations	The pattern consists of two parts – the primary part and the confidence part. The confidence contains the justification of the elements in a primary safety argument and the further justification of elements within a confidence argument. Two goals (GC2, GC3) in the confidence part are further developed for a special type of evidence (safety assessment models) in Part (a) of the model adequacy argument pattern (GM2, GM3).	
Consequences	There are a number of undeveloped goals after the application of this pattern, e.g. GC2, GC3, GC5, GC6, GC7 and GC8.	
Implementation	Implementation of this pattern involves instantiation of the primary branch of the pattern first and then the confidence part. Contextual information should be clarified as required and the choice of GC4 or GC5 should be based on the results of counter evidence inquiry efforts.	
Related Patterns	See Part (b) of the Model Adequacy Argument Pattern for an example usage.	

A.2 Safety Assessment Model Adequacy Argument Pattern

Safety Assessment Model Adequacy Argument Pattern			
Author	Linling Sun, Tim Kelly		
Created	08/07/2011 11:41:00	**Last Modified**	10/12/2012 23:06:00

Intent	This pattern provides a generic argument structure for presenting how the adequacy of safety assessment models as supporting evidence is considered.
Motivation	This pattern was developed: • To clarify the goals to be justified for the backing of the adequacy of a safety assessment model as supporting evidence • To present the relationship between the content in pre-modelling justification and the content in post-modelling justification • To present the relationship between justification based on process elements of modelling and justification based on product elements of modelling • To present an example use of counter considerations as presented in the two-sidedness argument pattern • To clarify the goals to be justified for the backing of the model consistency with consideration of counter evidence

Structure

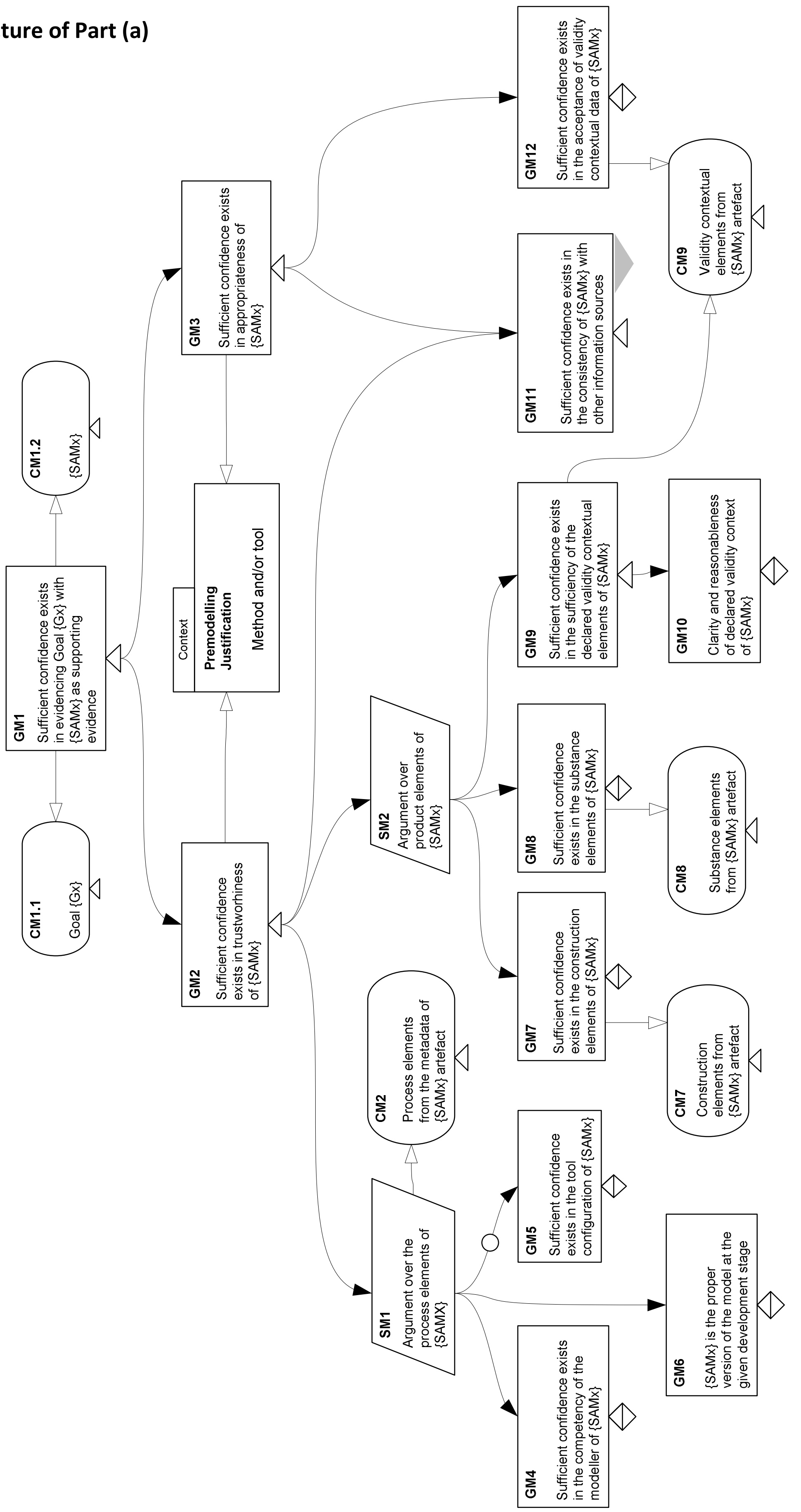
The structure of this pattern consists of three parts:

(a) the main view of post-modelling justification

(b) the branch view of model consistency justification

(c) the contextual module of pre-modelling justification

Structure of Part (a)


GM1
Sufficient confidence exists in evidencing Goal {Gx} with {SAMx} as supporting evidence
CM1.1
Goal {Gx}
CM1.2
{SAMx}
GM2
Sufficient confidence exists in trustworhiness of {SAMx}
GM3
Sufficient confidence exists in appropriateness of {SAMx}
Context
Premodelling Justification
Method and/or tool
SM1
Argument over the process elements of {SAMx}
CM2
Process elements from the metadata of {SAMx} artefact
SM2
Argument over product elements of {SAMx}
GM4
Sufficient confidence exists in the competency of the modeller of {SAMx}
GM5
Sufficient confidence exists in the tool configuration of {SAMx}
GM6
{SAMx} is the proper version of the model at the given development stage
GM7
Sufficient confidence exists in the construction elements of {SAMx}
CM7
Construction elements from {SAMx} artefact
GM8
Sufficient confidence exists in the substance elements of {SAMx}
CM8
Substance elements from {SAMx} artefact
GM9
Sufficient confidence exists in the sufficiency of the declared validity contextual elements of {SAMx}
GM10
Clarity and reasonableness of declared validity context of {SAMx}
GM11
Sufficient confidence exists in the consistency of {SAMx} with other information sources
GM12
Sufficient confidence exists in the acceptance of validity contextual data of {SAMx}
CM9
Validity contextual elements from {SAMx} artefact

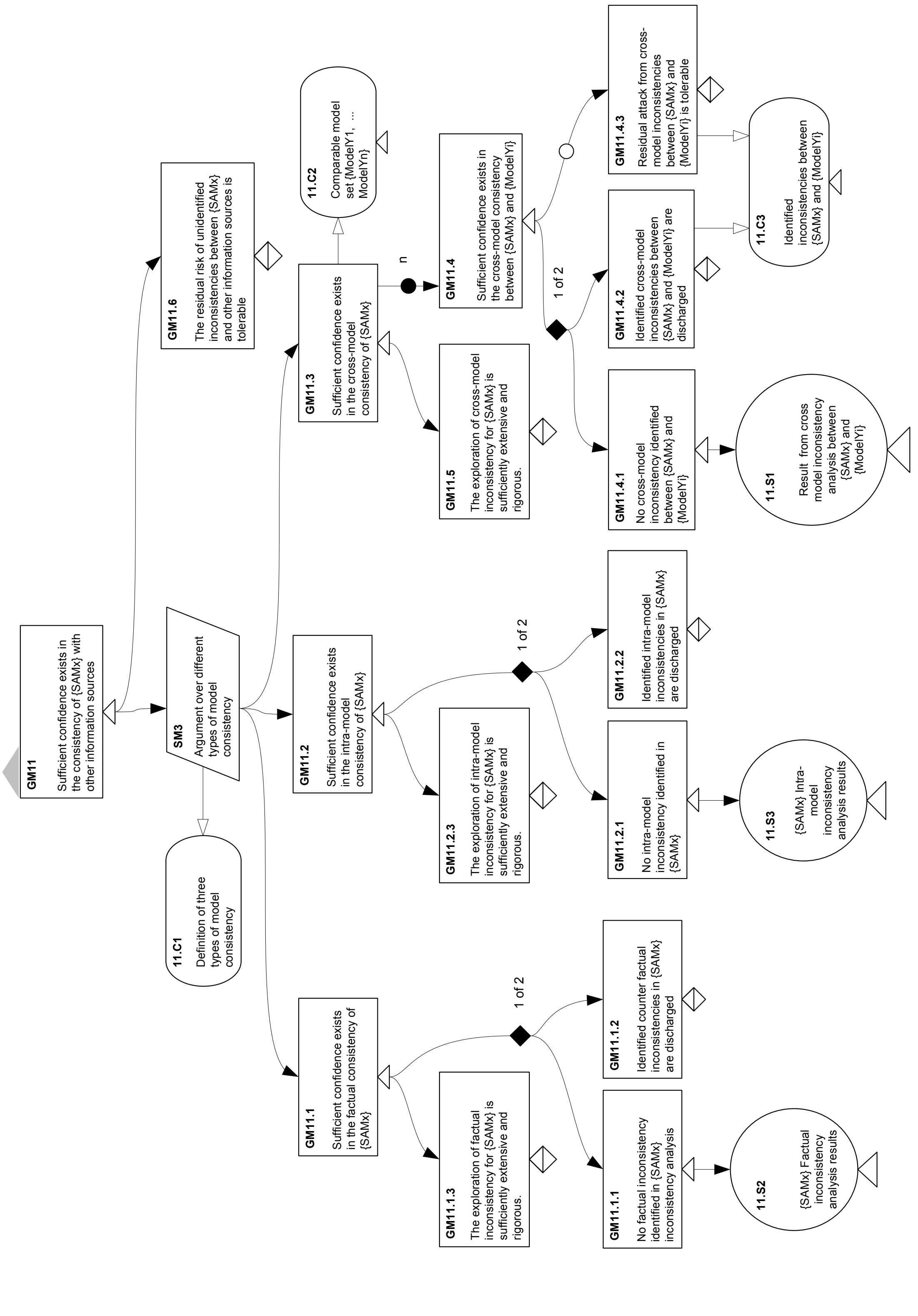
Participants of Part (a)	GM1	GS1 states a confidence claim to be assessed, which is specifically associated with one item of supporting evidence {SAMx} and a domain safety goal {G_x}. GM1 may be addressed by the trustworthiness of {SAMx} and the appropriateness of its use as evidence.
	CM1.1	The content of {G_x} as the context of GM1 that is available from the associated primary safety argument.
	CM1.2	The content of {SAMx} as the context of GM1 that is available from the associated primary safety argument.
	GM2	GM2 defines a confidence claim that is associated with the trustworthiness of {SAMx}. It is based on the context of the pre-justified modelling method and modelling tool selected for modelling the system in question.
	GM3	GM3 defines a confidence claim that is associated with the appropriateness of adopting {SAMx} as supporting evidence. It is also based on the context of the pre-justified modelling method and modelling tool selected for modelling the system in question.
	Premodelling Justification	A context module that captures the justification of the modelling method and tool selected for modelling the system under study.
	CM2	The contextual information that describes the process elements associated with the enactment of {SAMx} modelling process. Typical data items include ‘model version/date’, ‘modeller’, ‘modelling tool’, as shown in the subtypes of ‘MetaData’ in CoreDMM presented in Chapter 3.
	SM1	Provided at this point to support the decomposition of claim GM2 according to the associated process elements of {SAMx} modelling.
	SM2	Provided at this point to support the decomposition of claim GM2 according to the product elements of {SAMx} modelling. The product elements include not only the outcomes or processed modelling results, but also the decisions and assumptions adopted during modelling and the limitations of {SAMx}.

	GM4	A sub-goal that supports GM2 that is focused on the competency of the modeller of {SAMx}. This goal should be emphasized while arguing confidence on the basis of process elements. The relevant knowledge and experience of a modeller significantly affect the trustworthiness of {SAMx}.
	GM5	A sub-goal that supports GM2 that is focused on the tool configuration of {SAMx}. It exists if a modelling tool was adopted in the {SAMx} modelling process. This goal should be examined, especially when implementation parameters should be set or selected for data processing during modelling with the tool.
	GM6	A sub-goal that supports GM2 that is focused on the correct reference of {SAMx}. The latest version of {SAMx}, which is consistent with the current system design and operational situations, should be addressed in the primary safety argument, rather than any outdated versions of {SAMx}.
	GM7	A sub-goal that supports GM2 by examining the construction elements of {SAMx}. Depending on the method selected, the types of construction elements in a model could vary significantly. We can address this goal by evaluating the generic types of construction elements, assuming {SAMx} is represented on the basis of CoreDMM proposed in Chapter 3.
	GM8	A sub-goal that supports GM2 by examining the substance elements of {SAMx}. Depending on the method selected and application scenarios, the types of substance elements in a model may differ significantly. We can address this goal by evaluating some typical types of substance elements of common safety assessment models, assuming {SAMx} is represented on the basis of CoreDMM presented in Chapter 3.
	GM9	A sub-goal that supports GM2 by examining the declared validity context of {SAMx}. The information that demonstrates that we have a clear view of the boundary of the capability of a model must be explicitly stated and justified. The fulfilment of this goal enables a better understanding of the potential reasoning gap between {SAMx} presented and the support needed by Goal {G_x} in the primary safety argument.

	GM10	A sub-goal that supports GM9. It refines the content of GM9. The clarity and reasonableness (including sufficiency) of the declared validity context of {SAMx} should be justified.
	GM11	A goal that indicates the consistency of {SAMx} with other information sources. Depending on the concrete types of inconsistencies identified during various forms of consistency analysis, this goal may contribute to GM2 or GM3. It is further decomposed in the branch view of model consistency justification – Part (b) of this pattern.
	GM12	A goal that indicates the acceptance of declared validity context of {SAMx} while adopting {SAMx} as supporting evidence for Goal {G_x}. Using {SAMx} beyond its capability or its validity envelope means inappropriateness of the asserted evidence relationship between {SAMx} and Goal {G_x}. This goal should be considered in combination with GM9.
	CM7	The contextual information that describes the construction elements of {SAMx} artefact. Typical generic data items include 'condition', 'system element', 'logical relationship between conditions', as shown in the subtypes of 'ConstructionElement' in CoreDMM presented in Chapter 3.
	CM8	The contextual information that describes the substance elements of {SAMx} artefact. Typical data items include 'hazard set', 'MCS set', 'probability set', as shown in the subtypes of 'SubstanceElement' in CoreDMM presented in Chapter 3.
	CM9	The contextual information that describes the validity contextual elements of {SAMx} artefact. Typical data items include 'assumption', 'limitation', 'modelling purpose', 'modelling scope', 'data source', as shown in the subtypes of 'ValidityContext' in CoreDMM presented in Chapter 3.
Applicability of Part (a)	This part of the model adequacy argument pattern, the main view of post-modelling justification, is applicable to the presentation of layered claims for justifying the adequacy of safety assessment evidence with data elements obtained throughout a modelling process. The part (a) of the model adequacy argument pattern is decomposed under the situation that only one safety assessment model {SAMx} is	

	adopted as evidence for Goal {G_x}. Therefore, the sub-goals associated with the appropriateness of multiple items of supporting evidence are **not** included in Part (a) of this pattern. In case of the presence of multiple items of supporting evidence, more sub-goals related to the appropriateness of the overall evidencing sufficiency should be considered and generated in support of GM3, e.g. the independency between multiple items of evidence, the diversity of multiple items of evidence. Contextual information needed in Part (a) of the model adequacy argument pattern includes: • a contextual justification module of the adequacy of the selected modelling method and or modelling tool for {SAMx} • process data and product data of the {SAMx} modelling process This part of the pattern can be extended to be used as a reference or a source of inspiration for the justification of general qualitative models.
Collaborations of Part (a)	The model adequacy argument pattern Part (a) is in the context of the pre-modelling justification of the selected safety assessment modelling method and tool, Part (b). The GM11 in Part (a) of the model adequacy argument pattern is further developed in another separate view of the argument, Part (c). Two goals (GM2, GM3) in Part (a) of the model adequacy argument pattern are exemplar development of goals (GC2, GC3) in the two-sidedness argument pattern respectively.
Consequences	There are a number of undeveloped goals after the application of Part (a) of the model adequacy argument pattern: GM4, GM5, GM6, GM7, GM8, GM10, and GM12.
Implementation	Implementation of Part (a) of the model adequacy argument pattern involves instantiation of a variety of confidence claims. Contextual information from {SAMx} artefact should be clarified as required.
Related Patterns	Safety Assessment Model Adequacy Argument Pattern - Part (b) Safety Assessment Model Adequacy Argument Pattern - Part (c)

Structure of Part (b)

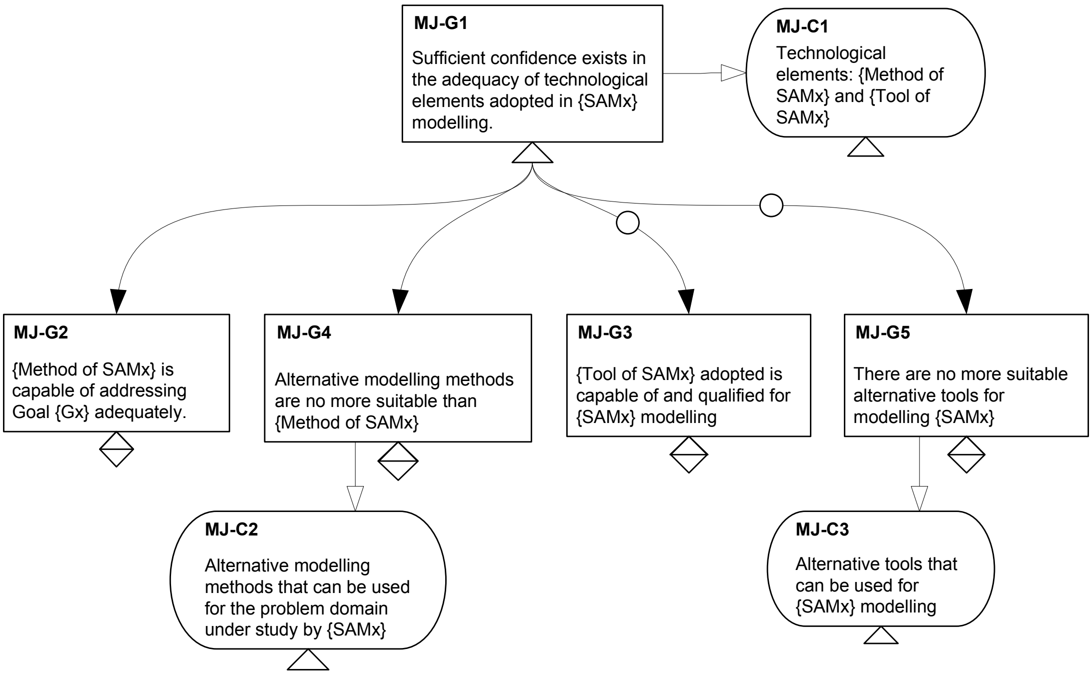

GM11
Sufficient confidence exists in the consistency of {SAMx} with other information sources
SM3
Argument over different types of model consistency
11.C1
Definition of three types of model consistency
GM11.6
The residual risk of unidentified inconsistencies between {SAMx} and other information sources is tolerable
GM11.3
Sufficient confidence exists in the cross-model consistency of {SAMx}
11.C2
Comparable model set {ModelY1, ... ModelYn}
n
GM11.4
Sufficient confidence exists in the cross-model consistency between {SAMx} and {ModelYi}
GM11.5
The exploration of cross-model inconsistency for {SAMx} is sufficiently extensive and rigorous.
1 of 2
GM11.4.3
Residual attack from cross-model inconsistencies between {SAMx} and {ModelYi} is tolerable
GM11.4.2
Identified cross-model inconsistencies between {SAMx} and {ModelYi} are discharged
11.C3
Identified inconsistencies between {SAMx} and {ModelYi}
GM11.4.1
No cross-model inconsistency identified between {SAMx} and {ModelYi}
11.S1
Result from cross model inconsistency analysis between {SAMx} and {ModelYi}
GM11.2
Sufficient confidence exists in the intra-model consistency of {SAMx}
GM11.2.3
The exploration of intra-model inconsistency for {SAMx} is sufficiently extensive and rigorous.
1 of 2
GM11.2.2
Identified intra-model inconsistencies in {SAMx} are discharged
GM11.2.1
No intra-model inconsistency identified in {SAMx}
11.S3
{SAMx} Intra-model consistency analysis results
GM11.1
Sufficient confidence exists in the factual consistency of {SAMx}
GM11.1.3
The exploration of factual inconsistency for {SAMx} is sufficiently extensive and rigorous.
1 of 2
GM11.1.2
Identified counter factual inconsistencies in {SAMx} are discharged
GM11.1.1
No factual inconsistency identified in {SAMx} inconsistency analysis
11.S2
{SAMx} Factual inconsistency analysis results

Participants of Part (b)	GM11	A goal that indicates the consistency of {SAMx} with other information sources.
	SM3	Provided at this point to support the decomposition of claim GM11 according to the types of model consistency as defined in Chapter 5.
	11.C1	Three types of model consistency for safety assessment models as defined in Chapter 5.
	GM11.1	A goal that defines a confidence claim that associated with the consistency of the content of {SAMx} and the accepted logic or facts existed in reality. It is further decomposed into GM11.1.1, GM11.1.2, and GM11.1.3.
	GM11.2	A goal that defines a confidence claim that associated with the internal consistency of the content of {SAMx} within itself. There should not be self-contradicted modelling elements within {SAMx}. It is further decomposed into GM11.2.1, GM11.2.2, and GM11.2.3.
	GM11.3	A goal that defines a confidence claim that associated with the external consistency of the content of {SAMx} with the content of other models. It is further decomposed into GM11.4 and GM11.5.
	GM11.4	A goal that defines cross-model consistency between {SAMx} and another model {ModelY$_i$}. It is further decomposed into GM11.4.1 and GM11.4.2.
	GM11.1.3 & GM11.2.3 & GM11.5	Goals stated that the efforts on the exploration of a specific type of model inconsistency (counter factual, intra-model, or cross-model) are sufficient, which may need to cover various consistency relationships, relevant information or data available and rigorous scrutiny. These goals are cases for **GC7** in the two-sidedness argument pattern.
	GM11.6	A goal that indicates the consideration and desired state of the epistemic limitation of our knowledge of inconsistency scenarios for {SAMx}. It complements goals on the sufficiency of exploration for inconsistencies, e.g. GM11.1.3, GM11.2.3, GM11.5. This goal is a case for GC8 in the two-sidedness argument pattern.

	GM11.1.1 & GM11.2.1 & GM11.4.1	Goals that support GM11.1, GM11.2, GM11.4 respectively. They indicate the absence of specific types of model inconsistency associated with {SAMx}. These goals are cases for **GC4** in the two-sidedness argument pattern.
	GM11.1.2 & GM11.2.2 & GM11.4.2	Goals that support GM11.1, GM11.2, GM11.4 respectively. They indicate the discharge of specific types of model inconsistency associated with {SAMx}. These goals are cases for **GC5** in the two-sidedness argument pattern.
	GM11.4.3	Goals that support GM11.4. It indicates that the residual risk associated with identified inconsistencies that are not discharged is tolerable. The goals associated with the residual risks of indentified inconsistencies are omitted for GM11.1 and GM11.2.
	11.C2	A list of models that can be used for cross-checking of model inconsistency between {SAMx} and them.
	11.C3	A list of identified inconsistencies models that can be used for cross-checking of model inconsistency between {SAMx} and them.
	11.S1 & 11.S2 & 11.S3	Solutions that represent various inconsistency analysis results provided. They are employed as evidence for the absence of inconsistencies. However, if there were inconsistencies identified during inconsistency analysis, these will serve as context of other goals that present the identified inconsistency scenarios.
Applicability of Part (b)	This part of the model adequacy argument pattern, the branch view of model consistency justification, is applicable to the presentation of layered claims for justifying the consistency of a safety assessment model with other information sources on the basis of inquiring three types of model inconsistency. Contextual information and evidence needed in Part (b) of the model adequacy argument pattern include: • the classification of model consistency for safety assessment models • the models adopted for the examination of the cross-model inconsistency • various inconsistency analysis results	

	Part (b) can be separately adopted for the justification of consistency between multiple items of safety assessment evidence if those items of safety assessment evidence were assessed as comparable ones in Phase one of the cross-model inconsistency analyses introduced in Chapter 5. **NB:** We omit the goals addressing the residual risk of undischarged inconsistencies for GM11.1 and GM11.2 in Part (b) of the pattern for a simplified view.
Collaborations of Part (b)	Part (b) of the model adequacy argument pattern is a component of Part (a) - the main view of post-modelling justification. The blocks below GM11.3 in Part (b) serve as the application scenarios of the cross-model inconsistency analyses, as described in Chapter 5. The justification of the adequacy of Solution 11.S1 is presented in another argument pattern – the cross-model inconsistency analysis adequacy argument pattern, as described in Section D.3. Some of the goals in Part (b) of the model adequacy argument pattern are generated on the basis of the thinking styles presented in the two-sidedness argument pattern, e.g. GM11.6 and the decompositions of GM11.1, GM11.2, GM11.4.
Consequences	There are a number of undeveloped goals after the application of Part (b) of the model adequacy argument pattern: GM11.1.3, GM11.2.3, GM11.5, and GM11.6.
Implementation	Implementation of Part (b) of the model adequacy argument pattern involves the instantiation of a series of consistency claims and referencing various inconsistency analysis results as evidence items.
Related Patterns	Safety Assessment Model Adequacy Argument Pattern - Part (a) Cross-Model Inconsistency Analysis Adequacy Argument Pattern

Structure of Part (c)


MJ-G1
Sufficient confidence exists in the adequacy of technological elements adopted in {SAMx} modelling.
MJ-C1
Technological elements: {Method of SAMx} and {Tool of SAMx}
MJ-G2
{Method of SAMx} is capable of addressing Goal {Gx} adequately.
MJ-G4
Alternative modelling methods are no more suitable than {Method of SAMx}
MJ-G3
{Tool of SAMx} adopted is capable of and qualified for {SAMx} modelling
MJ-G5
There are no more suitable alternative tools for modelling {SAMx}
MJ-C2
Alternative modelling methods that can be used for the problem domain under study by {SAMx}
MJ-C3
Alternative tools that can be used for {SAMx} modelling

Participants of Part (c)	MJ-G1	A goal that defines a confidence claim associated with the selection of a suitable modelling method and tool for safety assessment of the system under study. It is further decomposed in two directions – a line of argument justifying why the selected modelling method or tool is capable and fit for purpose, and why alternative ones are not as suitable.
	MJ-G2	A goal that indicates the capability of a modelling method. Firstly, the method should be able to generate the type of evidence needed by Goal $\{G_x\}$ in the primary safety argument. Secondly, the expressive power and processing power of the method should be adequate for modelling the system under study.
	MJ-G3	A goal that indicates the capability of a modelling tool. This goal exists if a tool is to be (or has been) adopted in $\{SAM_x\}$ modelling. The selected tool should be capable of supporting the selected modelling methods; and it should be qualified for the intended usage without itself introducing defects in modelling outputs.
	MJ-G4	A goal that indicates the unsuitability of alternative methods that can be used for the modelling task, by comparison with the selected one.

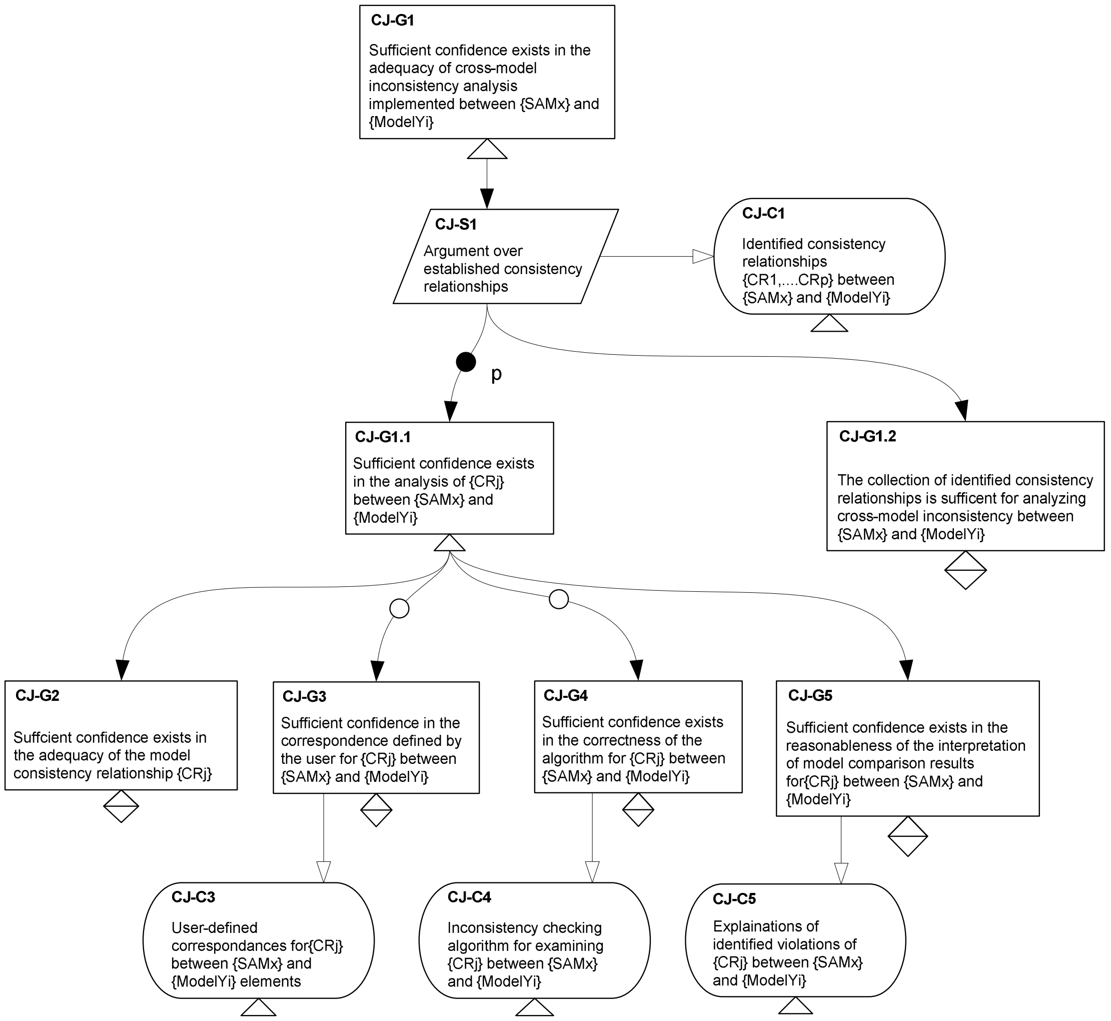
	MJ-G5	A goal that indicates the unsuitability of alternative tools that can be used for the modelling task, by comparison with the selected one. This goal exists if a tool is to be (has been) adopted in $\{SAM_x\}$ modelling.
	MJ-C1	Context of the modelling method selected and the modelling tool adopted. It can be obtained from the ‘MetaData’ of $\{SAM_x\}$ artefact.
	MJ-C2	Context of alternative modelling methods that can be used for the modelling task. It may come from domain knowledge.
	MJ-C3	Context of alternative modelling tools that can be used for the modelling task. It may come from domain experience.
Applicability of Part (c)	This part of the model adequacy argument pattern, the pre-modelling justification, is applicable to the presentation of layered claims for justifying the adequacy of selected modelling method and/or tool for a safety assessment task. Contextual information needed in Part (c) of the model adequacy argument pattern includes: • the selected method and tool of a safety assessment task • domain knowledge and experience on existing safety assessment modelling methods and tools Resources that are needed for the adequate enactment of the modelling process with a specific modelling method, such as data quality and availability and competent modellers, are not presented in Part (c) of the pattern, although they are usually considered during the process of making decisions on the choice of methods and tools adopted in a planned safety assessment process. Instead, those resource factors are considered in the post-modelling justification part of the pattern during confidence argument construction.	
Collaborations of Part (c)	Part (c) of the model adequacy argument pattern is a part of the main view of post-modelling justification, Part (a).	
Consequences of Part (c)	There are a number of undeveloped goals after the application of Part (c) of the model adequacy argument pattern: MJ-G2, MJ-G3, MJ-G4, and MJ-G5.	

Implementation	Implementation of Part (c) of the model adequacy argument pattern involves instantiation of a series of confidence claims and decision-making context.
Related Patterns	Safety Assessment Model Adequacy Argument Pattern - Part (a)

A.3 Cross-Model Inconsistency Analysis Adequacy Argument Pattern

Cross-Model Inconsistency Analysis Adequacy Argument Pattern			
Author	Linling Sun, Tim Kelly		
Created	08/07/2011 11:41:00	**Last Modified**	10/12/2012 23:06:00

Intent	This pattern provides a generic argument structure for presenting how the adequacy of cross-model inconsistency analysis as supporting evidence is considered.
Motivation	This pattern was developed: • To clarify the goals to be justified for the backing of the adequacy of a cross-model inconsistency analysis, performed on the basis of {SAM_x} and {ModelY_i}
Structure	


CJ-G1
Sufficient confidence exists in the adequacy of cross-model inconsistency analysis implemented between {SAMx} and {ModelYi}
CJ-S1
Argument over established consistency relationships
CJ-C1
Identified consistency relationships {CR1,....CRp} between {SAMx} and {ModelYi}
p
CJ-G1.1
Sufficient confidence exists in the analysis of {CRj} between {SAMx} and {ModelYi}
CJ-G1.2
The collection of identified consistency relationships is sufficent for analyzing cross-model inconsistency between {SAMx} and {ModelYi}
CJ-G2
Suffcient confidence exists in the adequacy of the model consistency relationship {CRj}
CJ-G3
Sufficient confidence in the correspondence defined by the user for {CRj} between {SAMx} and {ModelYi}
CJ-G4
Sufficient confidence exists in the correctness of the algorithm for {CRj} between {SAMx} and {ModelYi}
CJ-G5
Sufficient confidence exists in the reasonableness of the interpretation of model comparison results for{CRj} between {SAMx} and {ModelYi}
CJ-C3
User-defined correspondances for{CRj} between {SAMx} and {ModelYi} elements
CJ-C4
Inconsistency checking algorithm for examining {CRj} between {SAMx} and {ModelYi}
CJ-C5
Explainations of identified violations of {CRj} between {SAMx} and {ModelYi}

Participants	CJ-G1	A goal that defines a confidence claim on the adequacy of cross-model inconsistency analysis results as evidence for GM11.4.1 or GM11.4.2.
	CJ-S1	A strategy that explains that the decomposition is based on identified consistency relationships.
	CJ-C1	The contextual information that describes the defined model consistency relationships between $\{SAM_x\}$ and $\{ModelY_i\}$. (Model consistency relationships are the outputs of Phase 2 of the cross-model inconsistency analysis method depicted in Chapter 5.)
	CJ-G1.1	A goal that defines a confidence claim on the consistency between $\{SAM_x\}$ and $\{ModelY_i\}$ concerning the consistency relationship of $\{CR_j\}$.

	CJ-G1.2	A goal that the collection of identified consistency relationships is sufficient for representing cross-model consistency between {SAM_x} and {ModelY_i}.
	CJ-G2	A goal that defines a confidence claim on the reasonableness of model consistency relationship {CR_j} defined in the analysis.
	GJ-G3	A goal that defines a confidence claim on the correctness of user-defined correspondences between the modelling elements of {SAM_x} and the ones of {ModelY_i} for checking {CR_j}. This goal exists if {CR_j} requires user-defined correspondences between model elements as inputs to the inconsistency analysis.
	CJ-G4	A goal that defines a confidence claim on the correctness of algorithms that implement the consistency checking against specified {CR_j}. This goal exists if {CR_j} requires a checking algorithm to identify violations of the defined consistency relationship.
	CJ-G5	A goal that defines a confidence claim on the reasonableness of the explanations for the discharge of identified violations of {CR_j} as unharmful inconsistencies.
	CJ-C3	The contextual information that describes the user-defined correspondences between the modelling elements of {SAM_x} and the ones of {ModelY_i} for checking {CR_j}. (It is generated in Phase 3 of the cross-model inconsistency analysis method described in Chapter 5.)
	CJ-C4	The contextual information that describes the algorithms that implement the consistency checking against specified {CR_j}. (It is designed for some defined model consistency relationships and is used in Phase 3 of the cross-model inconsistency analysis method presented in Chapter 5.)
	CJ-C5	The contextual information that presents the explanations for identified violations of for checking {CR_j}. (It is considered and recorded in Phase 4 of the cross-model inconsistency analysis method depicted in Chapter 5.)

Applicability	This pattern is applicable to the justification of the quality of the cross-model inconsistency analysis performed according to the cross-model inconsistency analysis method defined in Chapter 5. It can be used in Phase 5 and 6 of the cross-model inconsistency analysis. Contextual information needed in this argument pattern includes: • the consistency relationships defined in the cross-model inconsistency analysis; • the user-defined correspondences between model elements for each $\{CR_j\}$, if there are any; • the inconsistency checking algorithms adopted for defined consistency relationships between two models; • the explanations of identified violations of defined model consistency relationships. This pattern can also be adopted for the evaluation of an instance of inconsistency analysis based on the inconsistency analysis method introduced in Chapter 5, even if a confidence argument was not required.
Collaborations	This inconsistency analysis adequacy argument pattern provides the generic argument structure for the further confidence argument associated with the asserted evidence relationship between 11.S1 and GM11.4.1 or GM11.4.2 in the branch view of model adequacy argument pattern - Part (b).
Consequences	There are a number of undeveloped goals after the application of this pattern: CJ-G2, CJ-G3, CJ-G4, and CJ-G5.
Implementation	Implementation of this argument pattern involves provision of a variety of contextual information.
Related Patterns	Safety Assessment Model Adequacy Argument Pattern - Part (b)

Appendix B Structured Models in EMF

This appendix presents the structured representation of three meta-models defined in the thesis. The models are represented in Eclipse Modelling Framework (EMF) [189], which is compliant with the Ecore meta-meta-models using the Emfatic text editor [70].

Section B.1 presents the safety assessment Core Data Meta-Model (CoreDMM) as defined in Chapter 3. Figure 73 shows the graphical view of CoreDMM in EMF, which is generated automatically in the modelling environment from the textual specification scripts followed.

Section B.2 presents the Evidence Model (EviM) as defined in Chapter 4. Figure 74 shows the Ecore diagram of EviM, followed by the textual specification.

Section B.3 presents the Information Model of Cross-Model Inconsistency Analysis as defined in Chapter 5. Figure 75 shows the Ecore diagram of inconsistency information model, followed by the textual specification.

B.1 CoreDMM in EMF

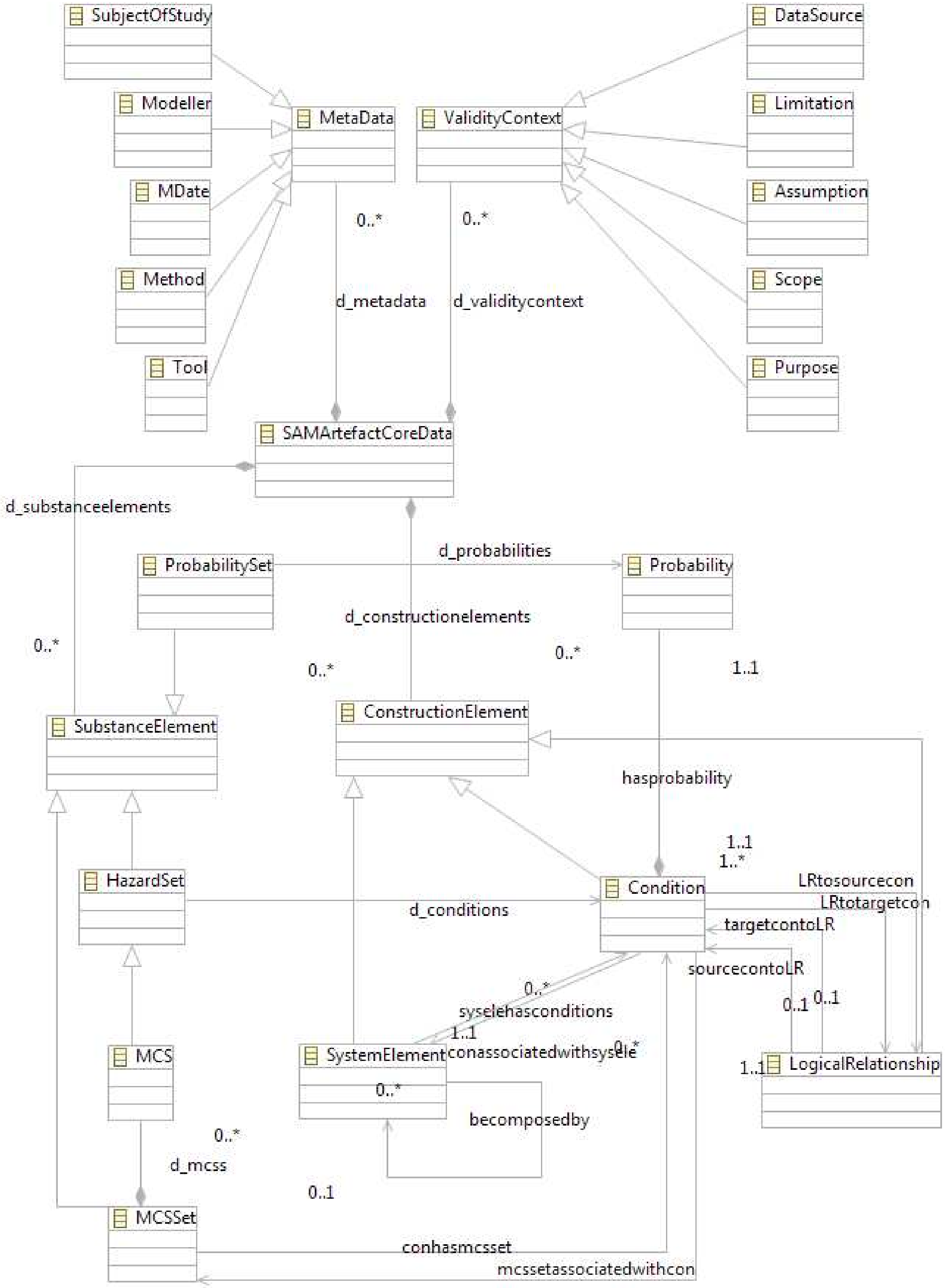


Figure 73 Safety Assessment Core Data Meta-Model in EMF

```
@namespace(uri="SAMCoreDataMM", prefix="")
package SAMCoreDataMM;

class SAMArtefactCoreData {
   val MetaData [*] d_metadata;
   val ValidityContext [*] d_validitycontext;
   val SubstanceElement [*] d_substanceelements;
   val ConstructionElement [*] d_constructionelements;
}

abstract class MetaData {
}

class SubjectOfStudy extends MetaData{
      attr String a_sosdescription;
}

class Modeller extends MetaData{
      attr String a_modellerdescription;
}

class MDate extends MetaData{
      attr String a_mdatedescription;
}

class Method extends MetaData{
      attr String a_methoddescription;
}

class Tool extends MetaData{
      attr String a_tooldescription;
}

abstract class ValidityContext {
}

class Purpose extends ValidityContext{
      attr String a_purposedescription;
}

class Scope extends ValidityContext{
      attr String a_scopedescription;
}

class Assumption extends ValidityContext{
      attr String a_assumptiondescription;
}

class Limitation extends ValidityContext{
      attr String a_limitationdescription;
}

class DataSource extends ValidityContext{
      attr String a_dsourcedescription;
}
```

```
abstract class SubstanceElement {
}

class ProbabilitySet extends SubstanceElement {
      ref Probability [*] d_probabilities;
}

class MCSSet extends SubstanceElement {
      val MCS [*] d_mcss;
      ref Condition [1] #mcssetassociatedwithcon conhasmcsset;
}

class HazardSet extends SubstanceElement{
      ref Condition [*] d_conditions;
}

class MCS extends HazardSet {
      attr String a_mcsdescription;
      attr int a_mcsorder;
}

abstract class ConstructionElement {
}

class Probability{
   attr double a_probabilityvalue;
   ref Condition [1] # hasprobability associatedwithcondition;
}

class Condition extends ConstructionElement{
      attr String a_conditiondescription;
      val Probability [1] #associatedwithcondition hasprobability;
      ref LogicalRelationship [?] #sourcecontoLR LRtosourcecon;
      ref LogicalRelationship [?] #targetcontoLR LRtotargetcon;

      ref MCSSet [?] #conhasmcsset mcssetassociatedwithcon;
      ref SystemElement [1] #conassociatedwithsysele
syselehasconditions;
}

class SystemElement extends ConstructionElement{
      attr String a_syseledescription;
      ref Condition [*] #syselehasconditions conassociatedwithsysele;
      ref SystemElement [*] #composedby becomposedby;
}

class LogicalRelationship extends ConstructionElement{
      attr String a_logicvalueANDOR;
      ref Condition [+] #LRtosourcecon sourcecontoLR;
      ref Condition [1] #LRtotargetcon targetcontoLR;
}
```

B.2 EviM in EMF

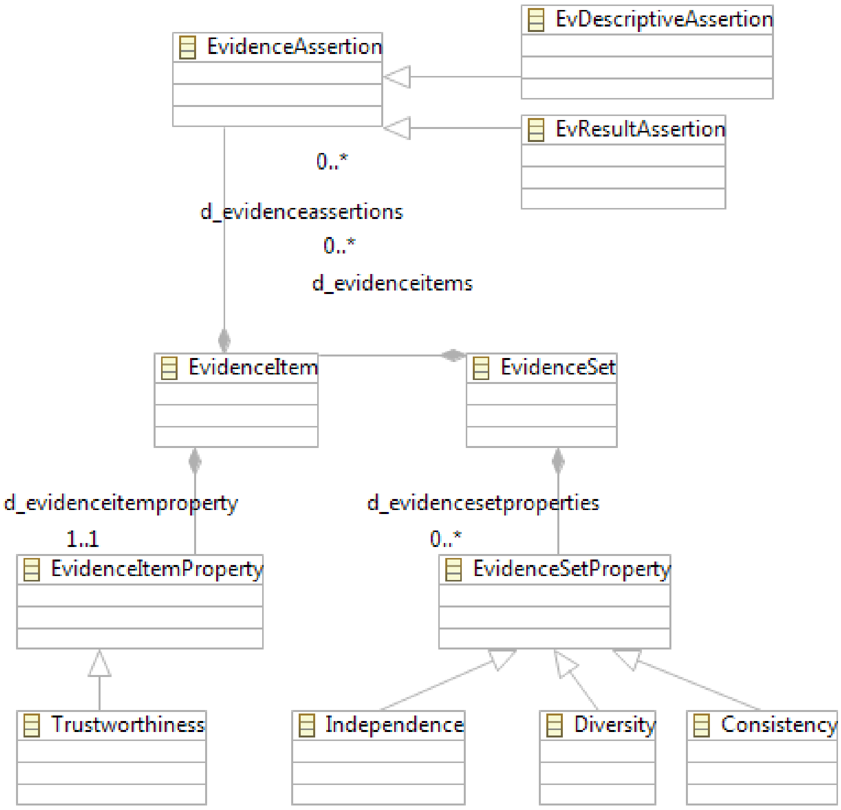


Figure 74 Model of Evidence in EMF

```
@namespace(uri="EvidenceM", prefix="")
package EvidenceM;

class EvidenceSet {
   val EvidenceItem [*] d_evidenceitems;
   val EvidenceSetProperty [*] d_evidencesetproperties;
}

class EvidenceItem {
   val EvidenceItemProperty [1] d_evidenceitemproperty;
   val EvidenceAssertion [*] d_evidenceassertions;
}

abstract class EvidenceItemProperty {
}

abstract class EvidenceAssertion {
}

class Trustworthiness extends EvidenceItemProperty{
      attr String a_trustworthinessdescription;
}
```

```
class EvResultAssertion extends EvidenceAssertion{
      attr String a_EVresultassertion;
}

class EvDescriptiveAssertion extends EvidenceAssertion{
      attr String a_EVdescriptiveassertion;
}

abstract class EvidenceSetProperty {
}

class Independence extends EvidenceSetProperty{
      attr String a_independencedescription;
}

class Diversity extends EvidenceSetProperty{
      attr String a_diversitydescription;
}
class Consistency extends EvidenceSetProperty{
      attr String a_consistencydescription;
}
```

B.3 Inconsistency Information Model in EMF

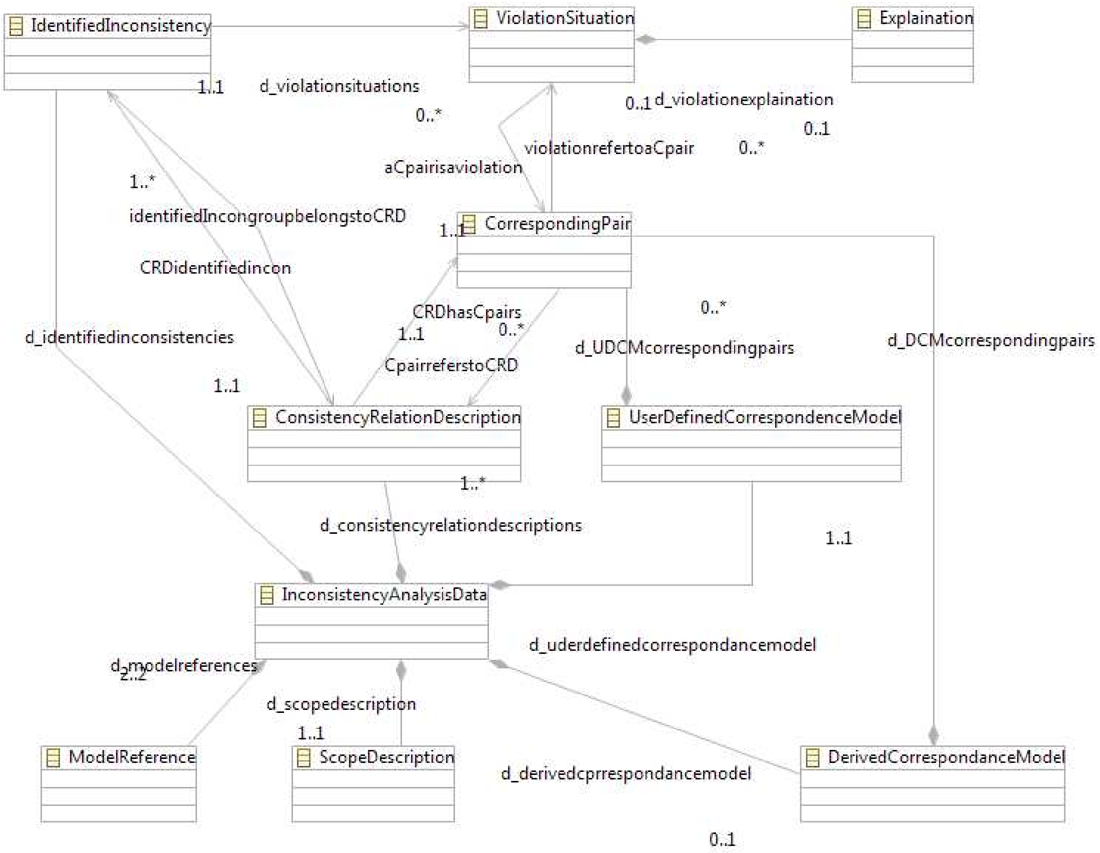


Figure 75 Information Model of Inconsistency Analysis in EMF

```
@namespace(uri="InconAnaInfoM", prefix="")
package InconAnaInfoM;

class InconsistencyAnalysisData {
   val ScopeDescription [1] d_scopedescription;
   val ModelReference [2] d_modelreferences;
   val ConsistencyRelationDescription [+]
d_consistencyrelationdescriptions;
   val UserDefinedCorrespondenceModel [1]
d_uderdefinedcorrespondancemodel;
   val DerivedCorrespondanceModel [?] d_derivedcprrespondancemodel;
   val IdentifiedInconsistency [+] d_identifiedinconsistencies;
}

class ScopeDescription {
      attr String a_scopedescription;
}

class ModelReference {
      attr String a_modelreference;
}
```

```
class ConsistencyRelationDescription {
      attr String a_consistencyrelationdescription;
      ref IdentifiedInconsistency [1] #CRDidentifiedincon
identifiedIncongroupbelongstoCRD ;
      ref CorrespondingPair [*] #CRDhasCpairs CpairreferstoCRD;
}

class UserDefinedCorrespondenceModel {
      attr String a_UDcorrespondencemodeldescription;
      val CorrespondingPair [*] d_UDCMcorrespondingpairs;
}

class DerivedCorrespondanceModel {
      attr String a_Dcorrespondencemodeldescription;
      val CorrespondingPair [*] d_DCMcorrespondingpairs;
}

class IdentifiedInconsistency {
      attr String a_UDcorrespondencemodeldescription;
      ref ViolationSituation [*] d_violationsituations;
      ref ConsistencyRelationDescription [1]
#identifiedIncongroupbelongstoCRD CRDidentifiedincon;
}

class CorrespondingPair {
      attr String a_modelAelementdescription;
      attr String a_modelBelementdescription;
      attr String ABcoresrelationdescirption;
      ref ViolationSituation [?] #aCpairisaviolation
violationrefertoaCpair;
      ref ConsistencyRelationDescription [1] #CpairreferstoCRD
CRDhasCpairs;
}

class ViolationSituation {
      ref CorrespondingPair [1] #violationrefertoaCpair
aCpairisaviolation;
      val Explaination [?] d_violationexplaination;
}

class Explaination {
      attr String a_violationexplaination;
}
```

Appendix C Case Study: Braking System Control Unit

In this section, we present a case study on the application of the inconsistency analysis method defined in Chapter 5 to investigate the inconsistency between two exemplar safety assessment models. We also evaluate the expanded argument construction process and the proposed argument patterns through the development of example arguments on the basis of the example models and the inconsistency analysis performed. A Braking System Control Unit (BSCU) of an aircraft Wheel Braking System (WBS) taken from ARP 4761[181] is chosen as the example system under study. We have two safety assessment models associated with this system at the PSSA stage (as described in Section 2.5.1).

Model I: Expanded Wheel Brake System taken from MISSA FLM handbook [131]

Coded as **OCAS.4761.WBS**.

Model II: Brake System Control Unit FTA taken from ARP 4761Appendix L [181]

Coded as **FT.4761.IB**.

The system description and model descriptions are excerpts from SAE ARP 4761 Appendix L [181] and MISSA Failure Logic Modelling Handbook [131]. The models are used for the evaluation purposes, but not exactly demonstrating the safety of a system in reality, due to the simplification and incompleteness of the information from example models.

C.1 System Description (from [181])

The primary purpose of the wheel braking system is to decelerate the aircraft on the ground without skidding the tires. The wheel braking system performs this function automatically upon landing or manually upon pilot activation. The Wheel Brake System is installed on the two main landing gears. Braking on the main gear wheels is used to provide safe retardation of the aircraft during taxiing and landing phases, and in the event of a rejected take-off. Braking on the ground is commanded either manually (via brake pedals) or automatically (autobrake) without the need for pedal application. In the NORMAL mode, the brake pedal position is electrically fed to a braking computer - the Braking System Control Unit (BSCU). This in turn produces corresponding control signals to the brakes. In addition, this computer monitors various signals which denote certain critical aircraft and system states, to provide correct brake functions and improve system fault tolerance, and generates warnings, *indications and maintenance information to other systems*. A block diagram of the WBS and the proposed BSCU architecture[18] is shown in Figure 76.

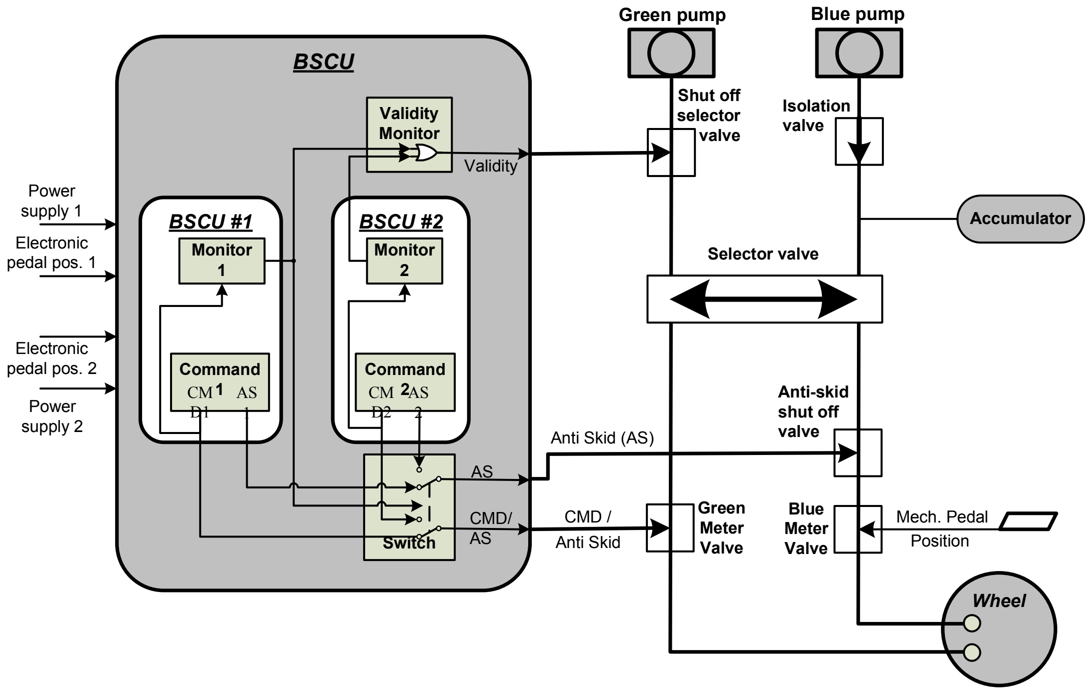


Figure 76 WBS and BSCU Architecture

The BSCU consist of two independent systems to meet the availability requirements. BSCU contain a command and monitor channel to meet the integrity requirements. Each BSCU system generates necessary voltages in its own power supply. A power supply monitor is provided to detect out of specification voltage conditions. Brake pedal inputs are provided to the command and monitor channels which compute the necessary braking commands. The commands generated by each channel are compared and if they do not agree, a failure is reported. The results of the power supply monitor and the comparator are provided to a System Validity Monitor. A failure reported by either system in a BSCU will cause that system to disable its outputs and set the System Validity Monitor to invalid. Each BSCU System Validity Monitor is provided to an overall BSCU Validity Monitor. Failure of both System 1 and System 2 will cause the selector valve to select the Alternate Brake System.

In normal operation, BSCU system 1 provides the brake and anti-skid commands to the wheel brakes. When System 1 reports a failure via its System Validity Monitor, the output of System 2, if valid, is switched in to provide the commands. In the event that System 2 subsequently fails, all BSCU outputs are disabled and the BSCU Validity Monitor is set to invalid.

[18] The figure is a merged version of Figure 3.0-1-Priliminary Wheel Brake System Diagram and Figure 3.0-1-Proposed BSCU Architecture in [181].

C.2 Model I Description (from [131])

This model[19] is developed on the basis of the prototype wheel braking system taken from ARP 4761, however, with much more details considered than the fault tree analysis of the system in the ARP example.

For the reason of the space needed, we cannot present all the codes of the model specified in AltaRica within OCAS Cecelia WorkShop[20]. The main graphical view of the model is presented in Figure 77. The embedded graphical views of the dual-BSCU design and the components of a single BSCU are presented in Figure 78.

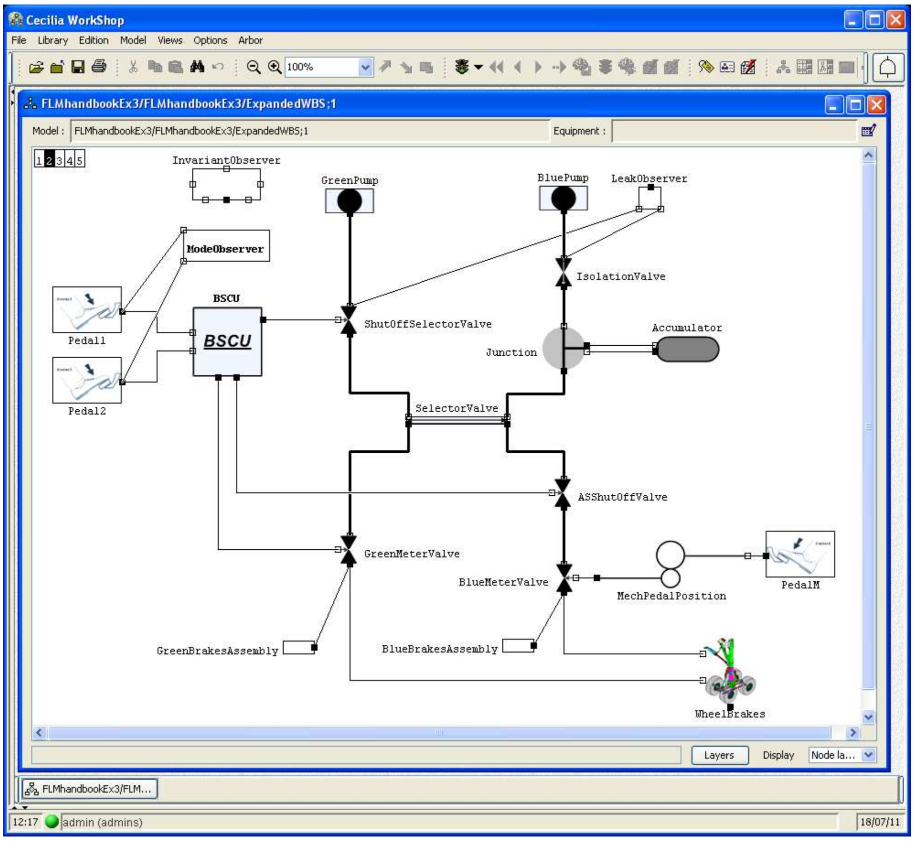


Figure 77 Main View of the WBS Model (OCAS.4761.WBS)

[19] This model is an early version of ARP WBS system generated in Period One of MISSA project. It has been revised and evolved with many versions during Period Two of the project for illustration of Failure Logic Modelling. For more comprehensive model descriptions of ARP WBS Failure Logic Modelling examples, readers should refer to ‘Failure Logic Modelling: A Pragmatic Approach’ [132].

[20] OCAS Cecelia WorkShop is a tool environment for modelling with AltaRica developed by Dassault Aviation.

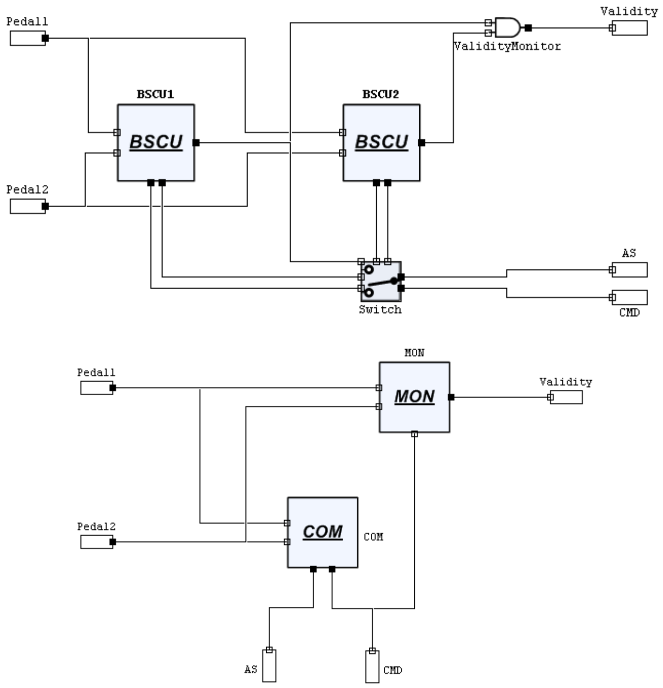


Figure 78 Independent BSCUs and the Components of a BSCU

The failure events[21] of BSCU components considered in OCAS.4761.WBS are presented in Table 15. Besides that, the model of OCAS.4761.WBS has also taken into account 28 common cause events.

Model Component	Failure Events
Command (COM)	*ASprocessStuck*
	ASprocessTerminated
	ProcessorError
	CMDprocessStuck
	CMDprocessTerminated
Monitor (MON)	*ProcessStuck*
	ProcessTerminated
Switch	*StuckIn1*
	SpontaneousTrip
	CircuitFailure
Validity Monitor	*ProcessStuck*
	ProcessTerminated

Table 15 Failure Events of Components in OCAS.4761.WBS

The definitions (in AltaRica) of the failure conditions examined in the qualitative analysis of OCAS.4761.WBS are presented in Table 16.

[21] The ‘failure event’ is an AltaRica concept that means the events that can trigger state changes of a component.

Failure Condition ID	Failure Condition Specification	Formalisation in terms of WBS FLM failure modes in AltaRica
FC-WBS1	*Total loss of braking capabilities*	*(Mode = **Normal** &* *GrMtrValve.HydOut.FWD.**PressureOmission** &* *~BlMtrValve.HydOut.FWD.**UnnecessaryPressure** &* *~BlMtrValve.HydOut.FWD.**TooLowPressure** &* *~BlMtrValve.HydOut.FWD.**TooHighPressure**) |* *(Mode = **Alternate** &* *BlMtrValve.HydOut.FWD.**PressureOmission** &* *~GrMtrValve.HydOut.FWD.**UnnecessaryPressure** &* *~GrMtrValve.HydOut.FWD.**TooLowPressure** &* *~GrMtrValve.HydOut.FWD.**TooHighPressure**)*
FC-WBS2	*Inadvertent application of brakes*	*GrMtrValve.HydOut.FWD.**PressureCommission** |* *BlMtrValve.HydOut.FWD.**PressureCommission***

Table 16 WBS Failure Condition Definitions in OCAS.4761.WBS

The excerpt of some qualitative analysis results of MCSs associated with FC-WBS2 is presented in Figure 79.

```
/*
orders(MCS('WheelBrakes.FailureCondition.InadvertentBraking')) =
orders      product-number
1     2
2     68
3     174
total 244
end
*/
products(MCS('WheelBrakes.FailureCondition.InadvertentBraking'))
=
{'GreenMeterValve.JamOpen'}
{'GreenMeterValve.SpringFailure'}
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.BSCU1.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.Switch.StuckIn1'}
{'BSCU.BSCU1.MON.ProcessStuck', 'Pedal1.DemandCommiss'}
{'BSCU.BSCU1.MON.ProcessStuck', 'Pedal2.DemandCommiss'}
{'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.Switch.SpontaneousTrip'}
{'BSCU.BSCU2.MON.ProcessStuck', 'Pedal1.DemandCommiss'}
{'BSCU.BSCU2.MON.ProcessStuck', 'Pedal2.DemandCommiss'}
{'BSCU.ValidityMonitor.ProcessStuck', 'Pedal1.DemandCommiss'}
{'BSCU.ValidityMonitor.ProcessStuck', 'Pedal2.DemandCommiss'}
{'BSCU.ValidityMonitor.ProcessTerminated',
'BlueMeterValve.JamOpen'}
{'BSCU.ValidityMonitor.ProcessTerminated',
'BlueMeterValve.SpringFailure'}
{'BSCU.ValidityMonitor.ProcessTerminated',
'MechPedalPosition.Jam'}
{'BSCU.ValidityMonitor.ProcessTerminated',
'PedalM.DemandCommiss'}
{'BlueMeterValve.JamOpen', 'GreenBrakesAssembly.HydOut_BWD_Leak'}
{'BlueMeterValve.JamOpen', 'GreenMeterValve.Rupture'}
{'BlueMeterValve.JamOpen', 'GreenPump.MechanicalFailure'}
{'BlueMeterValve.JamOpen', 'GreenPump.Rupture'}
{'BlueMeterValve.JamOpen', 'Pedal1.DemandCommiss'}
{'BlueMeterValve.JamOpen', 'Pedal1.DemandLow'}
{'BlueMeterValve.JamOpen', 'Pedal1.DemandOmiss'}
{'BlueMeterValve.JamOpen', 'Pedal2.DemandCommiss'}
{'BlueMeterValve.JamOpen', 'Pedal2.DemandLow'}
{'BlueMeterValve.JamOpen', 'Pedal2.DemandOmiss'}
{'BlueMeterValve.JamOpen', 'SelectorValve.LockFailure'}
{'BlueMeterValve.JamOpen', 'ShutOffSelectorValve.Rupture'}
{'BlueMeterValve.JamOpen',
'ShutOffSelectorValve.UnlocksSpontaneously'}
{'BlueMeterValve.SpringFailure',
'GreenBrakesAssembly.HydOut_BWD_Leak'}
{'BlueMeterValve.SpringFailure', 'GreenMeterValve.Rupture'}
{'BlueMeterValve.SpringFailure', 'GreenPump.MechanicalFailure'}
{'BlueMeterValve.SpringFailure', 'GreenPump.Rupture'}
{'BlueMeterValve.SpringFailure', 'Pedal1.DemandCommiss'}
{'BlueMeterValve.SpringFailure', 'Pedal1.DemandLow'}
{'BlueMeterValve.SpringFailure', 'Pedal1.DemandOmiss'}
```

Figure 79 Excerpt of MCS Results of FC-WBS2 in OCAS.4761.WBS

C.3 Model II Description

A fault tree, FT.4761.IB, is constructed based on the information from Appendix L of ARP 4761 [181]. The top event is "*BSCU commands braking in absence of braking input and causes inadvertent braking*". The three parts of Figure 4.2.1-2 (PSSA BSCU – FTA) in ARP 4761 is merged together as a single fault tree FT.4761.IB, as shown by the tree structure presented in Figure 80. The detailed description of the event labels is presented in Table 17-1 and Table 17-2. The qualitative analysis of the fault tree is performed in OpenFTA [11], an open source tool for fault tree analysis, as a single fault tree under qualitative analysis.

Figure 81 presents the graphical view of the BSCU fault tree in OpenFTA and the MCS analysis results obtained from OpenFTA. The graphical view in Figure 81 is exactly same as the tree shown in Figure 80.

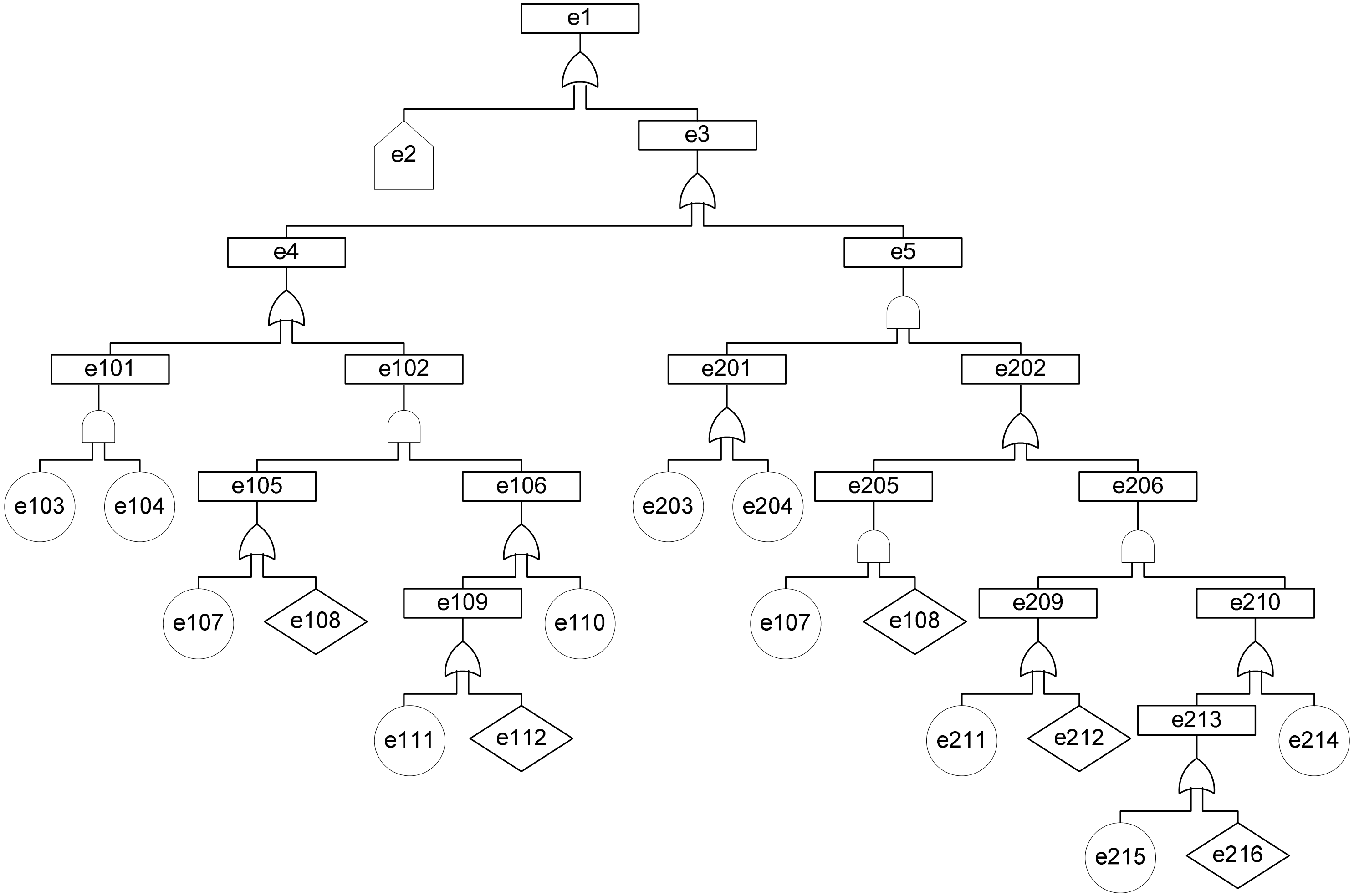


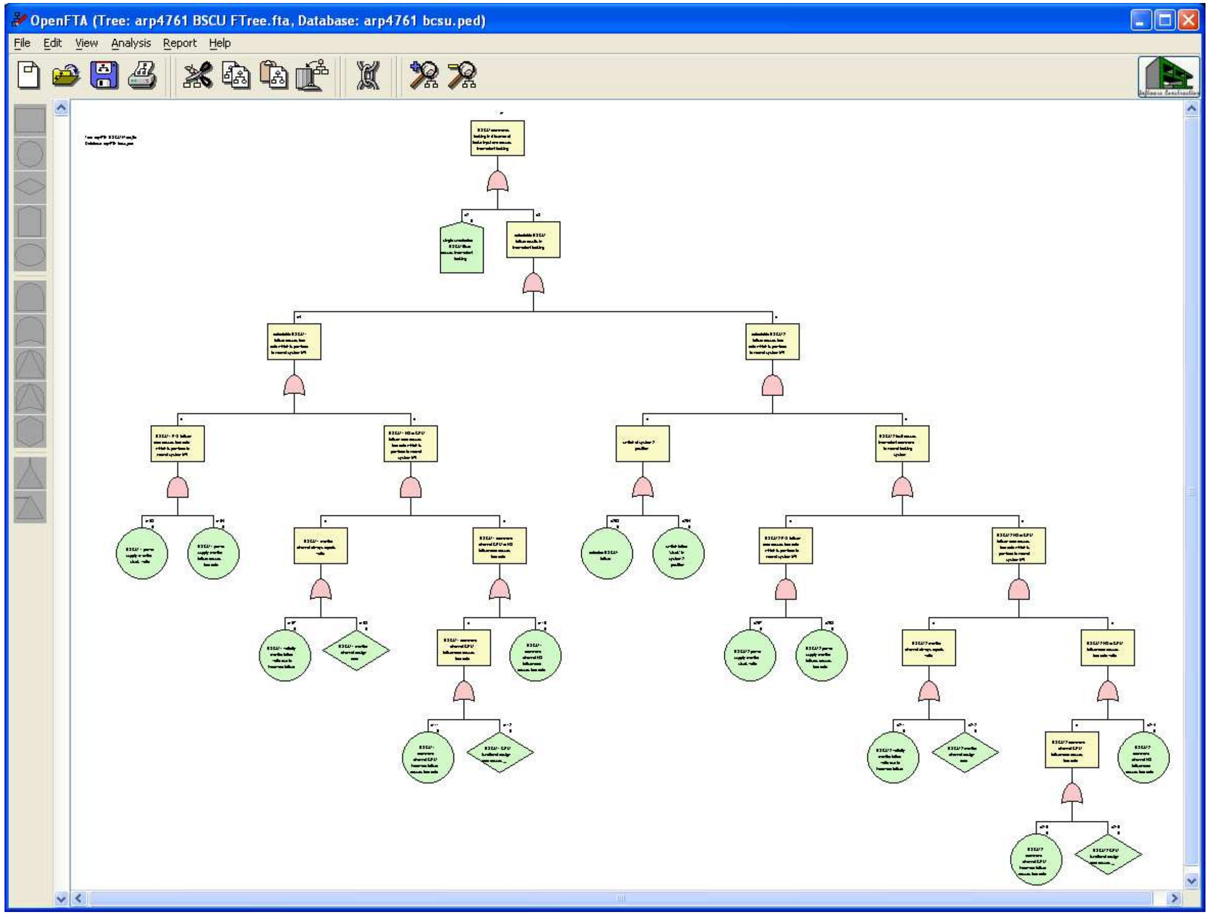
Figure 80 Graphical View of BSCU Fault Tree Structure (FT.4761.IB)

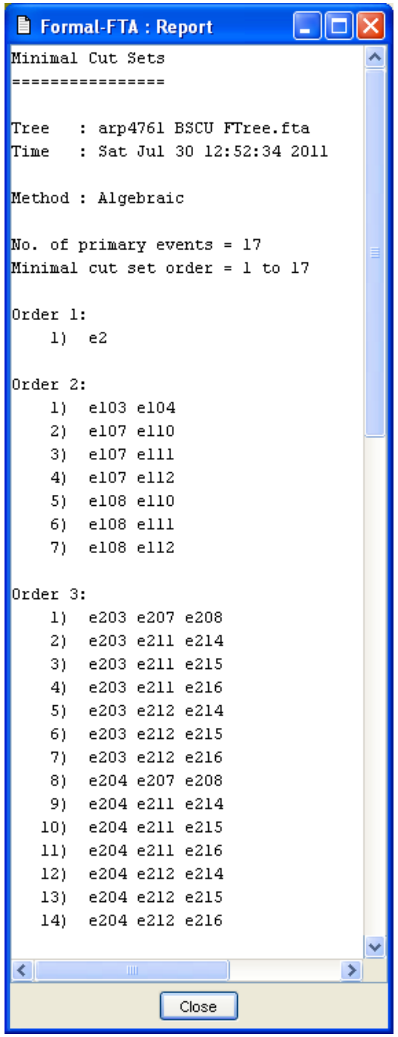
ID	Description (from ARP 4761 [181])	Notes
e1	*BSCU commands braking in absence of braking input and causes inadvertent braking*	Top event
e2	*Single undetected BSCU failure causes inadvertent braking*	External event Probability 0.
e4	*Detectable BSCU1 failure causes bad data which is provided to normal system MV*	Intermediate event
e5	*Detectable BSCU2 failure causes bad data which is provided to normal system MV*	Intermediate event
e101	*BSCU1 P/S failure/error causes bad data which is provided to normal system MV*	Intermediate event
e102	*BSCU1 I/O or CPU failure/error causes bad data which is provided to normal system MV*	Intermediate event
e103	*BSCU1 power supply monitor stuck valid*	
e104	*BSCU1 power supply failure causes bad data*	
e105	*BSCU1 monitor channel always reports valid*	Intermediate event
e106	*BSCU1 command channel I/O or CPU failure/error causes bad data*	Intermediate event
e107	*BSCU1 validity monitor failed valid due to hardware failure*	
e108	*BCSU1 monitor channel design error*	Undeveloped event Probability 0.
e109	*BSCU1 command channel CPU failure/error causes bad data*	Intermediate event
e110	*BSCU1 command channel I/O failure causes bad data*	
e111	*BSCU1 command channel CPU hardware failure causes bad data*	
e112	*BSCU1 CPU function design error causes bad data*	Undeveloped event Probability 0.
e201	*Switch at system 2 position*	Intermediate event
e202	*BSCU2 fault causes inadvertent command to normal braking system*	Intermediate event
e203	*Detected BSCU1 failure* *NB: this is not exactly a basic event – it is a intermediate event composed of e4 and the detection success.*	Can be developed as the sub tree of e4 if the failure rate of detecting BSCU1 failure is assumed to be 0.
e204	*Switch failed 'stuck' in system 2 position*	

Table 17-1 Event Description of FT.4761.IB

ID	Description (from ARP 4761 [181])	Notes
e205	*BSCU2 P/S failure/error causes bad data which is provided to normal system MV*	Intermediate event
e206	*BSCU2 I/O or CPU failure/error causes bad data which is provided to normal system MV*	Intermediate event
e207	*BSCU2 power supply monitor stuck valid*	
e208	*BSCU2 power supply failure causes bad data*	
e209	*BSCU2 monitor channel always reports valid*	Intermediate event
e210	*BSCU2 I/O or CPU failure/error causes bad data*	Intermediate event
e211	*BSCU2 validity monitor failed valid due to hardware failure*	
e212	*BCSU2 monitor channel design error*	Undeveloped event Probability 0.
e213	*BSCU2 CPU failure causes bad data*	Intermediate event
e214	*BSCU2 I/O failure causes bad data*	
e215	*BSCU2 command channel CPU hardware failure causes bad data*	
e216	*BSCU2 CPU function design error causes bad data*	Undeveloped event Probability 0.
Note: Intermediate events are not used in model comparison. They are not involved in the model elements correspondence definition in Phase 3 of the analysis.		

Table 17-2 Event description of FT.4761.IB


OpenFTA (Tree: arp4761 BSCU FTree.fta, Database: arp4761 bcsu.ped)
File
Edit
View
Analysis
Report
Help



```
Minimal Cut Sets
================

Tree   : arp4761 BSCU FTree.fta
Time   : Sat Jul 30 12:52:34 2011

Method : Algebraic

No. of primary events = 17
Minimal cut set order = 1 to 17

Order 1:
     1)  e2

Order 2:
     1)  e103 e104
     2)  e107 e110
     3)  e107 e111
     4)  e107 e112
     5)  e108 e110
     6)  e108 e111
     7)  e108 e112

Order 3:
     1)  e203 e207 e208
     2)  e203 e211 e214
     3)  e203 e211 e215
     4)  e203 e211 e216
     5)  e203 e212 e214
     6)  e203 e212 e215
     7)  e203 e212 e216
     8)  e204 e207 e208
     9)  e204 e211 e214
    10)  e204 e211 e215
    11)  e204 e211 e216
    12)  e204 e212 e214
    13)  e204 e212 e215
    14)  e204 e212 e216
```

Figure 81 BSCU Fault Tree (FT.4761.IB) in OpenFTA and MCS Result

C.4 Inconsistency Analysis - Model I & Model II

This section presents the key results of each phase of the inconsistency analysis between Model I and Model II.

C.4.1 Phase 1

The two models selected differ considerably in terms of the scope of the components and the failure modes involved in the modelling. The OCAS.4761.WBS contains much more information than the FT.4761.IB. It took some time to clarify the range of modelling elements considered by each model. The knowledge and understanding of the model under study is very important for this phase. The *ModelReference* and *ScopeDescription* defined in this phase are presented in Table 18.

ModelReference	Ma: an AltaRica model **- OCAS.4761.WBS**
	Mb: a traditional fault tree model **- FT.4761.IB**
ScopeDescription	The shared system elements modelled include: BSCU1, BSCU2, BSCU1 Monitor, BSCU1 Command, BSCU2 Monitor, BSCU2 Command, (overall) Validity Monitor, Switch. The overlapping concern is 'Inadvertent braking caused by BSCU in absence of brake inputs', coded as **FC.casestudy**. The shared type of substance results is: a set of MCSs. The common construction elements include: conditions. A condition means a 'failure event' shown in M_a and means a 'basic event' in M_b.

Table 18 Phase One Output of the Case Study

C.4.2 Phase 2

Consistency relationships defined for the Case study are presented in Table 19.

CaCR1: Each 'condition' shown in M_b has a corresponding 'condition' in M_a.

CaCR2: Each MCS in M_b associated with the top event has a corresponding MCS in M_a that contributes to the overlapping concern (**FC.casestudy**).

Table 19 Consistency Relationships Defined in the Case Study

CaCR1: This consistency relationship aims to check whether the conditions considered in the traditional fault tree have been considered in the AltaRica model. Because the description of basic events of FT.4761.IB is not in detail, we do not expect the conditions in two models correspond in a strict sense (may not represent the exact same conditions in reality).

CaCR2: This consistency relationship aims to check whether the condition combinations of MCS considered in the traditional fault tree can lead to the concerned condition of **FC.casestudy** in the AltaRica model. To investigate this consistency relationship, it is required to set up relations between corresponding conditions in two models and to examine the associated correspondences between MCSs in two models.

C.4.3 Phase 3

FT.4761.IB is constructed manually and described entirely in natural language. We manually process the model information and try to build up the correspondences between conditions of the two models gradually. Table 20 presents the initial corresponding results established based on the system components being modelled.

Through setting up the initial correspondences, we found that:

- It is *not* always possible to have 'equivalent-to' relationships between modelled conditions in two safety assessment models, especially when conditions are described with different modelling constructs and with different levels of details. In this case, human judgements on whether one condition was considered in another model are crucial for determining the violations.

For example, *BSCU2.command* has 5 failure events considered in OCAS.4761.WBS, whereas only 3 basic events considered in FT.4761.IB. However, it is found that all three basic events in FT.4761.IB describe the *bad data* failure mode of *BSCU2.command* caused by three lower level causes (namely by power supply 2 failure, I/O failure, and CPU failure). It is also found that only one of the 5 failure events (`CMDprocessStuck`) may lead to the inadvertent braking output by *BSCU2.command* in OCAS.4761.WBS. But it is unclear concerning the relationships of *bad data* of *BSCU2.command* in FT.4761.IB and the failure event and the output flow of *BSCU2.command* in OCAS.4761.WBS.

System component	Failure event in Ma OCAS.4761.WBS	Basic event in Mb FT.4761.IB
BSCU1.command	*ASprocessStuck* *ASprocessTerminated* *ProcessorError* *CMDprocessStuck* *CMDprocessTerminated*	e104 (command bad data by power supply 1) e110 (command bad data by I/O) e111(command bad data by CPU)
BSCU1.monitor	*ProcessStuck* *ProcessTerminated*	e107 (failed valid)
BSCU2.command	*ASprocessStuck* *ASprocessTerminated* *ProcessorError* *CMDprocessStuck* *CMDprocessTerminated*	e208 (command bad data by power supply 2) e214 (command bad data by I/O) e215(command bad data by CPU)
BSCU2.monitor	*ProcessStuck* *ProcessTerminated*	e211 (failed valid)
(Overall) Validity monitor	*ProcessStuck* *ProcessTerminated*	N/A
Switch	*StuckIn1* *SpontaneousTrip* *CircuitFailure*	e204(failed stuck at position 2)
BSCU1.PSmonitor (power supply monitor)	N/A	e103 (BSCU1 power supply monitor failed valid)
BSCU2.PSmonitor (power supply monitor)	N/A	e207 (BSCU2 power supply monitor failed valid)
(Detector)[22]	N/A	e203 (detected BSCU1 failure)
N/A: The system element associated with a 'condition' in one model is not modelled in other model used in the cross-model inconsistency analysis.		

Table 20 Initial Relations between Conditions of Two Models

Another issue is that not all failure events of system components modelled in OCAS.4761.WBS are shown in the MCS output of **FC.casestudy**. It is futile to examine those failure events modelled in OCAS.4761.WBS if they are irrelevant to **FC.casestudy** of our concern. In addition, we assume all the failure events associated with system elements modelled by OCAS.4761.WBS, but not by FT.4761.IB, with probability of 0. In this way, we cut down the scope of failure events (in OCAS.4761.WBS) under study in this inconsistency

[22] The behaviour of this system component is used in FT.4761.IB. However, it is not depicted as a part of the system architecture provided in ARP 4761.

analysis. The failure events modelled in OCAS.4761.WBS but not shown in focused MCS outputs (that are with members of pure failure events of BSCU components)[23] are omitted for further inconsistency analysis steps. Then we can shrink the failure events in OCAS.4761.WBS that need to consider correspondence with the basic events in FT.4761.IB (as shown in Table 21). The correspondence presented in Table 21 is not precise 'equivalences', but approximate matching relations. The 'conditions' in two models do not correspond with one-to-one relationship. We have adopted intermediate codes to represent corresponding conditions in two models for the implementation of the comparison of MCS for CaCR2.

System component	Failure event in M_a OCAS.4761.WBS	X_i /I_i	Basic event in M_b FT.4761.IB
BSCU1.command		X1	e104 (command bad data by power supply 1)
	ASprocessStuck- *ASprocessTerminated-* *ProcessorError* *CMDprocessStuck* *CMDprocessTerminated*	I1	e110 (command bad data by I/O) e111(command bad data by CPU)
BSCU1.monitor	*ProcessStuck*	X2	
	ProcessTerminated	I2	e107 (failed valid)
BSCU2.command		X3	e208 (command bad data by power supply 2)
	CMDprocessStuck	I3	e214 (command bad data by I/O) e215(command bad data by CPU)
BSCU2.monitor	*ProcessStuck*	X4	
	(ProcessTerminated)∗	X5	e211 (failed valid)
(Overall) Validity monitor	*ProcessStuck*	X6	N/A
Switch		X7	e204(failed stuck at position 2)
	StuckIn1	X8	
	SpontaneousTrip	X9	
BSCU1.PSmonitor (power supply monitor)	N/A	X10	e103 (BSCU1 power supply monitor failed valid)

[23] The failure events omitted in the following study are shown as members in other MCSs that contain member failure events that do not belong to BSCU, e.g. failure events of the Pedal, the Pumps, or the Valves.

<table>
<tr><td>BSCU2.PSmonitor (power supply monitor)</td><td>N/A</td><td>X11</td><td>e207 (BSCU2 power supply monitor failed valid)</td></tr>
<tr><td>(Detector)[24]</td><td>N/A</td><td>X12</td><td>e203 (detected BSCU1 failure)</td></tr>
<tr><td colspan="4">N/A: The system element associated with a ‘condition’ in one model is not modelled in other model used in the cross-model inconsistency analysis.
X_i: A temporary code of ‘conditions’ that do not have a corresponding ‘condition’ in the other model under analysis.
I_i: A temporary code of ‘conditions’ that are viewed as a generic ‘condition’, representing groups of corresponding ‘conditions’ in models under analysis. It performs as a media to substitute members of MCSs in order to implement the comparison of MCSs in the later step.
* It is modelled as a model element in OCAS.4761.WBS, but not shown in the MCSs generated for <b>FC.casestudy</b>.
<b>NB</b>: e107, e211 do not correspond to <i>ProcessStuck</i> because <i>ProcessStuck</i> only leads to <i>StuckNegative output</i> of BSCU monitors according to the specification of the model element of Monitor in OCAS.4761.WBS.</td></tr>
</table>

Table 21 Refined Relations between Conditions of Two Models

Based on the refined relations presented in Table 21, we identify **six** violations for CaCR1 after the comparison of conditions within the scope of inconsistency analysis.

- **V1-1**: The e104 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;
- **V1-2**: The e208 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;
- **V1-3**: The e103 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;
- **V1-4**: The e207 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;

[24] The behaviour of this system component is used in FT.4761.IB. However, it is not depicted as a part of the system architecture provided in ARP 4761.

- **V1-5**: The e204 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;

- **V1-6**: The e203 in FT.4761.IB does not have a corresponding condition in OCAS.4761.WBS;

In order to examine **CaCR2**, we need to take into account of conditions in FT.4761.IB and OCAS.4761.WBS that do not have corresponding part in the other.

- e203 is assumed as a true event in OCAS.4761.WBS. It is indicated by the failure output flows of *BSCU1.monitor* that are defined as *false negative* or *false positive.* In the model of OCAS.4761.WBS, BSCU1 is supposed to be able to have its failure detected.

- e103 and e104 are not modelled within BSCU1 in OCAS.4761.WBS. The power supply monitors of both BSCUs are not modelled in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU1 does not fail in mission.

- e207 and e208 are not modelled within BSCU2 in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU2 does not fail in mission.

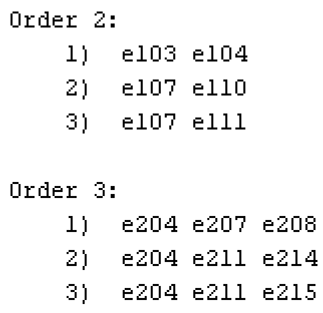
- e204 does not have a corresponding condition in OCAS.4761.WBS; any MCS in FT.4761.IB results with e204 will not have a corresponding MCS in OCAS.4761.WBS.

In order to examine CaCR2, we also need to filter out MCSs in OCAS.4761.WBS with system elements out of scope of the cross-model inconsistency analysis. After the filtering, we shrink the set of MCSs to a smaller range for the comparison purpose. The fifteen MCSs, as illustrated in Figure 82, from the total 244 MCSs (generated within order 3 for FC.casestudy as shown in Figure 79) are composed of conditions associated with BSCU components only.

```
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.BSCU1.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.Switch.StuckIn1'}
{'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.Switch.SpontaneousTrip'}
{'BSCU.BSCU1.COM.ASprocessStuck',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.ASprocessStuck',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
{'BSCU.BSCU1.COM.ASprocessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.ASprocessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessStuck',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessStuck',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.CMDprocessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
{'BSCU.BSCU1.COM.ProcessorError',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.COM.ProcessorError',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
{'BSCU.BSCU1.MON.ProcessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}
{'BSCU.BSCU1.MON.ProcessTerminated',
'BSCU.BSCU2.COM.CMDprocessStuck',
'BSCU.ValidityMonitor.ProcessStuck'}
```

Figure 82 MCSs in OCAS.4761.WBS for Comparison

If the undeveloped and external events which are assumed with a probability of 0 are removed from the tree, and if the failure rate of detecting BSCU1 failure is assumed to be 0 (to allow e4 sub tree to be the developed tree of e203), the MCS analysis results of FT.4761.IB is simplified and shown as below (in Figure 83).



```
Order 2:
    1)  e103 e104
    2)  e107 e110
    3)  e107 e111

Order 3:
    1)  e204 e207 e208
    2)  e204 e211 e214
    3)  e204 e211 e215
```

Figure 83 MCSs of Simplified FT.4761.IB

Each individual MCS in the simplified set of MCSs of FT.4761.IB is transformed into a form that is represented with the intermediate codes assigned in Table 21. The transformed MCSbi for **CaCR2** inconsistency analysis is presented in Table 22.

MCSs in M_b FT.4761.IB	MCSbi	Comments
{e103, e104}	{X10, X1}	Due to power supply monitor is not modelled in OCAS.4761.WBS, this MCS in M_b won't have corresponding MCS in M_a
{e107, e110}	{I1, I2}	`BSCU1.MON.ProcessTerminated,` as the corresponding failure event of e107 only shown in MCSs of OCAS.4761.WBS that are with a cardinality higher than 2.
{e107, e111}	{I1, I2}	
{e204, e207, e208}	{X7,X11, X3}	None of the events have been modelled in OCAS.4761.WBS.
{e204, e211, e214}	{ X7, X5,I3}	Due to the power supply monitor is not modelled in OCAS.4761.WBS, this MCS in M_b won't have corresponding MCS in M_a. However, e211, which is modelled in OCAS.4761.WBS, is not shown in any relevant MCSs of OCAS.4761.WBS under inconsistency analysis.
{e204, e211, e215}	{ X7, X5,I3}	

Table 22 FT.4761.IB MCS Transformation

Each individual MCS in the simplified set of MCSs of OCAS.4761.WBS is transformed into a form that is represented with the intermediate codes assigned in Table 21. The transformed MCSai for **CaCR2** inconsistency analysis is presented in Table 23. The manual comparison results of MCSai and MCSbi are listed in the final column of Table 23.

MCSs in M_a OCAS.4761.WBS	MCSai	A member of MCSbi (Y/N)
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.BSCU1.MON.ProcessStuck'}	{I1, X2}	N
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.Switch.StuckIn1'}	{I1, X8}	N
{'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.Switch.SpontaneousTrip'}	{I3, X9}	N
{'BSCU.BSCU1.COM.ASprocessStuck', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.COM.ASprocessStuck', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N
{'BSCU.BSCU1.COM.ASprocessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.COM.ASprocessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.COM.CMDprocessStuck', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N
{'BSCU.BSCU1.COM.CMDprocessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.COM.CMDprocessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N
{'BSCU.BSCU1.COM.ProcessorError', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.COM.ProcessorError', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N
{'BSCU.BSCU1.MON.ProcessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.BSCU2.MON.ProcessStuck'}	{I1, I3, X4}	N
{'BSCU.BSCU1.MON.ProcessTerminated', 'BSCU.BSCU2.COM.CMDprocessStuck', 'BSCU.ValidityMonitor.ProcessStuck'}	{I1, I3, X6}	N

Table 23 OCAS.4761.WBS MCS Transformation

After examining Table 22 and Table 23, we found the following **six** violations for **CaCR2**:

- **V2-1**: {e204, e211, e214} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

- **V2-2**: {e204, e211, e215} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

- **V2-3**: {e103, e104} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

- **V2-4**: {e107, e110} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

- **V2-5**: {e107, e111} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

- **V2-6**: {e204, e207, e208} does not have an approximate correspondence MCS in OCAS.4761.WBS analysis results.

C.4.4 Phase 4

Table 24 presents the potential explanations to the identified violations and states whether the identified violations are marked out as model inconsistencies.

Due to the incompleteness of the information from FT.4761.IB, we cannot cross-check the coverage of conditions addressed by OCAS.4761.WBS. The inconsistencies identified are limited to the scope of analysis. In the converse direction, however, according to OCAS.4761.WBS (which is constructed with the help from domain engineers), we can see that the FT.4761.IB is very simplified and incomplete version of a traditional fault tree[25]. For example, the multiple failure modes of the switch component of the BSCU should be considered in FT.4761.IB.

[25] As the editor notes in ARP 4761 stated, the fault tree example is for illustration of the analysis technique and the analysis process only.

Violation ID	Explanation	Inconsistency (Y/N)
V1-1	e104 are not modelled within BSCU1 in OCAS.4761.WBS. The power supply monitors of both BSCUs are not modelled in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU1 does not fail in mission.	N
V1-2	e208 is not modelled within BSCU2 in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU2 does not fail in mission.	N
V1-3	e103 are not modelled within BSCU1 in OCAS.4761.WBS. The power supply monitors of both BSCUs are not modelled in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU1 does not fail in mission.	N
V1-4	e207 is not modelled within BSCU2 in OCAS.4761.WBS. It is assumed that the power supply monitor of BSCU2 does not fail in mission.	N
V1-5	e204 does not have a corresponding condition in OCAS.4761.WBS; e204 is a condition that should be considered in OCAS.4761.WBS.	Y
V1-6	e203 is assumed as a true event in OCAS.4761.WBS	N
V2-1	Due to the Omission of e211 in related MCSs of OCAS.4761.WBS	Y
V2-2	Due to the Omission of e211 in related MCSs of OCAS.4761.WBS	Y
V2-3	Due to the assumption of non-occurrence of e103 and e104 in OCAS.4761.WBS	N
V2-4	Due to the Omission of {e107, e110} in related MCSs of OCAS.4761.WBS	Y
V2-5	Due to the Omission of {e107, e111} in related MCSs of OCAS.4761.WBS	Y
V2-6	Due to the omission of e204 and the assumption of non-occurrence of e207 and 208 in OCAS.4761.WBS	Y

Table 24 Explanation of Identified Violations

C.4.5 Phase 5

Constrained by the limitations of the models used in this case study, the quality of this specific inconsistency analysis cannot be reasonably evaluated. But as we described in Chapter 5, the four aspects of the analysis should be carefully examined for the ‘reasonableness’ of the analysis results. We skip this step in the case study.

But as the description of Phase 3 indicated, many subtle judgements are made during the implementation of the model comparison. It is desirable to have these judgements recorded for the evaluation of the inconsistency analysis at Phase 5.

C.4.6 Phase 6

This phase will be illustrated in combination of the case study for the model adequacy argument construction in Section C.5.

C.5 Model Adequacy Argument Construction

With respect to the evaluation of the expanded Six-Step Method, we looked at the process of developing an argument to the point at which the BSCU safety assessment results (used for evaluating the inconsistency analysis part) were addressed as evidence items. In this section, we first 'rerun' the primary safety argument development process to illustrate the consideration of the expanded argument construction process. Then we present an example argument for justifying the adequacy of citing OCAS.4761.WBS as an item of evidence in the primary safety argument.

C.5.1 Primary Safety Argument Case Study

Figure 84 depicts a simplified version of the primary safety argument for demonstrating the safety of Aircraft S18 according to the aircraft and system descriptions in ARP 4761. The (positive) structure of a primary safety argument outlined in Figure 84 is compliant to ARP 4754 safety assessment process (it is adapted from an early version of a generic primary argument structure presented in MISSA).

As Figure 84 illustrated, the shaded goals presents safety claims that are formulated based on the following safety requirements elicited in ARP 4761 L [181].

- One of the inputs to WBS PSSA provided by WBS FHA is: "Inadvertent wheel braking of all wheels during takeoff roll after V1 is a Catastrophic failure condition; it shall be less than 5E-9 per flight".

- One of the BSCU safety requirements defined in ARP 4761 example is "*No single failure of the BSCU shall lead to **inadvertent braking***". The planned safety analyses for this safety requirement include "*CMA and FMEA as necessary*".

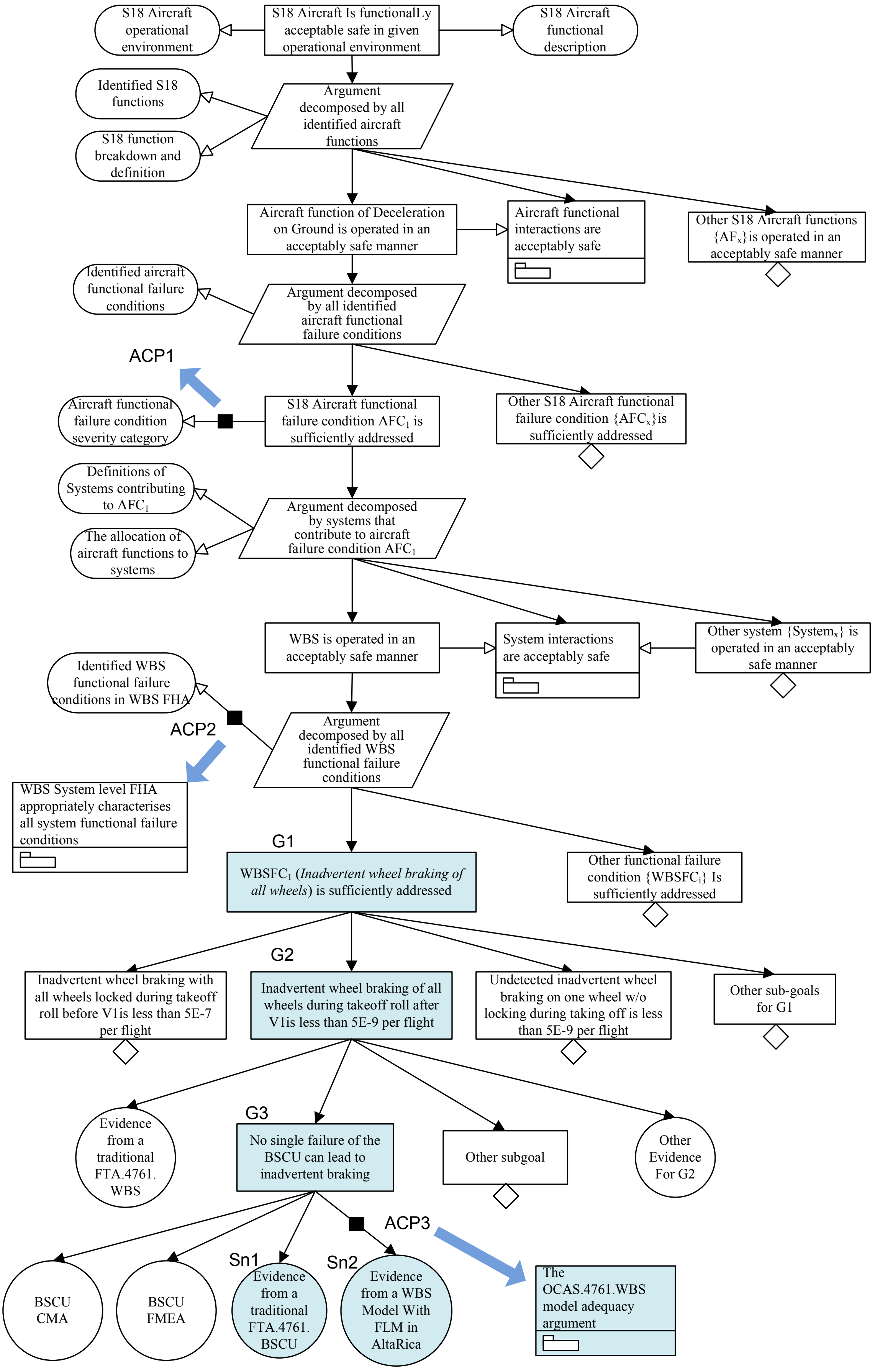

S18 Aircraft operational environment
S18 Aircraft Is functionalLy acceptable safe in given operational environment
S18 Aircraft functional description
Identified S18 functions
S18 function breakdown and definition
Argument decomposed by all identified aircraft functions
Aircraft function of Deceleration on Ground is operated in an acceptably safe manner
Aircraft functional interactions are acceptably safe
Other S18 Aircraft functions {AF$_x$} is operated in an acceptably safe manner
Identified aircraft functional failure conditions
Argument decomposed by all identified aircraft functional failure conditions
ACP1
Aircraft functional failure condition severity category
S18 Aircraft functional failure condition AFC$_1$ is sufficiently addressed
Other S18 Aircraft functional failure condition {AFC$_x$}is sufficiently addressed
Definitions of Systems contributing to AFC$_1$
The allocation of aircraft functions to systems
Argument decomposed by systems that contribute to aircraft failure condition AFC$_1$
WBS is operated in an acceptably safe manner
System interactions are acceptably safe
Other system {System$_x$} is operated in an acceptably safe manner
Identified WBS functional failure conditions in WBS FHA
ACP2
Argument decomposed by all identified WBS functional failure conditions
WBS System level FHA appropriately characterises all system functional failure conditions
G1
WBSFC$_1$ (Inadvertent wheel braking of all wheels) is sufficiently addressed
Other functional failure condition {WBSFC$_i$} Is sufficiently addressed
G2
Inadvertent wheel braking with all wheels locked during takeoff roll before V1is less than 5E-7 per flight
Inadvertent wheel braking of all wheels during takeoff roll after V1is less than 5E-9 per flight
Undetected inadvertent wheel braking on one wheel w/o locking during taking off is less than 5E-9 per flight
Other sub-goals for G1
Evidence from a traditional FTA.4761. WBS
G3
No single failure of the BSCU can lead to inadvertent braking
Other subgoal
Other Evidence For G2
ACP3
BSCU CMA
BSCU FMEA
Sn1
Evidence from a traditional FTA.4761. BSCU
Sn2
Evidence from a WBS Model With FLM in AltaRica
The OCAS.4761.WBS model adequacy argument

Figure 84 Primary Argument with BSCU Argument Elements

Through ‘rerunning’ the argument development underlying the primary safety argument structure presented in Figure 84 for the ARP 4761 example, we were able to explore how the extended process defined in Chapter 6 would have challenged the safety argument being developed at different points of the development process. For example, the following questions were raised by the expanded process:

- Is the aircraft S18 failure condition severity category definition compliant with the one suggested in ARP 4761? As shown by ACP1 in Figure 84, the failure condition severity category definition, as the basis of the goal, should be justified as suggested by the expanded step E1. Other severity category definitions can be considered at this point as suggested by the expanded step E2 and step E3.

- Is WBS FHA cited adequate for providing contextual information for the strategy employed? As shown by ACP2 in Figure 84, the model adequacy of WBS FHA, as the basis of the goal decomposition strategy, should be justified as suggested by the expanded step E4. Other potential basis of the strategy can be considered at this point as suggested by the expanded step E5 and step E6.

- Is the WBS Model by FLM in AltaRica cited adequate for being supporting evidence for G3? As shown by ACP3 in Figure 84, the model adequacy of OCAS.4761.WBS, as one of the solutions presented, should be justified as suggested by the expanded step E7. Other potential counter evidence for G3 should be explored in parallel to the presentation of the supporting evidence, as suggested by the expanded step E8 and step E9.

- Has the BSCU CMA results presented any violation of system independency required by FTA.4761.BSCU?

- Whether FTA.4761.BSCU and OCAS.4761.WBS are consistency models? If they were inconsistent, the trustworthiness of both models would be damaged.

In addition, although it is not shown in Figure 84, the evidence result assertion of OCAS.4761.WBS for supporting G3 is ‘MCS analysis of OCAS.4761.WBS indicates absence of Order One MCS’. With this evidence result assertion, if elicited, we can see the relevance of G3 and Sn2 and the reasoning gap between them.

C.5.2 Model Adequacy Argument Case Study

In terms of the focus of the thesis, the key points are to note how evidence items are used during the evolution of a safety argument (e.g. WBS FHA is cited as the context of a strategy; BSCU FMEA and BSCU FTA are cited as evidence for a goal). For evaluation purpose, we further develop the model adequacy argument associated with ACP3 as an example case, demonstrating the applicability of the model adequacy argument pattern and the use of evidence assertions.

Figure 85 presents the overview of the model adequacy argument in ASCE [16]. For evaluation and illustration purpose, two branches of argument have been developed with more detail. Figure 86 presents the decomposition of confidence argument of OCAS.4761.WBS on the basis of the construction elements of the model. Each system element being modelled or not being modelled in OCAS.4761.WBS have been considered in the example structure. The lower level supporting goals can be supported by the evidence descriptive assertions of OCAS.4761.WBS. For example, 'Gx4-1BSCU1.vaidity monitor is included/considered in OCAS.4761.WBS' can be supported by an evidence descriptive assertion of OCAS. 4761.WBS that describes the AltaRica Node of that system component – 'there is an AltaRica Node specified as `BSCU.BSCU1.MON`'.

Figure 87 presents the goals associated with the confidence associated with model consistency. The confidence claims based on model consistency have been put forward as suggested by the Model Adequacy Argument Pattern Part (b) as defined in Chapter 6. However, we cannot use our cross-model inconsistency to support these confidence claims. Because the six inconsistencies we identified through the analysis cannot be discharged. Some of the identified violations attack our confidence claim significantly. For example, if the failure mode 'failed stuck at position 2' is not considered for the Switch of BSCU in OCAS.4761.WBS, there is little confidence in citing OCAS.4761.WBS as the solution the domain safety claim of G3 in Figure 84. As illustrated in Figure 87, the confidence claim associated with model consistency have not been supported but been attacked (no tool support to described the *WeakenedBy* relationship in GSN yet). As a result, we should know that the top level claim 'GM1 – sufficient confidence exists in OCAS.4761.WBS as evidence for G3' in the example model adequacy argument is 'false' at this stage of safety case development.

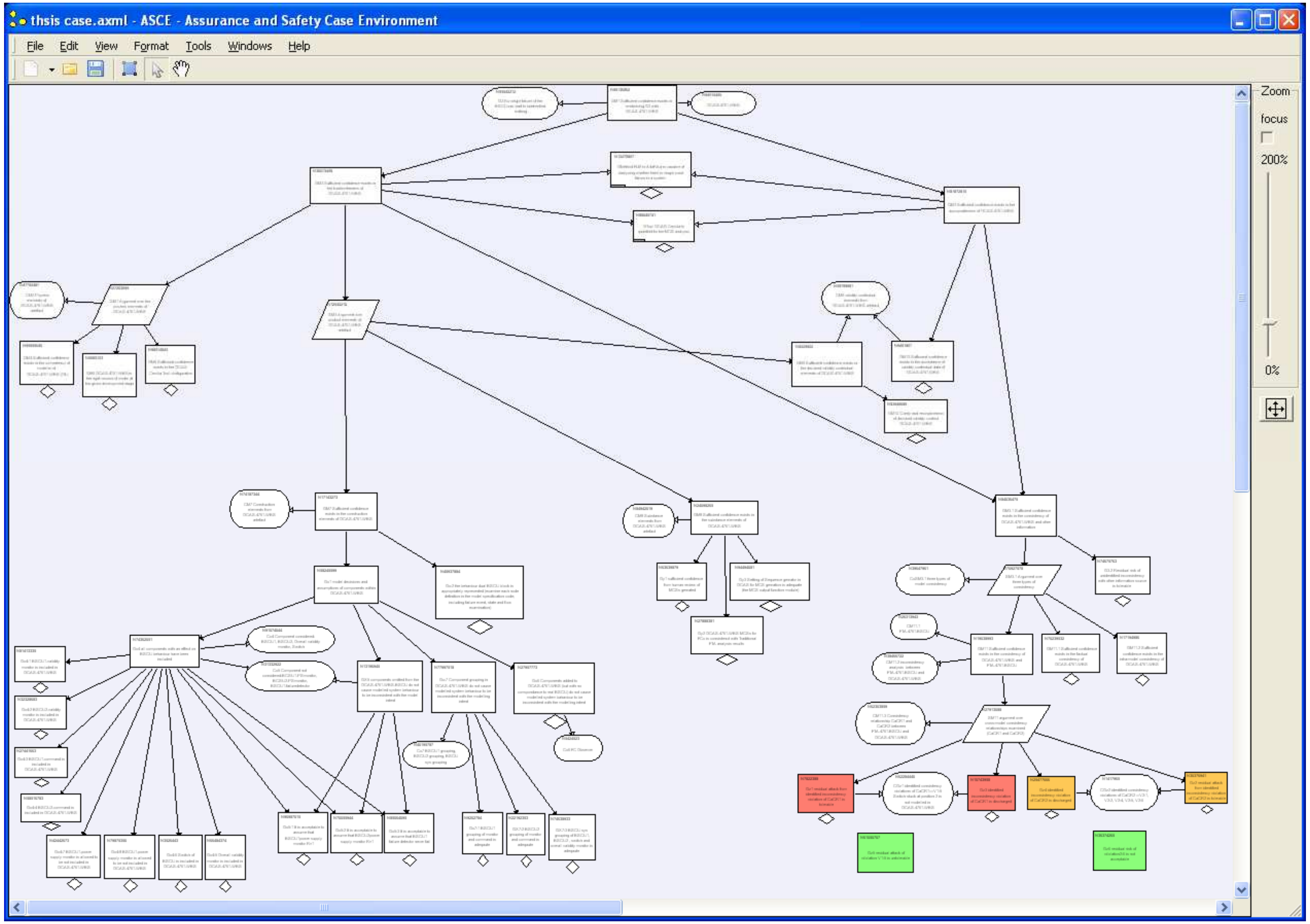
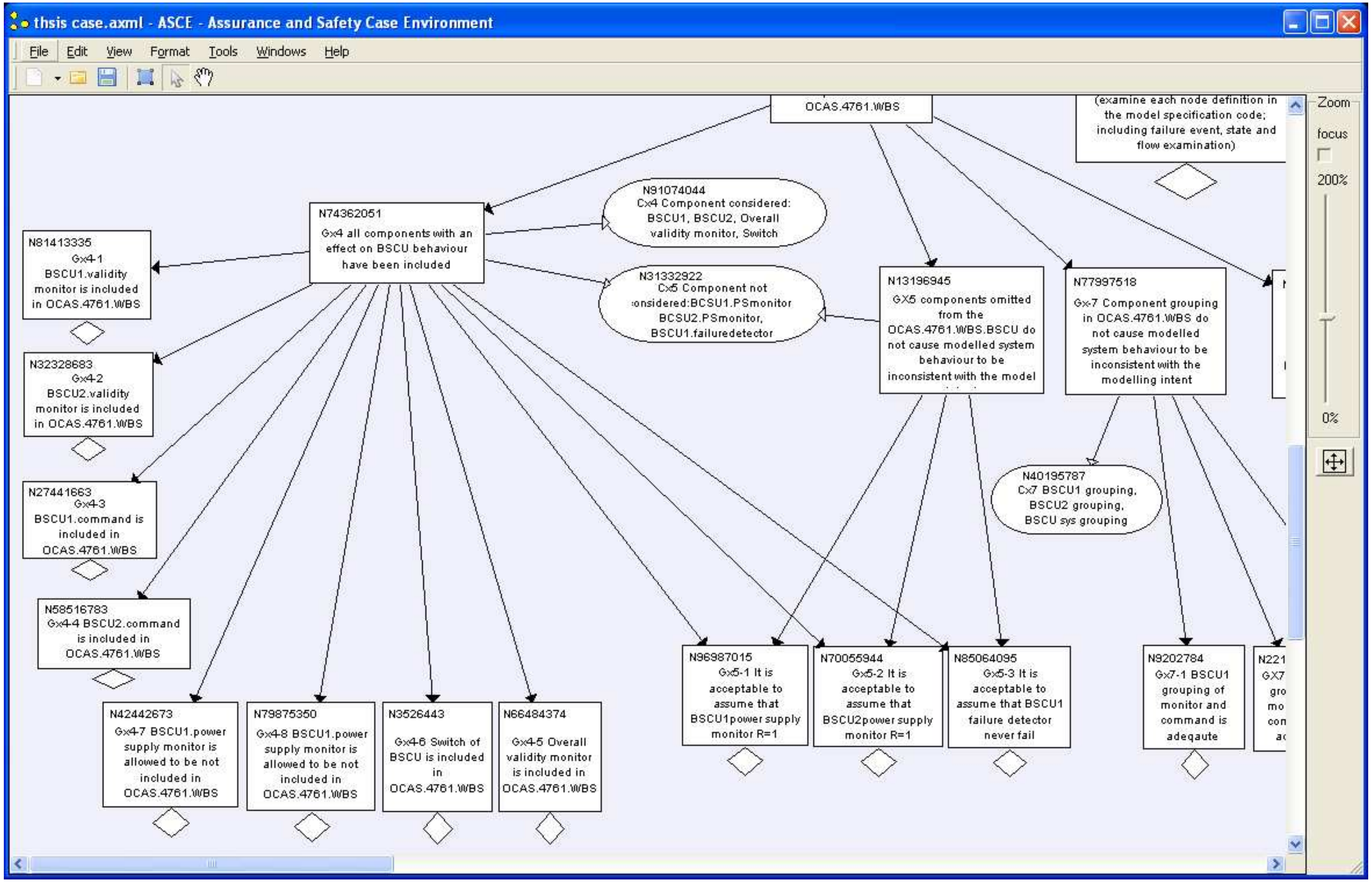

thsis case.axml - ASCE - Assurance and Safety Case Environment
File
Edit
View
Format
Tools
Windows
Help
Zoom
focus
200%
0%

Figure 85 Example OCAS.4761.WBS Model Adequacy Argument


thsis case.axml - ASCE - Assurance and Safety Case Environment
File
Edit
View
Format
Tools
Windows
Help
Zoom
focus
200%
0%

OCAS.4761.WBS

(examine each node definition in the model specification code; including failure event, state and flow examination)

N74362051
Gx4 all components with an effect on BSCU behaviour have been included

N91074044
Cx4 Component considered: BSCU1, BSCU2, Overall validity monitor, Switch

N31332922
Cx5 Component not :onsidered:BCSU1.PSmonitor BCSU2.PSmonitor, BSCU1.failuredetector

N13196945
GX5 components omitted from the OCAS.4761.WBS.BSCU do not cause modelled system behaviour to be inconsistent with the model

N77997518
Gx-7 Component grouping in OCAS.4761.WBS do not cause modelled system behaviour to be inconsistent with the modelling intent

N40195787
Cx7 BSCU1 grouping, BSCU2 grouping, BSCU sys grouping

N81413335
Gx4-1
BSCU1.validity monitor is included in OCAS.4761.WBS

N32328683
Gx4-2
BSCU2.validity monitor is included in OCAS.4761.WBS

N27441663
Gx4-3
BSCU1.command is included in OCAS.4761.WBS

N58516783
Gx4-4 BSCU2.command is included in OCAS.4761.WBS

N42442673
Gx4-7 BSCU1.power supply monitor is allowed to be not included in OCAS.4761.WBS

N79875350
Gx4-8 BSCU1.power supply monitor is allowed to be not included in OCAS.4761.WBS

N3526443
Gx4-6 Switch of BSCU is included in OCAS.4761.WBS

N66484374
Gx4-5 Overall validity monitor is included in OCAS.4761.WBS

N96987015
Gx5-1 It is acceptable to assume that BSCU1power supply monitor R=1

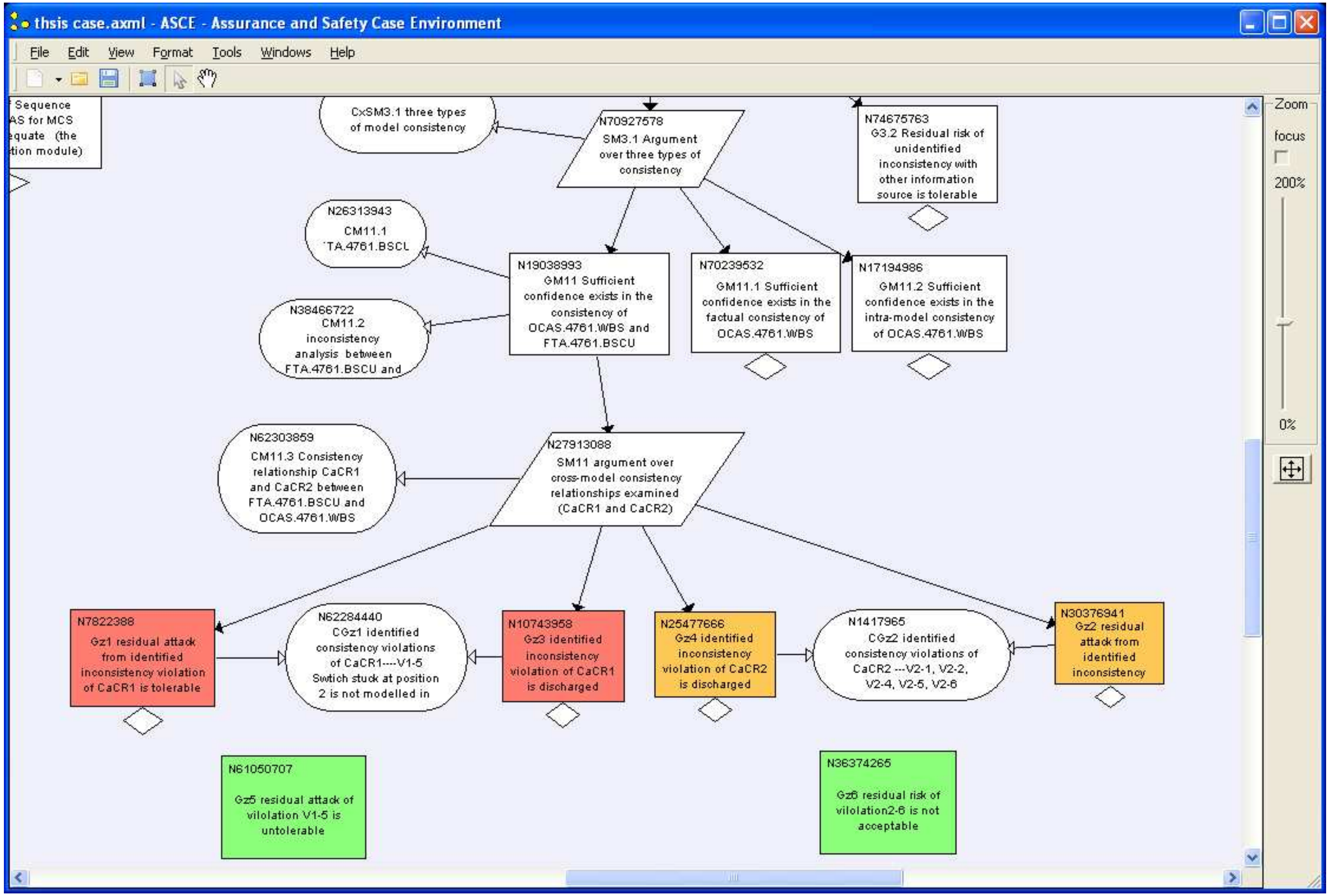
N70055944
Gx5-2 It is acceptable to assume that BSCU2power supply monitor R=1

N85064095
Gx5-3 It is acceptable to assume that BSCU1 failure detector never fail

N9202784
Gx7-1 BSCU1 grouping of monitor and command is adeqaute

N221
GX7
gro
mo
con
ac

Figure 86 Branch of Model Component Representation Justification


thsis case.axml - ASCE - Assurance and Safety Case Environment
File
Edit
View
Format
Tools
Windows
Help
Zoom
focus
200%
0%

f Sequence
AS for MCS
equate (the
tion module)

CxSM3.1 three types of model consistency

N70927578
SM3.1 Argument over three types of consistency

N74675763
G3.2 Residual risk of unidentified inconsistency with other information source is tolerable

N26313943
CM11.1
'TA.4761.BSCL

N19038993
GM11 Sufficient confidence exists in the consistency of OCAS.4761.WBS and FTA.4761.BSCU

N70239532
GM11.1 Sufficient confidence exists in the factual consistency of OCAS.4761.WBS

N17194986
GM11.2 Sufficient confidence exists in the intra-model consistency of OCAS.4761.WBS

N38466722
CM11.2
inconsistency analysis between FTA.4761.BSCU and

N62303859
CM11.3 Consistency relationship CaCR1 and CaCR2 between FTA.4761.BSCU and OCAS.4761.WBS

N27913088
SM11 argument over cross-model consistency relationships examined (CaCR1 and CaCR2)

N7822388
Gz1 residual attack from identified inconsistency violation of CaCR1 is tolerable

N62284440
CGz1 identified consistency violations of CaCR1----V1-5
Swtich stuck at position 2 is not modelled in

N10743958
Gz3 identified inconsistency violation of CaCR1 is discharged

N25477666
Gz4 identified inconsistency violation of CaCR2 is discharged

N1417965
CGz2 identified consistency violations of CaCR2 ---V2-1, V2-2, V2-4, V2-5, V2-6

N30376941
Gz2 residual attack from identified inconsistency

N61050707
Gz5 residual attack of vilolation V1-5 is untolerable

N36374265
Gz6 residual risk of vilolation2-6 is not acceptable

Figure 87 Branch of Model Consistency Justification

Glossary

ACP	Assurance Claim Points
ADAPT	Advanced Diagnostic and Prognostics Testbed
AIF	Argument Interchange Format
AMC	Acceptable Means of Compliance
ARM	Argumentation Metamodel
ARP	Aerospace Recommended Practice
ASA	Aircraft Safety Assessment
BSCU	Braking System Control Unit
CAE	Claims, Arguments and Evidence
CAIB	Columbia Accident Investigation Board
CCA	Common Cause Analysis
CoreDMM	Core Data Meta-Model
COTS	Commercial-Off-The-Shelf
CS	Certification Specifications
DS	UK Defence Standard
DSPN	Deterministic and Stochastic Petri Net
EMF	Eclipse Modelling Framework
ESACS	Enhanced Safety Assessment for Complex Systems
FDA	US Food and Drug Administration
FHA	Functional Hazard Assessment
FLM	Failure Logic Modelling
FMEA	Failure Modes and Effects Analysis
FPTA	Failure Propagation and Transformation Analysis
FPTC	Fault Propagation and Transformation Calculus
FPTN	Failure Propagation Transformation Notation
FTA	Fault Tree Analysis
GSN	Goal Structuring Notation
EC	European Commission
EviM	Evidence Model
ExCRs	Example Consistency Relationships
HAZOP	Hazard and Operability Study
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization

MCS	Minimal Cut Set
MDD	Model-Driven Development
MDE	Model-Driven Engineering
MISSA	More Integrated Systems Safety Assessment
MOD	Ministry of Defence
MOF	Meta-Object Facility
M&S	Models and Simulations
NASA	National Aeronautics and Space Administration
NLR	National Aerospace Laboratory
OMG	Object Management Group
PASA	Preliminary Aircraft Safety Assessment
PSSA	Preliminary System Safety Assessment
RAF	Royal Air Force
RCC	Reinforced Carbon-Carbon
SAE	Society of Automotive Engineers
SAEM	Software Assurance Evidence Metamodel
SACM	Structured Assurance Case Metamodel
SBVR	Semantics of Business Vocabulary and Business Rules
SCP	Supplementary Cooling Pack
SEAL	Safety Evidence Assurance Level
SEI	Software Engineering Institute
SSA	System Safety Assessment
STS	Space Transportation System
UML	Unified Modelling Language
WBS	Wheel Braking System

References

1. *AltaRica Description*. [retrieved 2012 19 Jan]; Available from: http://www.lix.polytechnique.fr/~rauzy/.
2. *"confidence". Oxford Dictionaries*. April 2010, Oxford University Press.
3. *"confirmation bias"*, in *A Dictionary of Psychology*, Colman, A.M., Editor. 2009, Oxford University Press.
4. *Control of Industrial Major Accidents Hazards Regulations (CIMAH)* 1984.
5. *IEC 61508 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems.* 1998.
6. *IEC 61511 Functional Safety - Safety Instrumented Systems for the Process Industry Sector*. 2003.
7. *ISO/DIS 26262 Road Vehicles - Functional Safety*. 2011.
8. *ISO/IEC 15026-2:2011 Systems and Software Engineering - Systems and Software Assurance - Part 2: Assurance Case*. 2011.
9. *ISO/IEC TR 15026-1:2010 Systems and Software Engineering - Systems and Software Assurance - Part 1: Concepts and Vocabulary*. 2010.
10. *ISO/IEC/IEEE 42010:2011(E) Systems and Software Engineering - Architecture Description.* (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000), 2011: p. 1-46.
11. *OpenFTA Software*. Auvation Software Limited [retrieved 2012 16 Jan]; Available from: http://www.openfta.com/.
12. *RAM Commander V8.2* 2011 [retrieved 2012 Feb 24]; Available from: http://www.aldservice.com/en/reliability-products/rams-software.html.
13. *Report of Columbia Accident Investigation Board, Volume I*. 2003 [retrieved 2012 Feb 09]; Available from: http://www.nasa.gov/columbia/home/CAIB_Vol1.html.
14. *Safety Methods Database Version 0.9*, Everdij, M.H.C. and Blom, H.A.P., Editors. 2010, Maintained by NLR-ATSI, available at http://www.nlr.nl/downloads/safety-methods-database.pdf.
15. Adelard. *Adelard Safety Case Development Manual V1.1*. 1998 [retrieved 2012 Feb 26]; Available from: http://www.adelard.com/resources/ascad.

16. Adelard, *The Adelard Safety Case Editor – ASCE*. 2003, Product description available at: http://adelard.co.uk/software/asce/.

17. Adelard. *CAE Notation Description*. [retrieved 2012 20 April]; Available from: http://www.adelard.com/asce/choosing-asce/cae.html.

18. Adeline, R., et al., *Toward a Validation Process for Model Based Safety Analysis*, in *the European Congress on Embedded Real-Time Software and Systems, ERTS²* 2010: Toulouse, France.

19. Adeline, R., et al., *Toward a Methodology for The AltaRica Modelling of Multi-Physical Systems*, in *European Conference on Safety and Reliability (ESREL)*. 2010, Taylor & Francis: Rhodes, Greece.

20. Alexander, C. and Brownstein, H., *A Pattern Language: Towns, Buildings, Construction*. 1977: Oxford University Press.

21. Alexander, R. and Kelly, T. *Simulation and Prediction in Safety Case Evidence*. in *the 26th International System Safety Conference*. 2008. Vancouver, Canada: System Safety Society.

22. Armstrong, J.M. and Paynter, S.E., *The Deconstruction of Safety Arguments through Adversarial Counter-Argument.* Reliability Engineering & System Safety, 2007. **92**(11): p. 1551-1562.

23. Arnold, A., et al., *The AltaRica Formalism for Describing Concurrent Systems.* Fundamenta Informaticae, 1999. **40**(2-3): p. 109-124.

24. Atego. *GSN Modeler*. 2008 [retrieved 2011 21 May]; Available from: http://www.atego.com/products/atego-gsn-modeler/.

25. Atkinson, C. and Kuhne, T., *Model-Driven Development: A Metamodeling Foundation.* Software, IEEE, 2003. **20**(5): p. 36-41.

26. Audi, R., *The Cambridge Dictionary of Philosophy*. 2nd ed. 1999: Cambridge University Press.

27. Ayoub, A., et al., *A Safety Case Pattern for Model-Based Development Approach*, in *the 4th NASA Formal Methods Symposium (NFM 2012)*, Goodloe, A.E. and Person, S., Editors. 2012, Springer-Verlag: Norfolk, VA, USA. p. 141-146.

28. Balci, O., et al. *Expanding Our Horizons in Verification, Validation, and Accreditation Research and Practice*. in *Proceedings of the Winter Simulation Conference*. 2002.

29. Barry, M.R. *CertWare: A Workbench for Safety Case Production and Analysis*. in *IEEE Aerospace Conference*. 2011.

30. Becker, J., Rosemann, M., and von Uthmann, C., *Guidelines of Business Process Modeling*, in *Business Process Management*. 2000. p. 241-262.

31. Bieber, P., et al., *Safety Assessment with AltaRica*, in *18th IFIP World Computer Congress, Topical Day on New Methods for Avionics Certification*. 2004. p. 505-510.

32. Bieber, P., Castel, C., and Seguin, C., *Combination of Fault Tree Analysis and Model Checking for Safety Assessment of Complex System*, in *Dependable Computing EDCC-4*. 2002. p. 624-628.

33. Bishop, P. and Bloomfield, R., *A Methodology for Safety Case Development*, in *Safety-critical Systems Symposium*. 1998: Birmingham, UK.

34. Bishop, P., Bloomfield, R., and Guerra, S. *The Future of Goal-Based Assurance Cases*. in *Proceedings of Workshop on Assurance Cases. Supplemental Volume of the 2004 International Conference on Dependable Systems and Networks* 2004.

35. Bishop, P.B., *Dependability of Critical Computer Systems, 3: Techniques Directory*. 1990: Elsevier.

36. Bishop, P.G., Bloomfield, R.E., and Froome, P.K.D., *Justifying the Use of Software of Uncertain Pedigree (SOUP) in Safety-Related Applications*, in *HSE Books*. 2001, Her Majesty's Stationery Office

37. Blackburn, S., *The Oxford dictionary of philosophy* Philosophy. 2005: Oxford University Press.

38. Blattnig, S.R., et al., *NASA Standard for Models and Simulations: Philosophy and Requirements Overview*, in *47th AIAA Aerospace Sciences Meeting and Exhibit*. 2009.

39. Bloomfield, R. and Bishop, P., *Safety and Assurance Cases: Past, Present and Possible Future – an Adelard Perspective*, in *Making Systems Safer*, Dale, C. and Anderson, T., Editors. 2010, Springer London. p. 51-67.

40. Bloomfield, R. and Littlewood, B. *Multi-Legged Arguments: The Impact of Diversity upon Confidence in Dependability Arguments*. in *Dependable Systems and Networks, 2003. Proceedings. 2003 International Conference on*. 2003.

41. Bloomfield, R.E., Littlewood, B., and Wright, D. *Confidence: Its Role in Dependability Cases for Risk Assessment*. in *the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*. 2007.

42. Bluvband, Z. and Grabov, P. *Failure Analysis of FMEA*. in *Reliability and Maintainability Symposium (RAMS)*. 2009.

43. Boiteau, M., et al., *The AltaRica Data-Flow Language In Use: Modeling Of Production Availability Of A Multi-State System.* Reliability Engineering System Safety, 2006. **91**(7): p. 747-755.

44. Bozzano, M. and Villafiorita, A., *Improving System Reliability via Model Checking: The FSAP/NuSMV-SA Safety Analysis Platform*, in *Computer Safety, Reliability, and Security (SAFECOMP)*. 2003. p. 49-62.

45. Braun, P., et al. *Model-Based Safety Cases for Software-Intensive Systems*. in *International Workshop on the Certification of Safety-Critical Software Controlled Systems (SafeCert)*. 2008.

46. Briones, J.F., et al., *Application of Safety Analyses in Model Driven Development*, in *Software Technologies for Embedded and Ubiquitous Systems*. 2007. p. 93-104.

47. Bullock, A. and Trombley, S., *The New Fontana Dictionary of Modern Thought* 1999, London: HarperCollins.

48. CAA. *CAP 670 Air Traffic Services Safety Requirements*. Civil Aviation Authority Safety Regulation Group 2012 [retrieved 2012 Mar 03]; Available from: http://www.caa.co.uk/docs/33/CAP670.PDF.

49. Cimatti, A., et al., *NuSMV 2: An OpenSource Tool for Symbolic Model Checking*. 2002, Computer Science Department, Paper 430, Carnegie Mellon University.

50. Clark, T., Sammut, P., and Willans, J., *Applied Metamodelling*, in *A Foundation for Language Driven Development*. 2008, Ceteva.

51. Clarke, D.M., *Recovering Errors in System Safety Analyses through Quality Checks.* Journal of System Safety 2012. **48**(2): p. 35-39.

52. Clemens, P., *Modeling in System Safety Analysis - A Significant Source of Error.* Journal of System Safety, 2009. **45**(1).

53. Clements, P. and Northrop, L., *Software Product Lines: Practices and Patterns.* 2002: Addison-Wesley.

54. Cockram, T., *Is This the Right Room for an Argument - Improving Arguments for Safety and Security* in *European Safety and Reliability Conference (ESREL)*. 2006: Estoril, Portugal.

55. Cohen, D.H., *Evaluating Arguments and Making Meta-Arguments.* Informal Logic, 2001. **21**(2): p. 73-84.

56. Conlin, H., Brabazon, P.G., and Lee, K., *Exploring the Role and Content of the Safety Case.* Process Safety and Environmental Protection, 2004. **82**(4): p. 283-290.

57. Crawford, J., *Some Ways of Improving Our Methods of Qualitative Safety Analysis and Why We Need Them*, in *Aspects of Safety Management*, Redmill, F. and Anderson, T., Editors. 2001, Springer.

58. Cullen, T.H.L., *The Public Inquiry Into the Piper Alpha Disaster 2 Volumes*. 1990: Her Majesty's Stationary Office.

59. Cyra, L. and Górski, J. *An Approach to Evaluation of Arguments in Trust Cases*. in *Dependability of Computer Systems, 2008. DepCos-RELCOMEX '08. Third International Conference on*. 2008.

60. Cyra, L. and Górski, J., *Support for Argument Structures Review and Assessment.* Reliability Engineering & System Safety, 2011. **96**(1): p. 26-37.

61. de Bono, E., *The Mechanism of Mind*. Pelican Books. 1971: Penguin Books.

62. de Bono, E., *Six Thinking Hats*. 2000: Penguin.

63. Denney, E., Pai, G., and Habli, I. *Towards Measurement of Confidence in Safety Cases*. in *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on*. 2011.

64. Despotou, G., *Managing the Evolution of Dependability Cases for Systems of Systems*. PhD Thesis, Dept. of Computer Science. 2007: University of York, UK.

65. DoD, *MIL-STD-882E Department of Defense Standard Practice: System Safety*. 2012.

66. Drabble, S., *Safety Process Measurement – Are We There Yet?*, in *Safety-Critical Systems: Problems, Process and Practice*, Dale, C. and Anderson, T., Editors. 2009, Springer London. p. 195-207.

67. Dugan, J.B., Bavuso, S.J., and Boyd, M.A., *Dynamic Fault-Tree models for Fault-Tolerant Computer Systems.* Reliability, IEEE Transactions on, 1992. **41**(3): p. 363-377.

68. EASA, *CS-25 Certification Specifications for Large Aeroplanes (Amendment 8).* 2009.

69. Easterbrook, S. and Nuseibeh, B., *Using ViewPoints for Inconsistency Management.* Software Engineering Journal, 1996. **11**(1): p. 31-43.

70. Eclipsepedia. *Emfatic*. 2008 [retrieved 2012 15 Jan]; Available from: http://wiki.eclipse.org/Emfatic.

71. Edwards, C., *Aircraft Operators have Built a Generic Hazard Model for Use in Developing Safety Cases.* ICAO Journal, 2000. **55**(1): p. 12-14.

72. Engels, G., et al., *A Methodology for Specifying and Analyzing Consistency of Object-Oriented Behavioral Models*, in *the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering*. 2001, ACM: Vienna, Austria.

73. EUROCONTROL. *Safety Case Development Manual*. 2006; 2.2:[Available from: http://www.eurocontrol.int/link2000/gallery/content/public/files/documents/Safety%20Case%20Development%20Manual%20V2.2_RI_13Nov06.pdf.

74. FDA, *Total Product Life Cycle: Infusion Pump - Premarket Notification Submissions (Draft Guidance).* Available from: http://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/UCM209337.pdf, 2010.

75. Felici, M., *Modeling Safety Case Evolution – Examples from the Air Traffic Management Domain*, in *Rapid Integration of Software Engineering Techniques*, Guelfi, N. and Savidis, A., Editors. 2006, Springer Berlin / Heidelberg. p. 81-96.

76. Fenelon, P., et al., *Towards Integrated Integrated Safety Analysis and Design.* Applied Computing Review, 1994. **2**(1): p. 21-32.

77. Fenn, J. and Jepson, B., *Putting Trust into Safety Arguments*, in *Constituents of Modern System-safety Thinking*, Redmill, F. and Anderson, T., Editors. 2005, Springer London. p. 21-35.

78. Fensel, D. and Benjamins, R., *Assumptions in Model-Based Diagnosis.* Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based System Workshop (KAW´96), 1996.

79. Firesmith, D., *Common Concepts Underlying Safety, Security, and Survivability Engineering*, in *CMU/SEI 2003-TN-033*. 2003.

80. Fischhoff, B., Slovic, P., and Lichtenstein, S., *Fault Trees: Sensitivity of Estimated Failure Probabilities to Problem Representation.* Journal of Experimental Psychology: Human Perception and Performance, 1978. **4**(2): p. 330-344.

81. Fowler, M., *Analysis Patterns: Reusable Object Models*. 1997: Addison Wesley.

82. Fox, J., *Probability, Logic and the Cognitive Foundations of Rational Belief.* Journal of Applied Logic, 2003. **1**(3-4): p. 197-224.

83. Goodenough, J., Weinstock, C.B., and Klein, A.Z., *Assessing Confidence in an Assurance Case*, in *the 5th Annual Layered Assurance Workshop (LAW 2011)*. 2011: Orlando, Florida, USA.

84. Govier, T., *A Practical Study of Argument*. 2010: Wadsworth, Cengage Learning.

85. Govier, T., *A Practical Study of Argument*. 1988: Wadsworth Pub. Co.

86. Greenwell, W.S., et al. *A Taxonomy of Fallacies in System Safety Arguments*. in *the 24th International System Safety Conference*. 2006. Albuquerque, USA.

87. Greenwell, W.S., Strunk, E.A., and Knight, J.C. *Failure Analysis and the Safety-Case Lifecycle*. in *IFIP Working Conference on Human Error, Safety and System Development (HESSD)*. 2004. Toulouse, France

88. Gross, R., *Decisions and Evidence in Medical Practice: Applying Evidence-Based Medicine to Clinical Decision Making*. 2001: Harcourt.

89. GSN Working Group, *GSN Community Standard Version 1*. 2011, GSN Working Group, Origin Consulting (York) Limited.

90. Habli, I., *Model-Based Assurance of Safety-Critical Product Lines*. PhD Thesis, Dept. of Computer Science. 2009: University of York, UK.

91. Haddon-Cave, C., *The Nimrod Review*. 2009: The Stationary Office (TSO).

92. Hamilton, V., *Criteria for Safety Evidence.* The Safety-Critical Systems Club Newsletter -Safety Systems, 2006. **16**(1).

93. Hardin, R., *Trust and Trustworthiness*. The Russell Sage Foundation Series on Trust ; v. 4. 2002: Russell Sage Foundation.

94. Hardy, T.L., *Using Lessons Learned to Promote a Healthy Skepticism in System Safety* in *the 28th International System Safety Conference (ISSC)*. 2010: Minneapolis, USA.

95. Hawkins, R., et al., *Using a Software Safety Argument Pattern Catalogue: Two Case Studies*, in *Computer Safety, Reliability, and Security*, Flammini, F., Bologna, S., and Vittorini, V., Editors. 2011, Springer Berlin: Heidelberg. p. 185-198.

96. Hawkins, R. and Kelly, T. *A Structured Approach to Selecting and Justifying Software Safety Evidence*. in *System Safety 2010, 5th IET International Conference on*. 2010.

97. Hawkins, R., et al., *A New Approach to Creating Clear Safety Arguments*, in *Advances in Systems Safety*, Dale, C. and Anderson, T., Editors. 2011, Springer London. p. 3-23.

98. Hawkins, R.D. and Kelly, T.P. *Software Safety Assurance - What is Sufficient?* in *Systems Safety 2009. Incorporating the SaRS Annual Conference, 4th IET International Conference on*. 2009.

99. Hitchcock, D. and Verheij, B., *Arguing on the Toulmin Model: New Essays in Argument Analysis and Evaluation*. 2006: Springer Netherlands.

100. Hitchcock, D. and Verheij, B., *Good Reasoning on the Toulmin Model*, in *Arguing on the Toulmin Model*. 2006, Springer Netherlands. p. 203-218.

101. Hodges, J.S. and Dewar, J.A., *Is It Your Model Talking? A Framework for Model Validation.* Rand Corporation Report R-4114-RC/AF, 1992.

102. Holloway, C.M. *Safety Case Notations: Alternatives for the Non-Graphically Inclined?* in *System Safety, 2008 3rd IET International Conference on*. 2008.

103. IAEA, *IAEA Safety Glossary*. 2007, Available at http://www-pub.iaea.org/MTCD/publications/PDF/Pub1290_web.pdf.

104. IEE, *Safety, Competency and Commitment: Competency Guidelines for Safety-Related System Practitioners*. 1999: Institution of Electrical Engineers, British Computer Society, Great Britain.

105. Isograph. *FaultTree+ V11 Technical Specification*. 2005 [retrieved 2012 9 Jan]; Available from: http://www.isograph-software.com/_techspecs/psa32techspec.pdf.

106. Jackson, D., Thomas, M., and Millett, L.I., *Software for Dependable Systems: Sufficient Evidence?* 2007: National Academies Press.

107. Jackson, M.A., *Problem Frames and Methods: Analysing and Structuring Software Development Problems*. 2001: Addison-Wesley/ACM Press.

108. Johnson, C., *Mapping the Impact of Security Threats on Safety-Critical Global Navigation Satellite Systems*, in *the 29th International Systems Safety Society*, Muniak, C.G., Editor. 2011, International Systems Safety Society: Las Vegas, USA.

109. Jolliffe, G. *Producing a Safety Case for IMA Blueprints*. in *Digital Avionics Systems Conference, 2005. DASC 2005. The 24th*. 2005.

110. Jones, M., *Potential Pitfalls in Relation to Safety Initiatives and other Aspects of Cultural Failure*, in *the 29th International System Safety Conference (ISSC)*. 2011: Las Vegas, USA.

111. Joshi, A. and Heimdahl, M.P.E., *Model-Based Safety Analysis of Simulink Models Using SCADE Design Verifier*, in *Computer Safety, Reliability, and Security*. 2005. p. 122-135.

112. Kelly, T., *Reviewing Assurance Arguments - A Step-by-Step Approach*, in *Workshop on Assurance Cases for Security - The Metrics Challenge, Dependable Systems and Networks (DSN)*. 2007.

113. Kelly, T., *A Systematic Approach to Safety Case Management*, in *SAE International, SAE World Congress*. 2003: Detroit, USA.

114. Kelly, T., *There's No Substitute for Thinking*. The Safety-Critical Systems Club Newsletter, 2011. **20**(3).

115. Kelly, T. and Bates, S., *The Costs, Benefits, and Risks Associated With Pattern-Based and Modular Safety Case Development*, in *Proceedings of the UK MoD Equipment Safety Assurance Symposium*. 2005.

116. Kelly, T. and McDermid, J., *Safety Case Construction and Reuse Using Patterns*, in *the 16th International Conference on Computer Safety and Reliability (SAFECOMP)*. 1997, Springer: York, UK.

117. Kelly, T.P., *Arguing Safety: A Systematic Approach to Managing Safety Cases*. PhD Thesis, Dept. of Computer Science. 1998: University of York, UK.

118. Kelly, T.P., *Can Process-Based and Product-Based Approaches to Software Safety Certification be Reconciled?*, in *Improvements in System Safety*, Redmill, F. and Anderson, T., Editors. 2008, Springer London. p. 3-12.

119. Kelly, T.P., *Concepts and Principles of Compositional Safety Cases*. 2001: COMSA/2001/1/1 - Research Report commissioned by QinetiQ.

120. Kelly, T.P. and McDermid, J.A., *A Systematic Approach to Safety Case Maintenance*. Reliability Engineering & System Safety, 2001. **71**(3): p. 271-284.

121. Kerr, J. *Impact Testing of the Orbiter Thermal Protection System*. 2003 [retrieved 2012 25 Feb]; Available from: http://research.jsc.nasa.gov/PDF/Eng-20.pdf.

122. Kim, I., Peak, R., and Sitaraman, S., *ROM: a Reliability Knowledge Representation for Collaborative System Design*. Engineering with Computers, 2010. **26**(1): p. 11-33.

123. Kirschner, P.A., Shum, S.J.B., and Carr, C.S., *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making*. Computer supported cooperative work, 1431-1496. 2003: Springer.

124. Kletz, T.A., *Hazop and Hazan: Identifying and Assessing Process Industry Hazards*. 1992: Institution of Chemical Engineers.

125. Kolovos, D.S., Paige, R.F., and Polack, F.A.C., *Model Comparison: a Foundation for Model Composition and Model Transformation Testing*, in *Proceedings of the 2006 international workshop on Global integrated model management*. 2006, ACM: Shanghai, China.

126. Konate, G. *AFI RVSM Post-Implementation Safety Case Version 1.0*. 2010; Available from: http://www.atns.co.za/PDF/AFI%20RVSM/AFI%20RVSM%20Post-Implementation%20Safety%20Case%20v1.0.pdf.

127. Last, J.M., *A dictionary of public health*. 2006: Oxford University Press.

128. Law, J. and Martin, E.A., *A dictionary of law*. Oxford dictionary of law. 2009: Oxford University Press.

129. Leveson, N., *SafeWare: System Safety and Computers*. 1995: Addison-Wesley.

130. Levins, R., *Strategy of Model Building in Population Biology*. American Scientist, 1966. **54**(4): p. 421-&.

131. Lisagor, O., *Failure Logic Modelling Handbook*, in *MISSA Project Deliverables*. 2011, University of York.

132. Lisagor, O., *Failure Logic Modelling: A Pragmatic Approach*. PhD Thesis, Dept. of Computer Science. 2010: University of York, UK.

133. Lisagor, O. and Kelly, T.P., *Application of the 'Lightweight Refinement' Relation to Establishing Confidence in Safety Assessment Models*, in *System Safety 2010, 5th IET International Conference on*. 2010. p. 1-7.

134. Littlewood, W. and Wright, D., *The Use of Multilegged Arguments to Increase Confidence in Safety Claims for Software-Based Systems: A Study Based on a BBN Analysis of an Idealized Example*. Software Engineering, IEEE Transactions on, 2007. **33**(5): p. 347-365.

135. Liu, S. and McDermid, J.A., *A Model-Oriented Approach to Safety Analysis Using Fault Trees and a Support System.* Journal of Systems and Software, 1996. **35**(2): p. 151-164.

136. Lloyd, E., Tye, W., and Great, B., *Systematic Safety: Safety Assessment of Aircraft Systems*. 1982: Civil Aviation Authority.

137. Long, R.A., *Beauty & the Beast – Use and Abuse of Fault Tree as a Tool,* in *the 17th International System Safety Conference*. 1999. p. 117-127.

138. Lutz, R. and Patterson-Hine, A. *Using Fault Modeling in Safety Cases*. in *Software Reliability Engineering, 2008. ISSRE 2008. 19th International Symposium on*. 2008.

139. Malhotra, M. and Trivedi, K.S., *Power-Hierarchy of Dependability-Model Types.* Reliability, IEEE Transactions on, 1994. **43**(3): p. 493-502.

140. Manion, M., *The Epistemology of Fault Tree Analysis: An Ethical Critique.* International Journal of Risk Assessment and Management, 2007. **7**(3): p. 382-430.

141. Mason, P.A.J., *MATrA : Meta-Modelling Approach to Traceability for Avionics*. PhD Thesis. 2002: University of Newcastle, Dept. of Compter Science.

142. Matsuno, Y. and Taguchi, K. *Parameterised Argument Structure for GSN Patterns*. in *Quality Software (QSIC), 2011 11th International Conference on*. 2011.

143. McCarl, B.A., *Model Validation: An Overview with Some Emphasis on Risk Models.* Review of Marketing and Agricultural Economics, 1984. **52**(03).

144. McDermid, J. and Rae, A., *Goal-Based Safety Standards: Promises and Pitfalls,* in *Achieving Systems Safety*, Dale, C. and Anderson, T., Editors. 2012, Springer London. p. 257-270.

145. Miguel, M.A.d., et al., *Integration of Safety Analysis in Model-Driven Software Development.* Software, IET, 2008. **2**: p. 260-280.

146. Mill, J.S., *A System of Logic, Ratiocinative and Inductive: Being a Connected View of the Principles of Evidence and the Methods of Scientific Investigation*. 10th ed. 1879, London: Longmans, Green, and Co.

147. Miser, H.J. and Quade, E.S., eds. *Handbook of Systems Analysis: Craft Issues and Procedural Choices*. 1988, North-Holland, New York.

148. MOD, *Defence Standard 00-56 Safety Management Requirements for Defence Systems Part 2 : Issue 4*, in *Guidance on Establishing a Means of Complying with Part 1*. 2007, Ministry of Defence.

149. MOD, *Defence Standard 00-56 Safety Management Requirements for Defence Systems, Part 1: Requirements, Issue 4*. 2007.

150. NASA, *Fault Tree Handbook with Aerospace Applications Vesion1.1*. 2002.

151. NASA, *NASA-STD-7009 Standard for Models and Simulations*. 2008.

152. NOPSEMA, *N-04300-GN0087 Guidance Note 'Safety Case Lifecycle Management' Rev 4*. 2011, National Offshore Petroleum Safety and Environmental Management Authority.

153. Nuseibeh, B., Easterbrook, S., and Russo, A., *Leveraging Inconsistency in Software Development.* Computer, 2000. **33**(4): p. 24-29.

154. O'Connor, P., *Standards in Reliability and Safety Engineering.* Reliability Engineering & System Safety, 1998. **60**(2): p. 173-177.

155. O'Connor, P.D.T., Newton, D., and Bromley, R., *Practical Reliability Engineering*. 4 ed. 2002: Wiley.

156. OMG, *Argumentation Metamodel*. 2010, ARM Working Document 1.0 Beta 1.

157. OMG, *Semantics of Business Vocabulary and Business Rules*. 2008, Version 1.0.

158. OMG, *Software Assurance Evidence Metamodel*. 2010, SAEM Working Document 1.0 Beta 1.

159. OMG. *Systems Assurance Task Force*. [retrieved 2012 15 Jan]; Available from: http://www.omgwiki.org/SysA/doku.php.

160. Oreskes, N., *Evaluation (not Validation) of Quantitative Models.* Environmental Health Perspectives, 1998. **106**: p. 1453-1460.

161. Oreskes, N. and Belitz, K., *Philosophical Issues in Model Assessment*, in *Model Validation: Perspectives in Hydrological Science*, Anderson, M. and Bates, P., Editors. 2001, John Wiley and Sons, New York. p. 23--42.

162. Oreskes, N., Shraderfrechette, K., and Belitz, K., *Verification, Validation, and Confirmation of Numerical-Models in the Earth-Sciences.* Science, 1994. **263**(5147): p. 641-646.

163. Ortmeier, F., et al., *Combining Formal Methods and Safety Analysis – The ForMoSA Approach*, in *Integration of Software Specification Techniques for Applications in Engineering*. 2004. p. 474-493.

164. Panesar-Walawege, R.K., et al. *Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard*. in *Software*

Testing, Verification and Validation (ICST), the 3rd International Conference on. 2010.

165. Papadopoulos, Y., et al., *Analysis and Synthesis of the Behaviour of Complex Programmable Electronic Systems in Conditions of Failure.* Reliability Engineering & System Safety, 2001. **71**(3): p. 229-247.

166. Pfeiffer, D. and Gehlert, A., *A Framework for Comparing Conceptual Models*, in *Workshop on Enterprise Modelling and Information Systems Architectures (EMISA)*. 2005. p. 108-122.

167. Pidcock, W. *What Are the Differences between a Vocabulary, a Taxonomy, a Thesaurus, an Ontology, and a Meta-model?* 2002 [retrieved 2011 5 April]; Available from: <http://infogrid.org/wiki/Reference/PidcockArticle>.

168. Pumfrey, D., *The Principled Design of Computer System Safety Analyses*. PhD Thesis. 1999: *Department of Computer Science,* University of York.

169. Rauzy, A., Gauthier, J., and Leduc, X., *Assessment of Large Automatically Generated Fault Trees by means of Binary Decision Diagrams.* Journal of Risk and Reliability, 2007. **221**(2): p. 95-105.

170. Reinhardt, D.W. and McDermid, J.A. *Assurance of Claims and Evidence for Aviation Systems*. in *System Safety 2010, 5th IET International Conference on*. 2010.

171. Rose, L.M., et al. *Enhanced Automation for Managing Model and Metamodel Inconsistency*. in *Automated Software Engineering, 2009. ASE '09. 24th IEEE/ACM International Conference on*. 2009.

172. Rothenberg, J., *The Nature of Modeling* in *AI, Simulation & Modeling*, Widman, L.E., Loparo, K.A., and Nielsen, N.R., Editors. 1989, John Wiley & Sons, Inc. p. 75−92.

173. Rouvroye, J.L. and van den Bliek, E.G., *Comparing Safety Analysis Techniques.* Reliability Engineering & System Safety, 2002. **75**(3): p. 289-294.

174. Rubinstein, M.F., *Patterns of Problem Solving*. 1974: Prentice-Hall.

175. Rumbaugh, J., Jacobson, I., and Booch, G., *The Unified Modeling Language Reference Manual*. 1999, Menlo Park: Addison-Wesley/Longman.

176. Rushby, J., *Formalism in Safety Cases*, in *Making Systems Safer*, Dale, C. and Anderson, T., Editors. 2010, Springer London. p. 3-17.

177. Russo, J.E. and Kolzow, K.J., *Where is the Fault in Fault Trees?* Journal of Experimental Psychology: Human Perception and Performance, 1994. **20**(1): p. 17-32.

178. Rykiel, E.J., *Testing Ecological Models: the Meaning of Validation.* Ecological Modelling, 1996. **90**(3): p. 229-244.

179. SAE, *ARP 4754 Certification Considerations for Highly-Integrated or Complex Aircraft Systems.* 1996.

180. SAE, *ARP 4754A Certification Considerations for Highly-Integrated or Complex Aircraft Systems.* 2010.

181. SAE, *ARP 4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment.* 1996.

182. Sargent, R.G. *Validation and Verification of Simulation Models*. in *Proceedings of the Winter Simulation Conference*. 2004.

183. Schum, D., et al., *Substance-Blind Classification of Evidence for Intelligence Analysis*, in *Ontology for the Intelligence Community (OIC)*. 2009: George Mason University, USA.

184. Shannon, R.E., *Simulation: A Survey with Research Suggestions.* AIIE Transactions, 1975. **7** (3): p. 289--301.

185. Shannon, R.E., *Systems Simulation: the Art and Science* 1975, Englewood, Cliffs, New Jersey : Prentice-Hall.

186. Sienou, A., et al., *Conceptual Model of Risk: Towards a Risk Modelling Language*, in *Web Information Systems Engineering – WISE 2007 Workshops*. 2007, Springer Berlin / Heidelberg. p. 118-129.

187. Spanoudakis, G. and Zisman, A., *Inconsistency Management in Software Engineering: Survey and Open Research Issues* in *Handbook of Software Engineering and Knowledge Engineering*, Chang, S.K., Editor. 2001, World Scientific Publishing Co. p. 329-380.

188. Spriggs, J., *GSN - The Goal Structuring Notation:A Structured Approach to Presenting Arguments*. 2012: Springer.

189. Steinberg, D., et al., *EMF: Eclipse Modeling Framework*. 2009: Addison-Wesley.

190. Storey, N., *Safety-Critical Computer Systems*. 1996: Addison-Wesley.

191. Straeten, R.V.D., *Inconsistency Management in Model-Driven Engineering*. PhD Thesis. 2005: Department of Computer Science, Vrije Universiteit Brussel.

192. Strigini, L., *Formalism and Judgement in Assurance Cases*, in *Workshop on Assurance Cases: Best Practices, Possible Obstacles, and Future Opportunities at International Conference on Dependable Systems and Networks (DSN)*. 2004: Florence, Italy.

193. Sun, L. and Kelly, T., *Evaluating Safety Analysis - a Meta-Model Perspective*, in *the 29th International System Safety Conference (ISSC)*. 2011: Las Vegas, USA.

194. Sun, L. and Kelly, T., *Managing Inconsistency in Safety Analysis: an Initial Exploration*, in *European Safety and Reliability Conference (ESREL)*. 2011: Troyes, France

195. Sun, L. and Kelly, T. *Safety Arguments in Aircraft Certification*. in *Systems Safety 2009. Incorporating the SaRS Annual Conference, 4th IET International Conference on*. 2009.

196. Sun, L., Lisagor, O., and Kelly, T. *Justifying the Validity of Safety Assessment Models with Safety Case Patterns*. in *System Safety, 2011 6th IET International Conference on*. 2011.

197. Suokas, J., *Evaluation of the Quality of Safety and Risk Analysis in the Chemical Industry*. Risk Analysis, 1988. **8**(4): p. 581-591.

198. Toulmin, S.E., *The Uses of Argument*. 1958, Cambridge: University Press.

199. Toulmin, S.E., Rieke, R.D., and Janik, A., *An Introduction to Reasoning*. 1979: Macmillan.

200. Vesely, W.E., et al., *Fault Tree Handbook (NUREG-0492)*. 1981.

201. Villemeur, A., *Reliability, Availability, Maintainability and Safety Assessment*. 1992: Wiley.

202. Walker, M., Bottaci, L., and Papadopoulos, Y., *Compositional Temporal Fault Tree Analysis*, in *Computer Safety, Reliability, and Security*. 2008. p. 106-119.

203. Walton, D., *Argument Visualization Tools for Corroborative Evidence*, in *the 2nd International Conference on Evidence Law and Forensic Science*. 2009, Institute of Evidence Law and Forensic Science: Beijing. p. 32-49.

204. Weaver, R., *The Safety of Software - Constructing and Assuring Arguments*. PhD Thesis, Dept. of Computer Science. 2004: University of York, UK.

205. Weaver, R., Mayor, P., and Kelly, T.P. *Gaining Confidence in Goal-based Safety Cases*. in *Proceedings of 14th Safety Critical Systems Symposium*. 2006: Springer.

206. Weisberg, M., *Forty Years of 'The Strategy': Levins on Model Building and Idealization.* Biology & Philosophy, 2006. **21**(5): p. 623-645.

207. White, T., *Fault Tree Analysis - More Than Just Minimal Cut Set Generation*. MSc Thesis, Dept. of Computer Science. 2004: University of York, UK.

208. Wigmore, J.H., *The Problem of Proof.* Illinois Law Review, 1913. **8**(2): p. 77-103.

209. Wijayarathna, P.G. and Maekawa, M. *Extending Fault Trees with an AND-THEN Gate*. in *the 11th International Symposium on Software Reliability Engineering (ISSRE)*. 2000.

210. Wilkinson, P.J. and Kelly, T.P. *Functional Hazard Analysis for Highly Integrated Aerospace Systems*. in *Certification of Ground/Air Systems Seminar (Ref. No. 1998/255), IEE*. 1998.

211. Wilson, S.P., Kelly, T.P., and McDermid, J.A., *Safety Case Development: Current Practice, Future Prospects*, in *1st ENCRESS/12th CSR Workshop*. 1995, Springer.

212. Wilson, S.P. and McDermid, J.A., *Integrated Analysis of Complex Safety Critical Systems.* The Computer Journal, 1995. **38**(10): p. 765-776.

213. Wong, J.-T. and Yeh, W.-C., *Validation of Fault Tree Analysis in Aviation Safety Management.* Journal of Air Transportation, 2007.

214. Ye, F., *Justifying the Use of COTS Components within Safety Critical Applications.* PhD Thesis, Dept. of Computer Science. 2005: University of York, UK.

215. Yuan, T. and Kelly, T., *Argument Schemes in Computer System Safety Engineering.* Informal Logic, 2011. **31**(2): p. 89-109.

216. Zoughbi, G., Briand, L., and Labiche, Y., *Modeling Safety and Airworthiness (RTCA DO-178B) Information: Conceptual Model and UML Profile.* Software and Systems Modeling, 2011. **10**(3): p. 337-367.