An ontology-based analysis method for assessing and improving the quality of hazard analysis results

Abigail Parisaca Vargas

Department of Computer Science
School of Mathematics, Computer Science, and Engineering
City University of London

Submitted in part fulfilment of the requirements for the degree of Doctor of Philosophy in Computer Science of City University of London and the Diploma of City University of London, August 2021

For my two Marias, my two Josefinas and my one Pavel

Contents

1 Introduction **17**

1.1 Motivation . . . 17

1.2 Problem Statement . . . 19

1.3 Structure of the Thesis . . . 19

2 Basic Concepts and Definitions **21**

2.1 System . . . 21

2.2 Systems Safety . . . 21

2.3 Hazards . . . 22

2.4 Ontologies . . . 24

2.4.1 Knowledge Representation and Inference Mechanisms . . . 24

2.4.2 OWL, Protégé and SWRL . . . 25

2.5 Model checking . . . 25

2.5.1 Definition . . . 26

2.5.2 Kinds of Model checking . . . 26

2.5.3 Temporal Logics . . . 27

2.5.4 MCMAS Model Checkers . . . 28

2.6 Hazard Analysis . . . 29

2.7 Effectiveness of Hazard Analysis Techniques . . . 29

2.7.1 Methods to measure effectiveness . . . 29

2.8 Identifying Hazards . . . 30

2.9 Discussion . . . 31

3 Related Work **33**

3.1 Traditional Hazard Analysis Techniques . . . 34

3.1.1 HAZOPS . . . 34

3.1.2 Fault Tree Analysis . . . 35

3.1.3 Failure Mode and Effects Analysis . . . 37

3.1.4 Preliminary Hazard Analysis . . . 38

3.1.5 Bow Tie Analysis . . . 40

3.1.6 Standards . . . 40

3.2 Recent Hazard Analysis Techniques . . . 42

3.2.1 STPA 42
3.2.2 OHA 44
3.3 Discussion 45
3.4 Related research on Ontologies and Hazard Analysis 46
3.4.1 Representing hazards 46
3.4.2 Representing Hazard Analysis Techniques 47
3.4.3 Using ontologies for modeling 49
3.5 The Utility of Ontologies in Hazard Analysis 50
3.5.1 Using ontologies to better describe hazards in a system 51
3.6 Uncertainty in Hazard Analysis 52
3.7 Reasoning, hazard knowledge and ontologies 53
3.7.1 Creativity 53
3.7.2 A different kind of knowledge 55
3.7.3 Common sense 56
3.7.4 Human-machine synergy 57
3.7.5 Thesis author's view 57

4 The Ontology-Based Analysis Method **58**
4.1 Overview of the analysis method 58
4.2 Foundational Ontology 59
4.2.1 Developing the Ontological Model 59
4.2.2 Core Ontology 61
4.2.3 Extended Ontology 64
4.2.4 Using the Ontology to Reason about Hazards 65
4.3 Step 1: Systematic Exploration of Dependencies 69
4.3.1 Stage 1: Hazard Analysis and System Description 69
4.3.2 Stage 2: Systematic Revision 69
4.3.3 Stage 3: Ontology-guided revision of the inputs 69
4.3.4 Stage 4: Search implicit and overlooked relationships 73
4.4 Second Part of the Method: State based hazard model analysis 79
4.4.1 Stage 5: Generation of State-Based Hazard Models from Ontology Rules 79
4.4.2 Stage 6: Exploration of the state-based hazard model 80
4.4.3 Stage 7: Visualizing Hazard Paths 81
4.5 Developing a Prototype 81
4.5.1 Overview 81
4.5.2 Translating from the intermediate format into a Transition System 81
4.5.3 Hazard Model 82

5 Evaluation Framework **89**
5.1 Summary 89

5.2 Outputs . . . 89
5.3 Outcome . . . 92
5.4 Empirical Studies . . . 93

6 Medication System Case Study 95
6.1 System Overview . . . 95
6.1.1 Hazard Analysis of the Medication Prescription System . . . 95
6.2 Application of the Ontology-Based Method to the Medication Prescription System Case Study . . . 97
6.2.1 Step 1: Carry out a systematic exploration of dependencies . . . 97
6.2.2 Step 2: Perform an automated search of hazard paths . . . 97
6.3 Discussion . . . 100

7 Infusion Pump Case Study 101
7.1 System Overview . . . 101
7.1.1 Hazard Analysis of the Data Entry Subsystem . . . 101
7.2 Application of the Ontology-Based Method to the Data Entry Subsystem of the Infusion Pump . . . 107
7.2.1 Step 1: Carry out a systematic exploration of dependencies . . . 107
7.2.2 Step 2: Perform an automated search of hazard paths . . . 110
7.3 Discussion . . . 111

8 Voting System Pret a Voter 112
8.1 System Overview . . . 112
8.1.1 Ballot Form Generation . . . 112
8.1.2 Vote Casting . . . 113
8.1.3 Audit of Ballot Forms . . . 113
8.1.4 Audit of Casted Votes . . . 114
8.2 Modelling and Analysis of Prêt á Voter in HAZOP . . . 114
8.2.1 Modelling . . . 114
8.2.2 Results of the HAZOP analysis . . . 117
8.3 Application of the Ontology-Based Method to the Prêt á Voter HAZOP Analysis 117
8.3.1 Step 1: Carry out a systematic exploration of dependencies . . . 118
8.3.2 Results . . . 143
8.4 Discussion and Conclusions . . . 146

9 The West Corridor Light Rail Transit(LRT) System 150
9.1 System Overview . . . 150
9.2 Preliminary Hazard Analysis of the LRT System . . . 151
9.3 Application of the Ontology-Based Method to the West Corridor LRT Case Study153

9.3.1 Step 1: Carry out a systematic exploration of dependencies 153
9.3.2 Stage 1: Hazard Analysis and System Description 153
9.3.3 Stage 2: Systematic revision 153
9.3.4 Stage 3: Ontology guided revision of inputs 154
9.3.5 Stage 4: Search implicit and overlooked relationships 162
9.3.6 Step 2: Perform an automated search of hazard paths 165
9.4 Discussion 166

10 In-Trail Procedure Case Study 167
10.1 System Overview 167
10.2 Preliminary Hazard Analysis for ITP 168
10.2.1 Application of the Ontology-Based Method to the ITP Case Study . . . 177
10.2.2 Discussion 192

11 Conclusion 196
11.1 Results 196
11.2 Limitation of the research 200
11.2.1 Human Factors 201
11.3 Limitations of the chosen evaluation framework 202
11.4 Practical implications and challenges that might arise when adopting this approach by new parties 202
11.5 Future Directions 203
11.5.1 Discussion 203

Bibliography 205

A Appendix A 214

List of Figures

3.1 Fault Tree Symbols (Based on figure from [Ste12]) 36
3.2 Basic Control Structure (Taken from [Lev13]) 43
3.3 Control Loop (Taken from [Lev13]) 44

4.1 Diagram presenting an overview of the stages of the analysis method presented in this thesis. 60
4.2 Concepts that are part of our ontology 61
4.3 Part of Ontology 61
4.4 Core classes of the ontology 62
4.5 Hazard in Systems. Based on [Rai01b]. 63
4.6 Class hierarchy of Hazards in the ontology 64
4.7 Different relationships represented in our Ontology 71
4.8 Similarity 73
4.9 Entailment 74
4.10 Synonymy 75
4.11 First: Synonymy Algorithm example 77
4.12 Second: Synonymy Algorithm example 77
4.13 Third: Synonymy Algorithm example 77
4.14 Transitions 80
4.15 Paths 80
4.16 Unlabelled Transition System 83
4.17 Transition System for our Example 85
4.18 State space of causes *f1* and *f2* and hazards *h1* and *h2* 86

6.1 Medication Prescription System Structure 97
6.2 Medication Prescription System Processes 98
6.3 Data Entry Hazard Development 99

7.1 User Interface System, Data Entry Subsystem Structure 107
7.2 Processes of the Data Entry Subsystem 108
7.3 Data Entry Hazard Development 111

8.1 Ballot form of the Prêt á Voter System 113
8.2 Detached sides of the Prêt á Voter Ballot form 113

8.3 First page of the Prêt á Voter HAZOP Analysis 115
8.4 Pret A Voter Structure 120
8.5 Protégé:Sample of components represented 123
8.6 Protégé:Sample of processes represented 123
8.7 Protégé:Sample of causes represented 124
8.8 Protégé: Sample of hazards represented 124
8.9 Protégé:Sample of components represented 125
8.10 Protégé:Rules in the ontology 125
8.11 Protégé:Starting the reasoner 126
8.12 Protégé:An example of the results 126
8.13 Process: Teller does not publish keys 127
8.14 Pret A Voter Processes Prevoting 134
8.15 Pret A Voter Processes Voting 136
8.16 Pret A Voter Processes Posvoting 140
8.17 Hazard: Voter does not present RHS for scanning 145
8.18 Hazard: Voter does not take receipt 146
8.19 Example of the relationships of deviations *Teller does not generate keys* 149

9.1 Schematic diagram of the catenary support system (modified from [www14c]) . . 151
9.2 The West Corridor LRT System top structure 154
9.3 The West Corridor LRT System main processes 157
9.4 Subsystems of LRT System represented in the ontology 160
9.5 Subset of Processes represented in the ontology 160
9.6 Subset of hazards represented in the ontology 161
9.7 Starting the inference engine 162
9.8 Example results produced by Protégé for driving and crossing. 163
9.9 Example result produced by Protégé for the catenary system. 163
9.10 Adding a non direct causality relationship 164
9.11 Hazard path for ”LRV Fire” 165

10.1 Basic Control Structure for ITP (Taken from [Lev13]) 178
10.2 Control Loop (Taken from [FC12]) 180
10.3 Control Loop(Taken from [FC12]) 181
10.4 Processes of the ITP System 183
10.5 Subset of Processes of the ITP System in Protégé 184
10.6 Parts of ITP System 185
10.7 Subset of Processes of the ITP System in Protégé 185
10.8 Subset of Hazards and Causes of the ITP System in Protégé 186
10.9 Results shown in Protégé 188

10.10 Hazard path that leads to hazard “Flight crew executes a cleared ITP which does not meet criteria” 190
10.11 Scenario for hazard “Flight crew executes a cleared ITP which does not meet criteria” 194
10.12 Scenario for hazard “Flight crew executes ITP procedure after request but without clearance” 195
10.13 Scenario for hazard “Pilot aborts an ITP clearance when criteria is met” 195

11.1 HAZOP strategy: How to fill in HAZOP worksheets 199

List of Tables

3.1 Results of hazards and operability study of olefin dimerisation unit (snippet). . . 36
3.2 FMEA worksheet 3-safety/reliability (based on [Eri05]). 38
3.3 PHA worksheet (based on [Eri05]). 39
3.4 STPA table 44
3.5 Two descriptions of the same hazard 50

4.1 Basic example of HazID worksheet 61
4.2 Hazard and Operability Studies Table 70
4.3 Preliminary Hazard Analysis Table 71
4.4 Classes and Elements of its class 88
4.5 Hazards and CauseHazards for modeling 88

5.1 Stages of the method and what was applied in each case study 91

6.1 Preliminary Hazard Analysis 96

7.1 Preliminary Hazard Analysis 103
7.2 109
7.3 Hazards and Cause Hazards with relationship $isCauseHazardOf$ 110

8.1 Structural representation of Cryptographic system 128
8.2 Structural representation of Scanning Services 129
8.3 Task (or processes) vs Phases 129
8.4 Teller Tasks 129
8.5 Teller Auditor Tasks 130
8.6 Ballot Generation Authority Tasks 130
8.7 Scanning Staff Tasks 130
8.8 Registration Staff Tasks 130
8.9 Scanning System Tasks 131
8.10 Web Bulletin Board Tasks 131
8.11 Prevoting phases vs Task and Processes 132
8.12 Prevoting phases vs Task and Processes 133
8.13 Voting phases vs Task and Processes 135
8.14 Voting phases vs Task and Processes 137

8.15 Voting phases vs Task and Processes 138
8.16 Voting phases vs Task and Processes 139
8.17 Post-voting phase vs Task and Processes 141
8.18 Post-voting phase vs Task and Processes 142
8.19 Deviations found to have similarities 148

9.1 Preliminary Hazard Analysis 152
9.2 Subsystem Track 155
9.3 Subsystem Overhead Contact System 155
9.4 Subsystem Stations 156
9.5 Subsystem Train Control 156
9.6 Subsystem Alignment 156
9.7 Process: Managing alignment 157
9.8 Process Transmission of Power 158
9.9 Process: Grade Crossing Management 158
9.10 Process: Managing Stations 159
9.11 Process: Managing Tracks 159

10.1 Preliminary Hazard Analysis for ITP 169
10.1 Preliminary Hazard Analysis for ITP 170
10.1 Preliminary Hazard Analysis for ITP 171
10.1 Preliminary Hazard Analysis for ITP 172
10.1 Preliminary Hazard Analysis for ITP 173
10.1 Preliminary Hazard Analysis for ITP 174
10.1 Preliminary Hazard Analysis for ITP 175
10.1 Preliminary Hazard Analysis for ITP 176
10.2 Hazards of PHA for ITP vs Unsafe control actions of STPA applied to ITP . . . 182
10.3 Hazard path that leads to hazard “Flight crew executes a cleared ITP which does not meet criteria” 190
10.4 Scenario where hazard and causes are shown 191
10.5 Scenario where a path of hazards and their locations are shown 191
10.6 Scenario for hazard ”Flight crew executes ITP procedure after request but without clearance” 192
10.7 Scenario for hazard “Flight crew executes a cleared ITP which does not meet criteria” 193
10.8 Scenario for hazard “Pilot aborts an ITP clearance when criteria is met” 194

Acknowledgements

I would like to express my special appreciation and thanks to my supervisors Professor Robin Bloomfield and Professor Lorenzo Strigini for the guidance, encouragement and advice they have provided throughout my PhD studies. I would also like to express my special thanks to Adelard LLP and City University of London for providing the funding which allowed me to undertake my research.

I am especially grateful to Dr. Paolo Masci for all his advice and comments on my thesis and on my research but, above all, for being such a good friend. Additionally, I would like to thank the team at the Centre for Software Reliability for their stimulating discussions and support.

I would also like to thank all of my friends who supported me and incented me to strive towards my goal, with a special mention to Achille.

Furthermore, I would like to thank my mother who has accompanied me and supported me in all ways she possibly could. A special thanks to my sisters Analia and Desiree for their support, encouragement and love, it has helped me to keep working. In addition, thanks to my mother in law who has always been keen to help when I was in need.

Last but not least, I would like to thank my husband Pavel and my daughters Anna-Josefina and Maria-Sofia for being the means by which I learnt that one can actually be born again and because through them work has acquired a different meaning.

Declaration of Originality

I hereby declare that the work presented in this thesis is my own.

Abstract

Safety-critical systems such as medical devices and avionics systems are developed using systematic processes and rigorous analysis methods. This is necessary to gain strong confidence that the system is not affected by latent design problems that may lead to system failures or unintended behaviours that, ultimately, could result in damage or harm to people or the environment. Whilst different guidelines and recommended best development practices are provided in different regulatory frameworks and standards, all processes share a common initial stage, known as hazard analysis. The aim of the hazard analysis is to identify all known and foreseeable scenarios and problematic situations. It is important that the hazard analysis is as accurate and as comprehensive as possible since the entire development process builds on the hazard analysis results. Any missed scenario or overlooked problematic situation could breach the mitigation strategies designed to guarantee the safety of the system.

Several hazard analysis techniques have been introduced over the last 50 years to improve the quality of the analysis. However, a known weakness of the current generation of techniques is that they often rely on manual analysis of information recorded in textual format. For realistic, complex systems, the amount of information is usually abundant and overwhelming. Because of this, even the most expert analyst can accidentally overlook important aspects of the system that should have been considered to ensure the safety of the system. The research work presented in this thesis aims to provide a systematic and comprehensive way to help the expert analyst with his task.

This thesis explores the development of a novel method and supporting analysis tool for the refinement of the hazard analysis results. The method is structured into a series of stages, each of which provides feedback to the analysts to help them gain confidence in the quality of the analysis. The method also helps to identify and resolve weaknesses in the analysis, if they are present. The research builds an ontology to represent knowledge collected during the hazard analysis. Inference rules are used to reason about possible scenarios, hazards, hazard causes and their relations. Formal (i.e., mathematically-based) tools are used to mechanise the exploration of scenarios, discover relations between hazards and causes that may have been overlooked during the analysis. The effectiveness of the proposed method is evaluated using various realistic case studies from different application domains.

1 Introduction

Any kind of system, from a simple medication prescription system to a complex avionics system, can pose risks that could cause incidents leading to damage to people or the environment, including injury and even death. Safety Engineering is a practice recommended in several standards and regulatory frameworks to avoid accidents and incidents. Within Safety Engineering, Hazard Analysis is the process of analysing a system to identify, evaluate, remove and mitigate all relevant hazards.

This thesis explores the development of a method and supporting tool to help safety analysts assess the quality of hazard analysis results, and improve the results when necessary. The proposed method uses knowledge engineerig tools and concepts combined wiht reasoning tools as well as model checking techniques that help to explore the hazard analysis even further.

1.1 Motivation

One of the keys to system safety is an effective analysis of hazards. Hazard Analysis techniques were created over 50 years ago, out of necessity for systematic and comprehensive methods to engineer safer systems. Hazard Analysis starts with Hazard Identification.

Hazard Analysis and Hazard Identification are a difficult and essential part of Safety Engineering. These activities are very demanding and mostly manual. There is an increasing need of improved analysis tools and techniques. Hazard analysis is the identification of hazards and their initiating causes [Int00]. It can be thought of as the process of investigating an accident before it actually occurs. Its aim is to identify exhaustively all possible causes of accidents, so that they can be eliminated or controlled before they occur [Lev11]. Hazard Analysis is extremely important, and lack of completeness in the analysis can have serious consequences.

Hazard Identification (HazID) is one of the most important parts of the hazard analysis, as it forms the basis of the activities carried out to design a safe system. By looking at the system from a safety perspective, safety analysts check important aspects of the system and try to identify hazards that could happen. The analysis is complex because all possible and foreseeable scenarios of operation need to be considered. Various studies highlighted that the most significant flaws in hazard analysis techniques applied at the early stages of system design are often related to the omission of hazards and hazard causes [Har10]. For example, one reason is often that most hazard analysis techniques rely on manual analysis of information recorded in text format. Some of this information would be recorded using different spreadsheet representations, called *hazard identification worksheets*. The most basic description of a worksheets is a table with three columns, where the first column describes a hazard, the second column describes the cause or causes of that hazards, and the third describes the consequences of the hazard. Information in hazard analysis worksheets is usually abundant and overwhelming for anyone attempting a manual analysis. Because of this, there is always a good chance that even expert analysts could accidentally miss important hazards, hazard causes, or scenarios, and therefore draw potentially erroneous conclusions because their reasoning is based on incomplete information. Nevertheless, hazard analysis worksheets are just one example, among others, where manually recorded safety information can get lost.

A number of different Hazard Analysis techniques have been created over the last 50 years, and they are currently broadly used by safety-critical industries. There are different examples of their use in complex systems [Lea10, ZJJ10, Cen10]. There are also examples of adaptations of standard hazard analysis techniques for identifying security hazards [WJG01]. Despite the broad use of the standard hazard analysis techniques, alternative and more recent techniques are also being proposed that are specifically designed for the analysis of hazards in today's software-intensive and socio-technical systems. For example, Nancy Leveson describes an approach to hazard analysis, STPA (System-Theoretic Process Analysis) [Lev11], based on the STAMP causality model. Another example is the Ontological Hazard Analysis (OHA) [Lad05, Lad10] proposed by Ladkin for the analysis and maintenance of safety hazards lists using a refinement approach.

Differently from the efforts cited above, the aim of this thesis is *not* to introduce a novel hazard analysis technique. Rather, in this thesis a method is presented that can be used in conjunction with existing hazard analysis methods, to check the quality of the results, and improve the hazard analysis results when gaps are identified in the results. The proposed method helps analysts to organise the hazard analysis process as a whole. An ontology is used to represent the knowledge acquired, make it more precise and address generic weaknesses in hazard analysis methods. A mechanised process is then used for systematic exploration of the dependencies between hazards and causes of hazards. These dependencies are represented as a transition system, and are analysed mechanically with a formal (i.e., mathematically-based) method tool that facilitates the exploration of all possible hazard path scenarios. The method includes different stages, each designed to give feedback to the analyst, as well as awareness of the hazard analysis performed. As it will be shown in this thesis, this approach can greatly help to improve the results of the hazard analysis performed. This method would therefore be a convenient way of supporting experts during the hazard analysis.

1.2 Problem Statement

The main goal of the research presented in this thesis is *to develop a method that facilitates the resolution of generic well-known weaknesses in the hazard analysis process*. The aim is to facilitate the assessment of hazard analysis results especially at the early stages of system design, as well as improve the results of the hazard analysis. An ontology is used to review preliminary hazard analysis results using a mechanised process based on a set of inference rules. It allows the safety analyst to check well-formedness of hazards, their causes and consequences, as well as to discover new relations among hazards and causes, if these were accidentally omitted in the hazards worksheets. The final result obtained using the method presented in this thesis is therefore an improved version of the initial hazards analysis, with disambiguated hazards and polished causal relationships.

The following sub-objectives are addressed in this thesis.

1. **An approach to modelling hazard analysis results.** An ontology is proposed in order to organise a hazard analysis, specially generated at the early stages of the system design. While organising it the safety analyst would be "forced" to be more precise and therefore a further "guided" study of the hazard analysis would be performed and more specificity would be added into the analysis.

2. **A mechanised analysis method for reasoning about hazards and hazards relations.** The proposed method facilitates the resolution of weaknesses in the hazard identification process. In this method, an ontology is used to add more structure to the hazard analysis, as well as to review hazard analysis results, especially at the early stages, using a a set of inference rules and an automatic tool. The approach allows analysts to check well-formedness of hazards, their causes, and consequences, as well as to discover new relations among hazards and causes, if these were accidentally omitted in the hazards worksheets. The final result obtained using our approach is therefore an improved version of the given hazard analysis, with disambiguated hazards and polished causal relationships.

3. **Generation and analysis of hazard paths.** Hazard paths are a sequence of hazards linked by a causal relationship. The proposed method, and supporting tool, makes it possible to analyse hazards, causes and relationships in a model checker. By this means, hazard paths can be explored in a systematic way, and analysts can check how the initial state of the system could evolve from certain situations to hazards menacing the overall safety of the system. It allows to highlight if some information initially considered irrelevant are in fact important and therefore need to be considered when designing for safety.

1.3 Structure of the Thesis

The thesis is organised as follows.

- Chapter 2 presents basic concepts and definitions that are foundational for the presented thesis work.
- Chapter 3 contrasts and compares related work to the work presented in this thesis.
- Chapter 4 presents the proposed method, including: the phases of the method, the proposed ontology, the technical transformations, and the adopted tools.
- Chapters 6– 10 present an evaluation of the effectiveness of the proposed method by applying it to various realistic case studies in different application areas, including: a medication prescription system (Chapter 6); a generic infusion pump user interface (Chapter 7); a booth-based electronic voting system (Chapter 8); a railway system (Chapter 9); and an avionics protocol for air traffic management (Chapter 10). Each case study offered different analysis challenges, and proved useful for both assessing the proposed method and consolidating its organisation and structure. The ontology structure developed at the beginning of the thesis work has in fact improved significantly thanks to insights obtained from the case studies. The same applies to the proposed reasoning mechanism.
- Finally, in Chapter 11, conclusions are presented, as well as possible future research directions.

2 Basic Concepts and Definitions

In this Chapter foundational concepts and definitions will be presented.

2.1 System

A system is an entity that interacts with other entities, i.e., other systems, including hardware, software, humans, and the physical world with its natural phenomena. These other systems are the environment of the given system. The system boundary is the common frontier between the system and its environment [ALRL04].

The structure of a system is what enables it to generate the behaviour. From a structural viewpoint, a system is composed of a set of components bound together in order to interact, where each component is another system, etc. The recursion stops when a component is considered to be atomic: Any further internal structure cannot be discerned, or is not of interest and can be ignored. Consequently, the total state of a system is the set of the (external) states of its atomic components [ALRL04].

The behaviour of a system is what the system does to implement its function and is described by a sequence of states. The total state of a given system is the set of the following states: computation, communication, stored information, interconnection, and physical condition [ALRL04]. The function of a system is what the system is intended to do.

2.2 Systems Safety

System Safety is a discipline for hazard identification and control to an acceptable level of risk. The fundamental objective of system safety is to identify, eliminate, control, and document system hazards. Safety is a property of a system. It is claimed that it is a different property than reliability. The safety property should be designed into the system.

A traditional definition of Safety by MIL-STD-882E [oDotUSoA12] would be as “freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property, damage to the environment”. Generally, this version is recognised as unrealistic since any system that produces some level of personal, social, technological, scientific, or industrial benefit contains an indispensable element of risk [Lev11]. Nevertheless, the need to ensure safety in a system is absolutely essential. Currently, what safety needs to assure is an acceptable level of risk associated with that system. One possible improvement of the previously presented MIL-STD-882E definition might be that safety "is a measure of the degree of freedom from risk in any environment" [Lev11]

2.3 Hazards

There are several definitions of hazard in the literature. The following three definitions are from two well-known books about safety and hazard analysis and from the Ministry of Defence of the United Kingdom. We present them here since we consider them relevant and they have small differences that might help to get a better insight into what a hazard is.

- A hazard is any real or potential condition that can cause injury, illness, or death to personnel; damage to or loss of a system, equipment or property; or damage to the environment [Eri05].
- A hazard is a situation that could occur during the lifetime of a product, system or plant that has the potential for human injury, damage to property, damage to the environment or economic loss [oD07]. (BS 4778)
- A hazard is a substance, object or situation that can give rise to injure or damage. Risk is the likelihood than an accident or damage of a particular type and severity will occur in a particular period of time or as a result of a particular action or event. A hazard may be serious but the risk from it may be small [Kle01].

In addition, at this point it is relevant to refer to British Standards Institution and its standard BS EN 61508-4 [bs610], there the following definitions are found and they are important to discuss and compare with the definitions given above.

The ISO/IEC Guide 51:1999 Standard [ISO] provides the following definitions:

- Harm: physical injury or damage to the health of people or damage to property or the environment (ISO/IEC Guide 51:1999, definition 3.3).
- Hazard: potential source of harm. The term includes danger to persons arising within a short time scale, for example, fire and explosion, and also those that have a long-term effect on a person's health, for example, release of a toxic substance (ISO/IEC Guide 51:1999, definition 3.5).

The FDA in its document Risk Management in the Design of Medical Device Software Systems [JJITJW02] discusses the current definition of Hazard for the ISO/IEC 14971. This is an ISO standard that represents the requirements for a risk management system of medical devices. In this standard, the definition of hazard is a potential source of harm where harm is the physical injury or damage to the health of people or damage to property or the environment.

FDA [JJITJW02] argues that the ISO/IEC definition of hazard contributes to a great deal of confusion when compiling a list of known or foreseeable hazards. They argue that the definition of a hazard is ambiguous in the sense that almost every state or event within the device system, given certain conditions, could be recognised as a potential source of harm. As a consequence, the distinction between a hazard and the events that lead to a hazard is left to subjectivity. This, finally, has as consequence widely different hazard lists for similar devices between manufacturers.

From these definitions, a hazard is either a source (of harm), or a condition, or a situation. According to the Oxford Dictionary a condition is:

- the state of something with regard to its appearance, quality, or working order: *the wiring is in good condition.*
- a situation that must exist before something else is possible or permitted: *for a member to borrow money, three conditions have to be met.*

A situation is:

- a set of circumstances in which one finds oneself; a state of affairs: *the situation between her and Jake had come to a head.*

And a source is:

- a place, person, or thing from which something originates or can be obtained: *mackerel is a good source of fish oil.*
- (technical) a body or process by which energy or a particular component enters a system: *major sources and sinks exist for atmospheric oxygen.*

Following the definition of the Oxford Dictionary about what is a source, we conclude that, when a source is not a process, these two definitions are synonyms:

- *a hazards is an object or substance that can give rise to injure or damage* [Kle01].
- *a hazard is a potential source of harm.*

A source of harm, either an object or a substance or any other physical matter is something that might appear at certain time during the life time of the system, after the life time of the system, or it was considered to be there even before the system was conceived. We argue that the presence of this matter give a particular property to the state of the whole system.

Following the definitions of the Oxford Dictionary about what is a condition and what is a situation. It is possible to conclude that:

- when a situation is a set of circumstances (according to definition) and we are referring to a system, we could call these circumstances properties of the system at some point of its lifetime or before/after its lifetime.
- when a condition is a situation, the explanation in the previous point applies to it.
- Otherwise, a condition is the state of something with regard to some property. We could call this something a system.

Finally, a hazard might have some different definitions which might create ambiguity but a common characteristic is that a hazard is always in relation with a system and it refers to the state of the system. A hazard then could be described either as a system state or a property of a system state. When a hazard is part of the state of the system, the system is in a hazardous state.

2.4 Ontologies

An Ontology is an explicit specification of a conceptualization [Gru95]. A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose. A conceptualization needs to be done to create a knowledge base, knowledge-based system, or knowledge-level agent. The term is borrowed from philosophy, where an Ontology is a systematic account of Existence. For AI systems, what ”exists” is that which can be represented. When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which a knowledge-based program represents knowledge. Formally, an Ontology is the statement of a logical theory.

Finally, within the context of sharing knowledge, an ontology is a description, of the concepts and relationships that can exist for an agent or a community of agents. This definition is consistent with the use of ontology as a set of concepts, but more general. However, it is surely a different sense of the word than its use in philosophy [Gru95].

2.4.1 Knowledge Representation and Inference Mechanisms

Building ontologies for large domains, such as medicine or arts, is a costly affair [WSWS01]. However, in many domains thesauri have been built that can be a basis for the construction of an ontology. In the ontology construction process additional knowledge should be added to the basic hierarchical structure of concepts derived from the thesaurus. This knowledge can come from different sources: the location of a concept in the hierarchy, additional sources such as Wordnet [Uni13], or special purpose documents.

The implementation of an Ontology needs a knowledge representation formalism and an inference mechanism [CGp00]. The domain knowledge describes the main static information and knowledge objects in an application domain. Knowledge ontologies can be specified using five kind of components: concepts (classes), relations, functions, axioms and instances. The inference mechanism describes how the static structures represented in the domain knowledge can be used to carry out a reasoning process. There is a strong relationship between both, since the structures for representing knowledge are the basis for the reasoning process.

Combining ontologies with rule languages are useful for supporting the expansion of knowledge that is contained in a knowledge base. In health care, there is a significant number of applications based in a declarative approach and rule-based systems. These applications are aimed to help decision making disease diagnosis and treatment.

2.4.2 OWL, Protégé and SWRL

The Ontology Web Language (OWL) is an ontology description language recommended by the W3C [COP]. OWL is a standard for knowledge representation when the goal is to provide interoperability between software systems. The current version known also as OWL 2 [GHM+08] is an evolution of the original standard proposed in 2004. OWL is used widely, for various reasons but mainly because there are a considerable number of tools to support it.

Protégé [GMF+03] is a software tool designed to build and edit knowledge bases. It is considered now the *de-facto* standard for ontology design. It was originally thought as an aid tool for aiding knowledge acquisition in some medical domain applications. It has evolved to a complete suite with capabilities to support several formats and languages for knowledge representation and visualization. Protégé-OWL [COP] is an extension of Protégé that provides support for OWL. One of the most compelling features of Protégé-OWL is its easy access to reasoners such as Racer [HM03], Pellet [SPG+07, Uni], HermiT [Her] and others.

The Semantic Web Rule Language (SWRL) [COP] adds to the OWL language with inference capabilities that go beyond the classification capabilities expressed by description logics and OWL. Similarly to a lot of other rule languages, SWRL rules are written as antecedent consequent pairs. In SWRL terminology, the antecedent is referred to as the rule body and the consequent is referred to as the head. The head and body consist of a conjunction of one or more atoms.

2.5 Model checking

In this section we will talk about Model Checking, which is a formal verification technique used for verifying various types of models representing complex reactive system.

2.5.1 Definition

Model checking [CGP99] is a technique for automatic verification of finite concurrent systems such as communication protocols and hardware circuits. It has many advantages over other approaches, it is fast, produces counterexamples and handles partial specifications.

The approach consists of three main tasks.

1. *Modeling* converts an algorithm into a formalism accepted by a model checker.
2. *Specification* states the properties that the algorithm must satisfy, this is given in some logical formalism.
3. *Verification* examines if the model satisfy the specification. It is automatic but it also usually involves human interaction, such as error tracing and result analysis.

2.5.2 Kinds of Model checking

Model checking is an automated verification method that systematically explores all possible behaviour of a system under examination. There are different Model Checking techniques, bellow the most common:

- *Temporal model checking* is based on Kripke structures $M = (S, R, L)$ that represents a concurrent system. Given a temporal formula φ expressing some desired specification, the goal is to find all states S that satisfy φ: $\{s \in S : M, s \vDash \varphi\}$. The system fulfils the requirements if all initial states are in this set. Linear temporal logic (LTL) and computation tree logic (CTL) are logical formalisms useful for specifying the systems in this setting.
- *Symbolic model checking* [BCM$^+$90] was proposed to address the state explosion problem that was present in explicit representations. Symbolic representation for the state transition system based on manipulation of Boolean formulas is an approach to solve the problem posed by large state spaces. Ordered binary decision diagrams (OBDDs) are a canonical form representation of these Boolean formulas. They are directed acyclic graphs obtained by optimising binary decision trees which represent these binary formulas. OBDDs are useful for obtaining representations of relations over finite domains and can be used to represent and analyse Kripke structures.
- *Bounded model checking* [PL03] technique reduces the problem of model checking to a problem of satisfiability for a Boolean formula (SAT). A state transition system is encoded as a propositional formula, which is satisfiable if and only if there exists a counterexample of bounded length k. The formula is given to a satisfiability solver and if the formula cannot be satisfied at length k the search is continued for larger k. SAT solvers do not

require exponential space and large designs can be checked very fast, however the method generally lacks completeness and properties that can be verified are limited.

- *Abstraction* [CGL94] is another method for reducing the state space explosion problem. The model checker builds a simplified model of a system if the actual model is too complex. It then can verify a property of the smaller model if the property is preserved in the original one.

2.5.3 Temporal Logics

Temporal logics are different types of logics where the modality is time. Below, we will focus on the main ones from the Model Checking perspective: LTL and CTL.

LTL

Linear Temporal Logic (LTL) is a modal logic, which is widely used to express specifications to verify models using model checkers. The elementary temporal modalities that are present in most temporal logics include the operators:

$\Diamond$ eventually in the future$F$ $\Box$from now and always in the future G

Until U and next X are some other operators. U is a binary operator ($\rho U \psi$) it represents a situation where "ρ holds continuously Until ψ becomes true(at least once)". $X\rho$ represents a situation where "ρ holds at the next instant time"

The syntax of a LTL formula φ is given as

$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid X\varphi \mid G\varphi \mid \varphi U \varphi$

with the convention that $F\varphi ::= \neg G \neg \varphi$

$$\begin{array}{ll}
(M,\rho) \models p & \textit{iff } s_0 \in \pi(p) \\
(M,\rho) \models X\varphi & \textit{iff } (M,\rho^1) \models \varphi \\
(M,\rho) \models G\varphi & \textit{iff } \text{ for each } i \geq 0 \text{ we have } (M,\rho^i) \models \varphi \\
(M,\rho) \models \varphi U \psi & \textit{iff } \exists_{j\geq 0} \text{ such that} \\
& \quad (M,\rho^j) \models \psi \wedge \\
& \quad (M,\rho^k) \models \varphi \forall 0 \leq k < j
\end{array}$$

CTL

Computation tree logic (CTL) is a branching-time logic that allows to express properties about execution paths of a system. The syntax of a CTL formula φ is given as

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid EX\varphi \mid AG\varphi \mid E(\varphi U \varphi)$$

The semantics is defined inductively, where π_s denotes all runs starting from a set of states s. We say that a system M with s is a *model* of formula φ (given as $(M, s) \models \varphi$) if:

$$\begin{array}{ll}
(M,s) \models p & \textit{iff } p \in AP(s) \\
(M,s) \models \neg\varphi & \textit{iff } (M,s) \not\models \varphi \\
(M,s) \models \varphi_1 \wedge \varphi_2 & \textit{iff } (M,s) \models \varphi_1 \texttt{ and } (M,s) \models \varphi_2 \\
(M,s) \models EX\varphi & \textit{iff } \exists_{\pi\in\pi_s} : (M,\pi[1]) \models \varphi \\
(M,s) \models AG\varphi & \textit{iff } \forall_{\pi\in\pi_s}\forall_{i\geq 0} : (M,\pi[i]) \models \varphi \\
(M,s) \models E(\varphi U \psi) & \textit{iff } \exists_{\pi\in\pi_s}\exists_{k\geq 0} : (M,\pi[k]) \models \psi \wedge \\
 & \quad \forall_{j<k}(M,\pi(j)) \models \varphi
\end{array}$$

Additional operators can be constructed by combination of the ones given above (e.g., $EF\varphi := \neg AG\neg\varphi$, $\varphi \rightarrow \psi := \neg\varphi \vee \psi$). Intuitively, $X\varphi$, $G\varphi$, $F\varphi$, are path formulas that hold if φ evaluates to true in the next state, in all states, or eventually in some state of the path. Similarly, $\varphi U \psi$ holds if φ holds until ψ holds. The prefixes to path formulas A and E denote that a formula holds in a state if the formula holds for all paths (A) or at least one path (E) starting from the current state. A system M satisfies a formula φ if $(M, I) \models \varphi$.

Given a CTL formula, a model checker computes the states in which the formula holds. This can be done effectively using BDDs [BCM$^+$90] as data structure to store states and transition relation.

2.5.4 MCMAS Model Checkers

There are many model checkers for different purposes currently under development. In this subsection we provide a short review of a leading model checker, MCMAS (Model Cheker for Multi-Agent Systems).

MCMAS [LQR09] is an OBDD based symbolic model checker specifically designed for multi-agent systems. It supports a number of modalities including CTL, ATL, and epistemic operators. The input language of MCMAS is Interpreted Systems Programming Language (ISPL), which has a rich specification capability to describe agents and multi-agent systems. Interpreted Systems [CGP99] provide the formal semantics of ISPL programs.

MCMAS is an open-source application implemented in C/C++ and relies the CUDD library [Som12] for BDD operations.

2.6 Hazard Analysis

Hazard Analysis is the estimation of frequency and consequences of hazards, and the comparison with a criterion and a decision on action over them [Kle01]. On the other hand, according to [Eri05], Hazard Analysis is the group of methods performed to identify: hazards, hazard effects and hazard casual factor and this analysis could be classified in two categories of Hazard Analysis: types and techniques.

The colloquially known Yellow Book 2 (YB2) [Rai01a] coins the acronym HIA that stands for Hazard Identification and Analysis. HIA consists of three phases: Hazard Identification; Consequence Analysis; and Causal Analysis. Therefore, according to YB2, Hazard Analysis consists of, firstly, an analysis of the consequences or chains of cause and effect that could be from hazards to accidents and it should be done to a level where the probability of accident occurrence could be assessed, and secondly, an analysis of chains of cause and effect of failure modes in a system which could lead to hazards and deeply enough to allow the probability assessment.

2.7 Effectiveness of Hazard Analysis Techniques

A safety Engineering team is never completely sure about the completeness and perfection of their analysis. In order to measure an approximate of their effectiveness some techniques are available [PRCF06] [SK89]. Using more than one hazard analysis technique for the analysis of a system is common practice. Moreover, subsequent analysis tools are used and are needed. In addition, a good understanding of the system is achieved by knowledge about previous mishaps in similar systems as well as previous knowledge about previous hazard analysis about the system. The way that data about the different hazard analysis is recorded and updated as well as documentation about the recommendations of the analysis done are therefore key. The expert knowledge of a safety engineer is often required, but it is not straightforward to measure how important is this knowledge regarding the effectiveness of the hazard analysis technique.

2.7.1 Methods to measure effectiveness

In [CC09], Cantrel et al. argue that to determine the effectiveness of a method for finding all hazards, the total hazard population must be discovered first. However, if such a method existed there would not be the need for other methods, since this could be used universally. Using this reasoning they conclude that either:

- it is possible to find all the hazards in a system but there is no way to demonstrate this, or
- it is impossible to learn the proportion of all the hazards that have been identified.

Despite the fact that people is unable to determine the proportion of hazards identified by a hazard analysis, the question “How do we know when we are done?” remains to be answered.

There are different experimental ways to measure the effectiveness of hazard analysis techniques, as well as the effectiveness of hazard identification processes. In [PRCF06], Guerra et al. make a direct comparison between two different hazard sets identified for the same system applying the same hazard analysis technique in both cases. Then they apply Capture-Recapture (CR) analysis technique [Bar02] to measure the effectiveness of the hazard identification process that produced the two hazard sets. CR proposes the following measure of the detection efficiency:

$$E_i = \frac{N_{ij}}{N_j}$$

where N_i and N_j are the number of hazards found in the two hazard identifications, and N_{ij} is the number of common hazards.

On the other hand, in [SK89] Soukas et al gives a method that measures the effectiveness of a hazard analysis technique. The first step is to get a first number of hazards founded by applying different times the hazard analysis technique that it is going to be evaluated as a primary technique. For example, if a hazard analysis technique X is under evaluation and lets say it has identified H_e hazards. In addition to that, by successive use of supplemental hazard analysis methods other hazards can be identified and even later hazards found during early operation of the system cam be added, this number of hazards accumulated to H_e will be H_s. Then, the hazard identification effectiveness (HIE) is as follow:

$$(HIE)\% = (\frac{H_e}{H_s}) \times 100$$

This method provides a relative measure of HIE for the primary method that had produced the H_e result.

There are some others studies comparing hazard analysis techniques, for example [RvdB02], where Expert Analysis, FMEA, FTA, among others are compared in terms of information needed for the analysis, actions performed during the analysis, results obtained, for example documentation.

Finally, there is no single hazard analysis technique capable of finding all the hazards in a system, and no method to verify all the hazards have been identified [CC09]. It is also unclear which analysis technique is the most suited in which situation [RvdB02]. In general, only one hazard analysis type does not identify all the hazards within a system, and more than one type is required [JGOD02, Eri05]. For these reasons, several techniques are often combined in practice and their application requires experience to obtain solid results.

2.8 Identifying Hazards

System safety ideally aims to develop a system free of hazards and, as it is discussed in previous section 2.7.1, it is generally not possible to eliminate all hazards, even less when the system is complex, the objective then becomes identify potential hazards, assess their risk, and implement corrective actions to eliminate or mitigate the identified hazards [Eri05].

A key process in hazard analysis techniques is to identify hazards [Kle01], [Rai01a] and [Eri05]. When, in this thesis, the term *hazard identification* is mentioned it refers only to the process of describing a hazard, causes and consequences, and the initial identification of ways or scenarios where the system design or operation can lead to an accident.

In spite that there is no hazard analysis technique that discovers all the hazards in a system under study, an exhaustive hazard identification process is always needed because it is pointless to quantify some hazards when the larger ones have been missed, consequently, the biggest errors in Hazard Analysis originate in the failure to foresee all the causes of hazards or all the hazards that can arise [Kle01].

2.9 Discussion

There is not perfect hazard analysis technique. For example, although HAZOP has been widely used as a hazard analysis technique it has been criticised for putting too much emphasis on hardware failure while ignoring the operation related hazards [Saf10]. Studies have shown that the most significant flaws in hazard analyses are often errors of omission during the identification of hazards and hazard causes [Har10].

A interesting study about accidents and the identification of hazards [Har10] give some recommendations about what fails when identifying hazards. The most important are considered below:

- Sometimes hazards are too generic.
- The boundaries of the hazard analysis for the system are too tight.
- There might be a common cause to a number of hazards but it gets lost during the hazard analysis process.
- As the development cycle proceeds new hazards are uncovered and some hazards may no longer be relevant.
- Many hazard analysis ignore human error or consider it in overly simplistic terms. Which it is insufficient as humans operate in complex ways, and resulting accidents reflect that complexity. In addition, many times accidents do not occur because of one single human error but rather because of a series of small, seemingly insignificant errors that, when added together, result in catastrophic consequences. Human error must be treated systematically.
- Failure to consider hazards related to poor design processes.

Finally, there is room for improvement and automation of the current hazard analysis techniques and it needs to be addressed. In order to help in the process of identifying hazards one might look at previous analyses on the system to be analysed or past experiences on similar systems. In addition, the use of multiple tools and techniques are necessary for hazard identification and proper use of previous knowledge could contribute to help the safety analyst into get more insight about the systems and its hazards. This thesis's proposal could be seen as a help to improve the results of these hazard analysis techniques and as a help for further analysis.

3 Related Work

In this chapter, the intention is to show broadly used hazard analysis techniques that rely on worksheets as one of the means for performing the hazard analysis as well as to communicate with different stakeholders. In addition, it is also important to show that some of these hazards analyses count with international standards, and also even when they do not count with a particular standard, like Preliminary Hazard analysis, they are recomended in international safety standards. In the case of Bowtie Analysis, this is also included because it is a powerful tool to show scenarios where a top event is analysed and it shows how some events or causes might develop intothe top event (or hazard) and it also shows the consequences of that top event, in addition, this is visualized.

In addition, we aim to show new proposed hazard analysis techniques, STPA is shown not only because currently it has created wide interest in the safety academy and industry but also because they do propose a graphical notation only for the purpose of the hazard analysis. OHA is considered because it uses an ontological basis in order to produce safety requirements as well as the inclusion of HAZOP as part of the analysis and that intends to be evaluated with formal methods techniques. Moreover, the author discusses the different analyses provided.

Moreover, this chapter also intends to cover related research done on the use of ontologies in hazard analysis, this includes how hazards and hazard analysis have been represented in ontologies, and also the usefulness of ontologies in the hazard analysis process. A discussion about the role of uncertainty in hazard analysis is provided, as it tackles criticism to broadly used analysis techniques. Also, we discuss the extend to which AI could be applied to the hazard analysis domain in order to automate them. Finally, the challenges of reasoning about hazards and the importance of humans in hazard analysis is discussed.

Section 3.1 covers the widely used FTA, FMEA and HAZOP hazard analyses, Section 3.1.6 covers briefly, the standards used in safety analysis, Section 3.2 covers recent techniques STPA and OHA. Section 3.3 provides a discussion about the limitations of mentioned hazard analysis techniques.

Section 3.4 covers research done with regards the contribution that ontologies can have to the safety industry and also to the aumotamic generation of system models that are used in the safety analysis process. This contribution involves the representation of hazards(seen in Subsection 3.4.1), the representation of hazard analysis techniques (seen in Subsection 3.4.2) and research advancing of automation in the creation of system models for safety analysis purposes (Subsection 3.4.3).

Section 3.5 is a discussion about how ontologies can help the safety analyst to provide a better description of a hazard and also provide more detailed information with regards the location of the hazard within the system which can be useful in following steps of the hazard analysis process (Subsection 3.5.1).

Section 3.6 is a discussion about how uncertainty is dealt with in traditional hazard analysis techniques, Section 3.7 is a discussion about the automation of hazard analysis techniques using Artificial Inteligence (AI), the challenges that this automation faces and the role of humans within the hazard analysis domain and within the automation of hazard analysis techniques.

3.1 Traditional Hazard Analysis Techniques

Hazard Analysis techniques define a unique analysis methodology that is performed following a specific set of rules and provides specific results. There exists over 100 different hazard analysis techniques [Eri05].

The following list contains the most popular ones:

- Preliminary Hazard List (PHL)
- Fault Tree Analysis (FTA)
- Failure Modes and Effects Analysis (FMEA)
- HAZOP (Hazard and Operability Studies).

Other common techniques include [Eri05]:

Preliminary Hazard Analysis (PHA); Subsystem Hazard Analysis (SSHA); System Hazard Analysis (SHA); Operating and Support Hazard Analysis (O & SHA); Health Hazard Assessment (HHA); Safety Requirements/Criteria Analysis (SRCA); Event Tree Analysis (ETA); Fault Hazard Analysis; Functional Hazard Analysis; Sneak Circuit Analysis (SCA); Petri Net Analysis (PNA); Markov Analysis (MA); Barrier Analysis (BA); Bent Pin Analysis (BPA); Cause Consequence Analysis (CCA); Common Cause Failure Analysis (CCFA); MORT Analysis; and Software Safety Assessment (SWSA).

3.1.1 HAZOPS

The HAZOP (Hazard and Operability Studies) analysis was established as a hazard analysis technique by the Institute of Chemical Industry (ICI) in the United Kingdom in the 1970s. It was used for the safety analysis of chemical process plants. Shortly, HAZOP became more widely used within the chemical process industry after the Flixborough disaster, this was an explosion at a chemical plant close to the village of Flixborough, North Lincolnshire, England, on 1 June

1974. It killed 28 people and seriously injured 36 out of a total of only 72 people on site. The HAZOP analysis was later adopted by the petroleum industry and other industries [Kle01].

HAZOPs is a technique for hazard identification (or analysis) of a system. It can be applied from the concept phase through decommissioning. HAZOPs can be applied very early in design phase and identify safety concerns early in the design process. The HAZOP analysis uses key guide words and system diagrams (design representations) to identify system hazards. The guide words could be adjectives such as more, no, less. These words are combined with process/system conditions such as speed, flow, pressure. From these combinations some deviations would be found. The potential system deviations then lead to possible system hazards. A HAZOP analysis is performed by a team of multidisciplinary experts in a brainstorming session under the leadership of a HAZOP team leader. The key to a successful HAZOP is the selection of the right team leader and the selection of the appropriate team members. The HAZOP analysis is applied in a structured way by the team, and it relies upon their imagination in an effort to discover credible causes of deviations from design intent. The HAZOP analysis is applicable to all types of systems and equipment, with analysis coverage given to subsystems, assemblies, components, software, procedures, environment, and human error. HAZOP analysis can be conducted at different abstraction levels, such as conceptual design, top-level design, and detailed component design [Kle01, Vin14, Eri05].

Current international standard for HAZOP is *BS EN 61882:2016 Hazard and operability studies (HAZOP studies). Application guide* [bs616].

HAZOP Worksheets

To perform HAZOP analysis. It is advisable to use a specialized worksheet. Typically, columnar-type worksheets are most used. As a minimum, the following basic information is required from the HAZOP analysis worksheet:

- Guide word
- Deviation
- Possible Causes
- Consequences
- Actions required

Table 3.1 shows an example of a HAZOP worksheet [Kle01].

3.1.2 Fault Tree Analysis

The FTA (Fault Tree Analysis) technique was created in the Bell Labs. Shortly after, FTA started to be used in the commercial aircraft and the nuclear power industries. Fault tree

Table 3.1: Results of hazards and operability study of olefin dimerisation unit (snippet).

Guide Word	**Deviation**	**Possible Causes**	**Consequences**	**Action Required**
NONE	No flow	(1) No hydrocarbon available at intermediate storage.	Loss of feed to reaction section and reduced output. Polymer formed in heat exchanger under no flow contidions.	(a) Ensure good communication with intermediate storage operator.

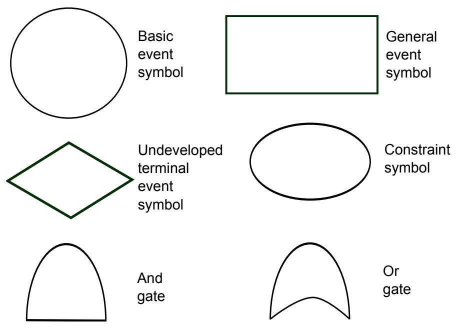


Figure 3.1: Fault Tree Symbols (Based on figure from [Ste12])

analysis is a well structured methodology requiring the application of some particular rules of Boolean algebra, logic, and probability theory. There are two basic approaches to FTA, the qualitative approach uses deductive logic to determine how an undesired top event could occur. The quantitative approach adds reliability or probability to the analysis. Current standard for FTA is *BS EN 61025:2007 Fault tree analysis (FTA)* [bs607].

The Fault Tree (FT) is a logic diagram where all the events that can cause the top undesired event to occur are drawn. Those events that must occur in order to the top event to happen are represented by an AND gate and those events that can occur in order to the top event to happen are represented by an OR gate. Once the FT is finished, a new evaluation will follow in order to define the critical cut sets and probability of failure. Cut sets are different combinations of failure events that can prompt the top event to occur. The FT demonstrates how specific events will cause an outcome. In addition, if there is known probability data for these events then it is possible to calculate the likelihood of reaching the top event and therefore the FTA can be used to support risk management decisions.

Symbols

Now, we will explain the basic symbols used to chart a FT. See Figure 3.1

1. The rectangles is used to identify the top event as well as secondary events. All that appears under a rectangle must be analysed on lower levels.

2. The Circle represents a basic or root event, the first to have occurred and which does not need further analysis.

3. The Diamond represents an event that has not been developed either because of lack of information or because of the complexity of the event. Usually this kind of events are indicated for further development in the future.

4. The Oval represents a conditional event or a conditional input. It may represent a restriction on the occurrence of the event based on the occurrence of other events on the causal chain.

5. Logic gates, the logic gates represented are the AND gate and the OR gate. The AND gate means that all the contributing events to the primary event, through the gate, must occur in order to the primary event to occur. The OR gate means that if any of the events connected to a primary event through an OR gate occurs then the main event will also occur.

3.1.3 Failure Mode and Effects Analysis

The Failure Mode and Effects Analysis (FMEA) was developed for the U.S. military as a formal analysis technique. It was initially used as a reliability evaluation technique to determine the effect of system and equipment failures. Failures were classified according to their impact on mission success and personnel/equipment safety. In the late 1970s, Ford Motor Company reintroduced FMEA. The FMEA technique is a reliability tool, it analyses potential failure modes and calculates subsystem, assembly, or unit failure rates. The evaluation of the severity and probability of failure modes generates a prioritized list for corrective actions. The FMEA technique can be used to identify hazards. However, FMEA focus only on single component failure modes but hazards can be the result of multiple hazards and events other than failure modes. Therefore, FMEA is not recommended as a single tool for hazard identification but together with other hazard analysis techniques. The basic concepts in FMEA are failure, failure mode, failure cause and failure effect [Vin14, Eri05, Ste12]. Current standard for FMEA is *BS EN IEC 60812:2018 Failure modes and effects analysis (FMEA and FMECA)* [bs618].

1. Failure is when an item moves from its intended operation, function, or behavior. The inability of a system, subsystem, or component to perform its required function.

2. Failure mode is either the form by which an item fails or the form in which the item is after it fails.

3. Failure cause is the mechanism responsible for the failure mode.

4. Failure effect is the consequence(s) that a failure mode has on the operation, function, or status of an item and on the system.

Table 3.2: FMEA worksheet 3-safety/reliability (based on [Eri05]).

Failure Mode and Effects Analysis										
System:				Subsystem:			Mode/Phase:			
Item	Failure Mode	Failure Rate	Causal Factors	Immed. Effect	System Effect	Method of Detection	Current Controls	Hazard	Risk	Rec. Action
...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...

FMEA Worksheets

It is advisable to perform the FMEA using a worksheet. Typically, columnar forms or text-type forms are used. An FMEA that supports system safety and hazard analysis should contain the following information, as a minimum:

1. Failure mode
2. System effect of failure mode
3. System-level hazards resulting from failure
4. Mishap effect of hazards
5. Failure mode and/or hazard causal factors
6. How the failure mode can be detected
7. Recommendations (such as safety requirements/guidelines that can be applied)
8. The risk presented by the identified hazard

Table 3.2 shows an example of a very basic FMEA worksheet format, primarily for use by the reliability organization.

3.1.4 Preliminary Hazard Analysis

The Preliminary Hazard Analysis (PHA) technique is a safety analysis tool for identifying hazards, their causes, consequences, risks and mitigations when detailed design information is not available. The PHA is probably the most commonly performed hazard analysis technique. In most cases, the PHA identifies the majority of the system hazards. The remaining hazards are usually uncovered when subsequent hazard analyses are generated and more design details are available. Subsequent hazard analyses refine the hazard cause effect relationship and uncover previously unidentified hazards and refine the design safety requirements [Eri05].

Table 3.3: PHA worksheet (based on [Eri05]).

Preliminary Hazard Analysis									
System:			Subsystem:			Analyst:		Date:	
No.	Hazard	Causes	Effects	Mode	IMRI	Recommended Action	FMRI	Comments	Status
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...

The PHA might start from the PHL (Preliminary Hazard List). Then, the initial collection from PHL is evaluated in more detail. Moreover, the hazard checklists are compared with the design knowledge to identify unforeseen hazards, this way, the analyst identify possible hazards. In order to perform the PHA, the system safety analyst should have a basic understanding of the system design, hazards, hazard sources. Knowledge about hazard comes primarily from hazard checklists and from lessons learned on the same or similar systems. Some techniques can be used instead of PHA however is not recommended. For example, a modified failure mode and effects analysis (FMEA) could be used as a PHA, but this is not recommended since the FMEA primarily looks at failure modes only, while the PHA considers many more system aspects [Vin14, Eri05]

PHA Worksheets

It is recommended to perform the PHA using a specialized worksheet. The PHA is a detailed hazard analysis utilizing structure and rigor. It is desirable to perform the PHA using a specialized worksheet. There is not strict format for the PHA worksheets. It is recommended that the PHA generate the following information:

1. System hazards
2. Hazard effects, comsequences
3. Hazard causal factors (or potential causal factor areas)
4. Risk assessment (before and after design safety features are implemented)
5. Safety critical functions and top level accidents
6. Recommendations for eliminating or mitigating the hazards

Table 3.3 shows the columnar format a basic PHA worksheet.

3.1.5 Bow Tie Analysis

A Bow Tie is a diagram that visualises the risk of a determined organisation (operation, system) within one, generally simple but not small, diagram. The diagram, as the name suggests, is shaped like a bow-tie. A hazard and a top event are placed at the centre of the bow-tie. On the extreme left hand side of the bow-tie, there are the causes of the top event, and on the extreme right hand side of the top event, there are the consequences of the top event. Between the causes and the top event there are barriers which aim to prevent the top event to happen. Between the top event and the consequences there are barriers that, once the top event is activated, aim to protect the organisation (or system) from the consequences to happen. In a Bow Tie diagram there are also escalating factors which cause a barrier to fail. It is worth noting that the Bow Tie terminology differs slightly from Safety Management Systems, for example the term “Hazard” [Aut14]. A hazard in the Bow Tie method describes the potential source of harm under consideration. It will often describe a “normal” aspect within the operating environment and sets the context and scope of the Bow Tie, for example driving a car on a busy motorway, this is an activity where risks are present [Aut14]. Then, the Top Event describes the point where there is no longer adequate control over the hazard. It is usually an unsafe event that is not yet an accident.

The Bow Tie analysis tool is a qualitative approach, it is particularly appropriate for complex environments and/or organizations, industries, where it is practically impossible to quantify all the interactions between people, equipment, time, weather and many other different factors. An example of such an organisation is aviation [Dev14].

The Bow Tie analysis does not require detailed and systematic information of the system or organization in the Bow Tie diagram. However, the description of different Bow Tie diagrams, specially the most “complicated”, show a vast and deep knowledge of the organisations where the Bow Tie diagrams are done. This is because the team members need to define the context and scope of the Bow Tie diagram with subject experts and stake holders [Dev14].

3.1.6 Standards

Succesful safety programs need that the complete organization holds a real commitment to a safety culture. For example SUBSAFE, which is a quality assurance program of the United States Navy created to assure the safety of the nuclear submarine fleet; the certification focuses on structure, systems, and components that are critical to the watertight integrity and recovery of the submarines. Since it was created in 1963, there has not been any submarine lost in the USA Navy. In 1963, a SUBSAFE certification boundary was defined. Certification focuses on the structures, systems, and components that are critical to the watertight integrity and recovery of the submarine [Lev11]. Different other organisation such as the U.S. Occupational Safety and Health Administration(OSHA) [oL20], U.S. Food and Drug Administration (FDA) [FA], NHS (National Health Service), Transport organizations, NASA, etc., require for the development of different projects and systems the development of Safety Programs and Safety Analysis, this Safety Analysis includes the use of different Hazard Identification and Hazard Analysis techniques. For example, OSHA is part of the U.S. Department of Labor and it was created by the U.S. Congress to assure safe and healthful working conditions for working men and women by setting and enforcing standards and by providing training, outreach, education and assistance. OSHA standards are rules that describe the methods that employers must use to protect their employees from hazards. There are OSHA standards for Construction work, Agriculture, Maritime operations, and General Industry, which apply to most worksites. These standards limit the amount of hazardous chemicals workers can be exposed to, require the use of certain safe practices and equipment, and require employers to monitor hazards and to keep records of workplace injuries and illnesses. The guidance, which is published online, includes the Process Hazard Analysis. The process hazard analysis is a thorough, orderly, systematic approach for identifying, evaluating, and controlling the hazards of processes involving highly hazardous chemicals. An employer must perform an initial process hazard analysis (hazard evaluation) on all processes covered by this standard. The process hazard analysis methodology selected must be appropriate to the complexity of the process and must identify, evaluate, and control the hazards involved in the process. The employer must use one or more of the following methods, as appropriate, to determine and evaluate the hazards of the process being analysed [oL20]:

- What-if,
- Checklist,
- What-if/checklist,
- Hazard and operability study (HAZOP),
- Failure mode and effects analysis (FMEA),
- Fault tree analysis, or
- An appropriate equivalent methodology.

Additionally, OSHA promotes the use of the Job Hazard Analysis which is a technique that focuses on job tasks as a way to identify hazards before they occur. It focuses on the relationship between the worker, the task, the tools, and the work environment.

Likewise, another example, the U.S. Department of Transportation (USDT) has also a guideline for the purpose of the projects realised within their ambit: Hazard Analysis Guidelines For Transit Projects. In these guidelines it is written that a Preliminary Hazard Analysis (PHA), a Failure Modes and Effects Analysis (FMEA), and Operating Hazard Analysis (OHA) shall be performed in the Transport Projects presented to them. Meanwhile both different purposes administrative offices, OSHA focuses on the safety of general industry workers, especially construction, maritime and agricultural workers. USDT focuses on transportation. While OSHA and USDT have different focus, they both require the use of Hazard Analysis. However, despite of the differences, same Hazard Identification or Hazard Analysis techniques are recommended. What seems clear is that some very well-known techniques, such as PHA, HAZOP, and FTA are techniques that can be applied in projects with different system specifications. There are also hazard analysis techniques like JHA that are totally created and focused on a purpose. Different ISOs have been created for different safety processes. ISO 14971 (2007) does not impose the use of any specific risk analysis technique. It references and describes five of them for informative purposes: PHA, FTA, FMEA, HAZOP and HCCP. It recommends that more than one be used to take advantage of their respective strengths.

3.2 Recent Hazard Analysis Techniques

3.2.1 STPA

STPA (System Theoretic Process Analysis) is a new hazard analysis technique based on the STAMP causality model, which is an accident investigation method. They both make use of concepts of systems theory on their conception. STPA has its own definition of hazard. It defines hazard as a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident (loss). The creators of STPA, stress out that the word failure is not mentioned in STPA. Hazards are not identical to failures -failures can occur without resulting in a hazard and hazards may occur without any precipitating failures. STPA starts by identifying a short list of high-level system hazards. Once the system level hazards have been identified the next step is to identify the system safety requirements and constraints. This is per each high-level system hazard. The constraints are design constraints necessary to prevent the hazards from occurring. The last step is to identify the safety control structure of the system under study. When STPA is applied to an existing design, this information is available when the analysis process begins. When STPA is used for safety-guided design, only the system -level requirements and constraints may be available at the beginning of the process. In the latter case, these requirements and constraints are refined and traced to individual system components as the iterative design and analysis process proceeds [Lev11, Lev13]. STPA uses the following process for the identification of the causal factors of hazards in system design:

1. Represent the system under analysis as a control model.

2. Using a set of guidewords similar to those used in HAZOP (not provided, too early, etc., see Table 3.4).

3. Use the identified unsafe control actions to identify an initial set of safety requirements and constraints.

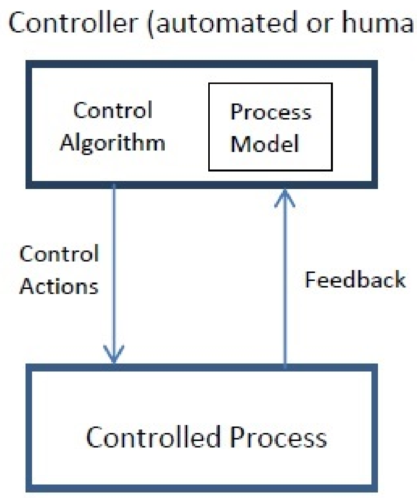


Figure 3.2: Basic Control Structure (Taken from [Lev13])

4. Use the STPA causal factors model (see Figure 3.3) to identify the causes in system design of possible unsafe control actions. Each control action will be analysed in order to identify causes of the unsafe control action. More safety requirements are also identified at this step.

Identifying Unsafe Control Actions

STPA recognises four types of unsafe control actions:

- A control action required for safety is not provided.
- An unsafe control action is provided that leads to a hazard.
- A potentially safe control action provided too late, too early, or out of sequence.
- A safe control action is stopped too soon or applied too long (for a continuous or non-discrete control action).

STPA creators have identified that a table is a convenient way to document the specific unsafe control actions but any format could be used. The general form of the table that they use is in 3.4:

The entries in the table can then be the basis for the creation of the initial safety constraints(requirements).

Identifying the Causes of the Unsafe Control Actions

The identification of potential causes in system design of unsafe control actions is usually performed by a group of analysts. In this step, the authors suggest that the scenario where an

Table 3.4: STPA table

Control Action	Not providing causes hazard	Providing causes hazard	Too early / too late, wrong order causes hazard	Stopping too soon/ applying too long causes hazard

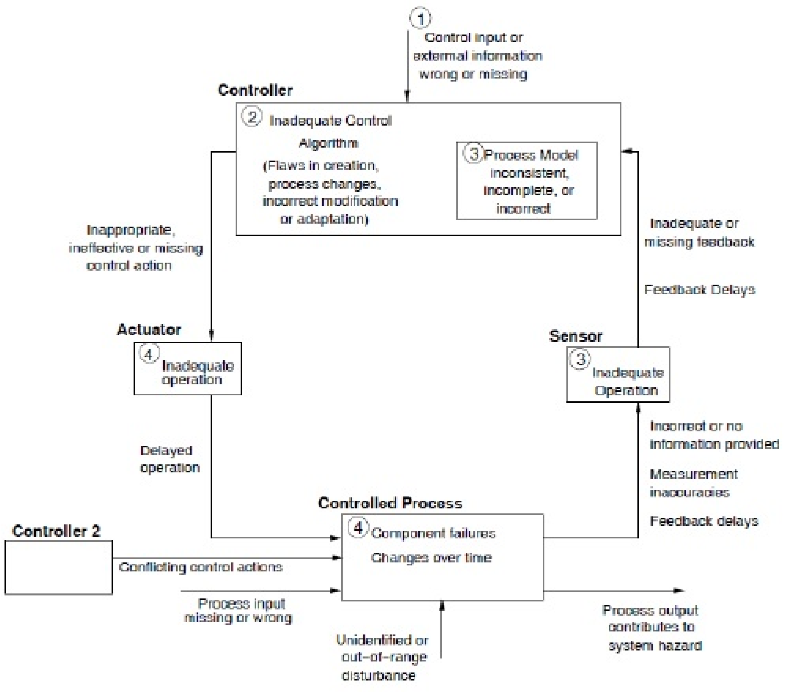

1 Control input or external information wrong or missing
Controller
2 Inadequate Control Algorithm (Flaws in creation, process changes, incorrect modification or adaptation)
3 Process Model inconsistent, incomplete, or incorrect
Inappropriate, ineffective or missing control action
Inadequate or missing feedback
Feedback Delays
Actuator
4 Inadequate operation
Sensor
3 Inadequate Operation
Delayed operation
Incorrect or no information provided
Measurement inaccuracies
Feedback delays
Controlled Process
4 Component failures
Changes over time
Controller 2
Conflicting control actions
Process input missing or wrong
Unidentified or out-of-range disturbance
Process output contributes to system hazard

Figure 3.3: Control Loop (Taken from [Lev13])

"inadequate execution of a control action required for safety" should be analysed. Also in this step, additional safety requirements are hoped to be identified. In addition, more information should be generated here about how to eliminate or mitigate the potential causes of the hazards. In order to apply Step 2, the analyst needs to create and examine the causal factors error model (Figure 3.3). This model guides the analysts through the exploration of three broad types of design issues: (i) Feedback is inadequate, missing, or delayed; (ii) Control algorithm is inconsistent, incomplete, or incorrect; and (iii) Wrong or missing inputs / external information.

3.2.2 OHA

Ontological Hazard Analysis (OHA) [Lad05] is a recent Hazard Analysis technique. In [Lad10] an example of its application is illustrated on a generic digital-communication bus, which is an example of programmable electronics that can be found in modern transportation vehicles. OHA uses a strict control of vocabulary, as well as a refinement in order to control the preliminary hazard analysis. OHA controls the vocabulary required to express system properties. It starts at a very high level with few object types, properties, and relations and extends the vocabulary only when necessary, i.e., when hazardous happenstances (HazHapps) are identified and need to obtain names in the vocabulary. This process of gradually extending the vocabulary is called formal refinement in the formal methods community.

OHA uses HAZOP as the method to identify the HazHapps. The formal refinement facilitates a level by level construction of cumulative tables of identified hazards. Mitigating methods can also be applied in the implementation to assure coverage of as many hazards as possible by using the controlled vocabulary. This brings the advantages of hierarchical development to hazard analysis.

In [Lad05], it is argued that the preliminary hazard analysis even at a high level of abstraction, as presented in there, can be sophisticated and can make use of complex control, such as the formal refinement. They claim that the major benefits of OHA come from the control that formal refinement brings, rather than from subtleties in application of HAZOP guide-words. The technique uses HAZOP as the basis for his ontological hazard analysis representation, the idea is represent hazard specific concept and relations and refining these representations until final representation is obtained and where it is aimed to get safety requirements. OHA is a hazard analysis technique that is using HAZOP concepts and represent them in an ontology, it is also combined the use of ontologies and formal methods.

The proposed approach, in this thesis, also combine ontologies and formal methods (model checking), however the focus is not on the refinement of the safety requirements.

3.3 Discussion

In this section discussion about limitations of the hazard analysis of previous section is presented.

FMEA, as the name suggests, aims to find failures, or to identify potential failure modes in a system and their causes and effects. This is also one of the main problems with FMEA because it was created to discover various failure modes in a system. This means that hazards which are not failure modes might not be discovered by performing FMEA. In addition, complex socio-technical systems include the interaction between people and technology and FMEA might have some difficulty in showing hazards that occur in such interactions since FMEA the FMEA does not consider the human factors element.

A problem with HAZOP is that it is not always straightforward to write a certain hazard using the provided *guide words* and it might be the case that some hazards might be better written without using *guide words*. Similar criticism could be applied to STPA and their *unsafe control actions* which play similar role in the analysis of hazards than HAZOP's *guide words*. However, STPA promotes a unique notation for the representation of systems, only for the purpose of the hazard analysis; this would not only facilitate the communication among stakeholders but would also focus on a representation of the system under study that aims to facilitate the discovery of hazards.

FTA main purpose is the analysis of faults in a determined system. Hazards that are not faults are not analysed with this method. It is exhaustive in analysing the *top event*. FTA makes an important distinction between events that all need to occur in order to the top or next level event to happen and events where one or more must occur in order to the top or next level event to happen. Bowtie also analyses exhaustively a *top event*. FTA and Bowtie analyses are good showing the scenario that leads to a *top event* and that is their main purpose. Bowtie being rather a qualitative approach and FTA being quantitative approach. They are usually performed after other initial analyses have been done. It could be argued that because of that, an improvement on the identification of hazards is less likely at this level. STPA proposes a tabular form for the analysis of each *unsafe control action* that has been found, where each unsafe control action would be like a *top event*. A tabular form is proposed because the STPA wants to avoid the direct cause chain model and rather a table is presented to analyse scenarios. The hazard analysis OHA focuses more on the generation of safety requirements and their refinement. Later some other formal methods techniques can be applied in order to demonstrate that those safety requirements are met.

3.4 Related research on Ontologies and Hazard Analysis

In the domain of hazard analysis and hazard representation there are a number of research and different representations developed, like ontologies for HAZOPs [SOR10, DSSO11], ontologies for hazard analysis techniques [LEE01, WB], and ontologies for hazard representation and risk assessment in a particular domain [MPB$^+$08]. Below, these examples are going to be explored and discussion will also be provided on what is proposed there and what is proposed in this thesis.

3.4.1 Representing hazards

In [LWW10], ontologies are used to represent geological hazards, specifically Isomerous geological hazard. They use ontologies in order to represent different concepts of these geological hazards. Relationships are represented in three levels: top-level ontology; domain ontology; and application ontology. The domain ontology plays the key role of link between top-level ontology and application ontology. Their use is to build semantic integration and share model. They differ from traditional data dictionary and meta-data method. The final result is a geo-hazard domain ontology. The knowledge from the ontology is taken from related standards such as industry professional standards, geological hazard information processing standards and expert knowledge. It is implemented in Protégé 3.4.4 and tested using geological hazard data of Three Gorges Area in China.

The aim is to integrate the information and resolve the problem of decision making when single hazard or group hazards occur. Ontology is used to carry out semantic sharing model to improve interoperability capacity of geological hazard information. Research in [LWW10] is very specialised in geological hazards and provide knowledge exclusively related to it. They do not use any hazard analysis technique and they are not interested in decomposing the hazard definition: initiating mechanisms and causes. Their work is related to the research proposed in the current thesis, in the sense that it uses ontologies as the means to reason about hazards but they focus on geological hazard solely. Nevertheless, this reinforces the idea of using ontologies to reason about hazards but in the research proposed in this thesis ontologies are also used to generate the inputs for the creation of state based models of hazard and hazard paths.

In another example about representing hazards, a risk ontology is described in [MPB+08], where risk and risk assessment are represented including concepts as hazards and threats. It shows how ontologies are used for risk assessment in a specific domain, the domain is prevention of floods. This is an example of the use of ontologies for safety analysis.

3.4.2 Representing Hazard Analysis Techniques

Research to combine ontologies and hazard analysis is presented in [WB, WB11] where it is introduced a framework for representing and updating Job Hazard Analysis (JHA) knowledge. JHA is a hazard analysis technique that focuses on job tasks as a way to identify hazards before they occur. It focuses on the relationship between the worker, the task, the tools, and the work environment. Usually, JHA is performed for construction activities to highlight and react to potential hazards. A company's personnel involved in JHA rely first on their experience and also on the company's internal knowledge represented in the form of safety rules. To perform JHA, as in any other hazard analysis technique, is not an easy task and adjusting JHA quickly, when changes in the construction methods and the schedule are made, is sometimes avoided.

The framework proposed in [WB, WB11] aims to improve access to a company's JHA knowledge. The framework uses ontologies for structuring knowledge about activities, job steps, and hazards. There is also a reasoning mechanisms to help evaluate the applicability of JHA concepts and identify applicable JHA safety rules. JHA safety rules can be tied to concepts of activities, job steps, and/or hazards in the framework. Hazards are represented as a concept exclusively related to Job Hazard Analysis technique and Job Hazard Analysis documents. For example, the heading of a JHA document should contain an activity. This activity is the primary activity class and it has the association relationship *hasStep*, which connects to the primary job step class. This is the representation of a the semantic connection of a Job Hazard Analysis final document where "an activity hasStep job step(s)". In the same way, another association relationship *hasHazard* is defined to enable the semantic connection of the primary job step and hazard classes.

On the contrary, the approach presented in this thesis represents the hazard concept as an independent concept, which has a global meaning that is not related to any particular Hazard

Analysis Technique but it could fit to any such technique. Still, the aim is to represent these hazards and related meaning and go deeper in order to help the safety engineer to identify possible overlooked hazards.

In [LG10], two ontologies are implemented: the Hazard Ontology that represents the possible hazards in specific domain (a shrimp supply chain scenario), and the Hazard Analysis Critical Control Point (HACCP) ontology representing a HACCP system. HACCP could be called a hazard analysis technique that systematically tries to improve safety in food products, and avoid or be aware of allergenic, chemical, and biological hazards in production processes that can cause the finished product to be unsafe, and designs measurements to reduce these risks to a safe level. These ontologies are tailored for companies in the food industry, such as supermarkets. The paper shows two ontologies for supporting hazard based reasoning in food supply chain. The aim is to provide the set of argumentation schemes based on the main requirement of the HACCP systems and in this way to record all the justifications on which a safety decision was taken. Another paper related to food hazards and ontologies is [YGHS12], where a knowledge model of domain ontology with the aim of hazard information extraction from Chinese food complaint documents was designed based on the ontology theory and some algorithms are implemented to extract hazard information by recognizing hazard words.

Research proposed in this thesis does not focus on an special hazard analysis technique like in [LG10], but it does want to combine different techniques to try to increase the knowledge about the hazards in a specific system similar to [YGHS12]. However, it is not focus just on extracting hazard by recognizing hazard words. Similarly to [LG10], two differentiated domains are represented.

Research in [SOR10, DSSO11] proposes a semi-automation of HAZOPs, since they want to assist but not replace the human expert in the conduct of HAZOP analysis. They use a kind of template called boilerplates. This template is used to organize requirements and a domain ontology to keep track of applicable deviation for the HAZOP analysis. They used three HAZOPs guide-words: omission; commission; and stuck. An ontology is the basis for a tool called KROSA. This tool prototype is used to facilitate the reuse of previous experience and in this way try to reduce the amount of human effort expended in HAZOP. It is also claimed that the tool can even attain higher significance in situations where highly skilled or experienced HAZOP experts are not available by enabling a platform whereby reliable previously documented cases can be reused in new scenarios by less experienced HAZOP team.

The work presented in [SOR10, DSSO11] has a similar purpose with regards to the proposed approach in this thesis, which is to help the safety analysis, experts and stakeholders with their tasks. However, the proposed approach does not represent a hazard analysis technique in particular.

3.4.3 Using ontologies for modeling

The authors of [EKM$^+$07] face the problem of failure management in the automotive domain, starting from domain models for logical and deployment models of automotive software. In [EKM$^+$07] a taxonomy is represented that later on contributes to develop their models. These models capture interaction patterns as a critical part of both logical and deployment architectures, introducing failure detection and mitigation as "wrapper" services to "unmanaged services", i.e. services without failure management. Finally, they use the failure management models to verify that a particular architecture meets its requirements under the stated failure hypothesis. The taxonomy is a source of information that can be consulted in order to develop their models. Research in [EKM$^+$07] combines domain representation that is used, as a source of information, to produce a state based model. This is something similar to what is proposed in this thesis. However, one of the aims of this thesis is to generate the state based models from the ontology representation.

Authors of [LEE01] describe the DAEDALUS framework. It is a methodology which, according to their claims, integrates design, diagnosis tasks, models, and modelling environments around a common Domain Ontology and Product Models Library. The Domain Ontology is used as a concept dictionary in support of multiple product-modelling environments (Task Applications). These tools are organized around graph-based modelling tasks in an enterprise task work-flow. The (Bayesian Network) BN-FMEA design FMEA task is configured as an instance of a Task Application with a Bayesian reasoner. The graphical editor is specialized for the construction of belief network graphs and allows the designer to generate a BN-FMEA Application Model. The BN-FMEA model is constructed using a set of functional, component, and physical quantity variables. The model is then mapped applying a series of different tools. It also uses, after some classification, a Task Ontology that contains the following sub-ontologies of direct relevance to this task: Functions; Components; Failure_States; Failure_Events; and Severities. The ontology is also used to facilitate semantic consistency of variables and relations in constructing Bayesian networks for design and diagnosis. Finally, the framework is presented as one possible approach for improved integration of generalised design and diagnostic modelling and knowledge exchange. The DAEDALUS framework combine modelling tools with ontologies, the models are saved in the ontologies so they can be used to combine diagnostic models with task models. The ontologies are used to help in this combinations, as well as to provide knowledge that improves these models. The improved models, in turn, can also be saved as new knowledge in the ontologies. In contrast, the approach proposed in this thesis aims to generate hazard state based models from hazard knowledge obtained from any hazard identification technique.

Table 3.5: Two descriptions of the same hazard

Poor example	Good example
`Round fired prematurely.`	`Artillery round fired from gun explodes or detonates prior to safe separation distance, resulting in death or injury to personnel within safe distance area.`

3.5 The Utility of Ontologies in Hazard Analysis

A good hazard description creates a picture that is invaluable in preventing a potential accident [Vin14]. Good practice is required but it is still common that hazards are expressed in less detail than it is required. The causes are many, sometimes due to the nature of the task, which is demanding, time consuming, perhaps even tedious. Because of many reason and human factors, sometimes hazard might be described in a poor way. Table 3.5 shows an example of a bad written hazard and the proper way to write it [Vin14]. This example shows how two hazard expressions refer to the same hazard yet the descriptions are quite different. It is possible to find well written hazards and poor written ones in the same hazard analysis. Moreover, it is possible that the experts performing the hazard analysis have a quite detailed understanding of what they mean but still write a poor description of the hazard. It could also be the case that the understanding is vague and because of that the hazard description is poor.

Systems can be complex, nested, have different stake holders and evolve over time. Hazards are described in text and they are part of an associated hazard analysis of a system model where hazard identification is key. Yet, Hazard analysis requires clarity and precision. Below, another example is shown where different problems with system boundaries and hazard description have been detected.

To address safety issues associated with infusion pumps, the FDA undertook various initiatives. For example, the FDA developed a number of hazard analyses for various infusion pump types. The Generic Patient Controlled Analgesia (GPCA) Pump Hazard Analysis [Arn], the FDA Guidance for Industry and FDA Staff [Cen10] and the Generic Insulin Infusion Pump (GIIP) Hazard Analysis [ZJJ10]. These analyses have some differences that would be useful to discuss. From reading the analysises, it is found that in the FDA guidance [Cen10] the hazards are hazards related to the device, in the GIIP analysis [ZJJ10] the hazards are related to the patient and in the GPCA analysis [Arn] the hazards are related to both patient and the device. As an example, in the FDA guidance [Cen10], the operational hazard for the infusion pump `Air in line`, is instead, in the analysis for the GPCA [Arn], the cause of the operational hazard `underinfusion`. It seems that the intended boundaries differ, but analysis [Arn], where the boundary seems to go beyond the infusion pump and focus on the hazards to the infused patient, has as operational hazard `failure to alarm` which is a hazard where the boundary is the infusion pump. In addition to the differences between the intended boundaries of their analysis, there are different ways to write similar, synonymous, or closely related hazards. For example, in [ZJJ10] the authors describe the patient hazard `incorrect treatment` and in the other two hazard analyses that were reviewed, there were hazards such as `incorrect therapy` or `incorrect dose or drug`. All these three differently described hazards might have been intended to mean the same and, as per the written text, it could just mean different ways of expressing the same patient hazard. Furthermore, it is possible to argue that `incorrect treatment` seems to be more general than `incorrect dose or drug`. Additionally, `incorrect treatment` might be a hazard directly related to the medical doctor prescribing the wrong treatment when `incorrect dose or drug` might be related to the infusion pump delivering the incorrect dose. There are various interpretations to be given. Yet, the experts that performed the analysis have a much better and precise understanding of what they meant with their textual descriptions. All these difficulties have arised from a relatively small example such as an infusion pump, when the systems become more complex so does the difficulties in the hazard analysis and the hazards descriptions. Moreover, hazard analyses used in wider systems need to be clear to a wide range of stakeholders, not only just experts or authors of the analysis; additionally, they are created to be reused in the future and long lived.

Well described and precise hazards are therefore important to improve the results of a hazard analysis techniques. However, well described, precise, yet contextualised hazards are of the great importance for the results of a hazard analysis, not least because hazard identification plays a role in the communication between different stakeholders and disciplines. For instance, the hazard `virus affecting the system`, when referring to an infusion pump medical device, is ambiguous between its Biology and Computer Science senses and therefore system boundary of the hazard, and can be disambiguated by assigning the correct domains to the contexts where it actually occurs with the help of an ontology.

3.5.1 Using ontologies to better describe hazards in a system

Ontologies, or explicit representations of domain concepts, provide the basic structure or framework around which knowledge bases can be built [Dev02, ST99]. Ontologies are specific, high-level models of knowledge underlying all things, concepts, and phenomena. As with other models, ontologies do not represent the entire world of interest. Rather, ontology designers select aspects of reality relevant to their task [Dev02, VRMS99].

Hazards can have slightly different definitions depending on the various standards, safety techniques, safety methods or approaches. Industries such as: oil and gas industry, chemical industry, construction industry, food industry among others can vary on the definition of hazards as well. The differences and commonalities motivate the development of an ontology based method for hazard identification that could be of use to the safety industry because it provides a systematic approach to review the hazard descriptions and the whole hazards analysis by means of the structure that the analyst need to follow in order to populate the ontology for further analysis. This way, it gives the opportunity to either remind the experts into giving more detail already known or push the expert to go deeper and try to find the details that he needs on the hazard description. In addition, it also provides a way to make the experts notice what seems to be missing in the description and add it.

Finally, an ontology has been devepoled to capture domain information related to hazards, systems, and hazards within a system. The ontology is designed to help the analysts in the task of hazard identification using the knowledge already gathered, and the relations that are already present in the documentation. An ontology provides the means to structure information by classes, property descriptions and relations between classes and individuals. Analysts can use this ontology for reasoning about missing relations between hazards, causes and consequences.

3.6 Uncertainty in Hazard Analysis

In [Lev11] Levenson argues that the current hazard analysis techniques, widely used in safety industry, encourage limited notions of linear and direct causality, where event A is linked directly to event B because it is seen as the direct cause. It is also claimed that nonlinear relationships, which are not of the kind "if A happens then B will happen", are difficult to incorporate in current popular approaches. An example is provided to develop the claim: Consider the statement "smoking cause lung cancer". It is argued that such a statement would not be allowed in the event-chain model of causality because there is no direct relationship between the two. This is because, many smokers do not get lung cancer, and some people who get lung cancer are not smokers. However, it is widely accepted that there is some relationship between the two, even though, this relationship might be complex and "nonlinear"(no direct). Hazard analysts often use probability approaches to capture this uncertainty.

Every model of a system is an abstraction, when modelling a system there might be uncertainties to deal with. There are two kinds of uncertainties: epistemic uncertainty, due to lack of complete knowledge of the underlying mechanism of a system under study, or aleatory uncertainty due to randomness, for example, certain conditions in the environment of a system under study. The analyst alleviate some of the uncertainty by performing the hazard analysis because it will increase the understanding of the system and its behaviour.

The analyst remove uncertainty by performing the hazard analysis. Therefore, to consider *smoking* a hazard might, considerably, depend on the model of the system and the way epsitemic and aleatory uncertainties have been modelled. This is because there is an underlying mechanism that causes that smoking influence the development of lung cancer.

Current research shows that the main way that smoking causes cancer is by damaging our DNA, including key genes that protect us against cancer. Over the years the built up damage might turn up normal cells into cancer. Current known facts are that smoking might cause certain damage to DNA and changes in DNA might cause cells to become cancer, but more facts are yet to known. Nevertheless, there could be a model and hazard analysis where `smoking` could directly cause `damage DNA in lung cells` and `damaged DNA in lung cells` could directly cause `lung cancer`. But, there might be other models where the relationship between `smoking` and `lung cancer` is not transitive nor direct, all depending on the level of knowledge about the system and the environment of the system. For instance, recent findings have increased the knowledge of the specific mechanisms underlying airflow obstruction, Chronic obstructive pulmonary disease (COPD), and tobacco addiction, these findings show substantial shared genetic architecture underlying airflow obstruction across individuals, irrespective of smoking behaviour and other airway disease [WSM+15]. This means, certain DNA profiles had lower risk of Chronic obstructive pulmonary disease (COPD) and certain DNA profiles had higher risk of COPD, explaining why some people develop COPD despite never having smoked in their lives. These genes appeared to affect the way lungs grow and respond to injury. Additionally, the research also found that 5 DNA sections are related to smoking addiction. This new knowledge will contribute to the study of smoking, lung cancer and their relationships and the model of the system will evolve accordingly. Meaning that this increased understanding will bring more detailed hazard analises for different people, and different controls and mitigations. Even so, uncertainty, both epistemic and aleatory, about hazards, causes and consequences will remain.

Traditional and broadly used hazard analysis techniques offer a systematic, formal and structured investigative way to study a system and potential hazards. In reality, traditional approaches like HAZOP or Bow-Tie not only continue to be applied to the industries where they originated but also to other various industries like software or cybersecurity. Nevertheless, some changes to current approaches are also proposed for these new industries.

3.7 Reasoning, hazard knowledge and ontologies

This section explores the challenges of the thesis with regards the automation of hazard analysis: representation of knowledege and reasoning about this knowledge. It discusses, in general, what is possible to reason about with ontologies applied to a hazard identification, and also hazard analysis, domain, and what it is aimed to reason about with our ontology representation. Finally, it also recognise the really important role of humans in hazard analysis but the need of support in this task.

3.7.1 Creativity

In the third edition of his book about HAZOP and HAZAN [Kle01] (1992), Trevor Kletz discusses the feasibility of the automation of HAZOPs by applying techniques from other fields, such as Artificial Intelligence (AI). The author explores if these techniques could replace the Safety Analysts. Kletz recognises the usefulness of case-based reasoning for recalling details of previous hazards or accidents in a HAZOP study. This is useful in helping the HAZOP team to have more detail on certain hazards and prevent their recurrence. Nevertheless, Kletz remains sceptic about the possibility that a computer or computer application might replace a HAZOP team to perform a HAZOP by itself. He gives two important objections to the complete (or almost complete) automation of HAZOPs

The first point is that "HAZOP is a creative exercise and those who are best at it are people who can let their minds go free and think of all the possible ways in which deviations might occur and possible methods of prevention and control". Artificial Intelligence (AI) techniques can manipulate logical rules but logic is just one aspect of human intelligence, and usually overrated. For example, most of the scientists who have recounted how they came to make an important discovery or to achieve a significant breakthrough have stressed that when they found the answer to the crucial problem they intuitively recognised it to be right and only subsequently went back and worked out why it was right [Kle01].

Artificial Intelligence is a broad research area that has different ways to model intelligence. There are the symbolic model, the connectionist model, the evolutionary model and the corporeal model. The dominant model in AI has been the symbolic, in fact, it remains very important and is currently considered the classic model in AI. It is a top-down model that is based on logical reasoning and heuristic search as pillars for problem solving, without the intelligent system needing to be part of a body or located in a real environment. That is, symbolic AI operates with abstract representations of the real world that are modeled by representation languages based mainly on mathematical logic and its extensions. The connectionist and the evolutionary model use biology as inspiration trying to kind of imitate neurons or evolution respectively. The corporeal model is becoming the current trend, where it is believed body is fundamental for knowledge, with Developmental Robotics as a field that is rapdily growing [LdM15]. It is right to say that current AI is not only a logic representation but wider than that.

As early as 1962, Douglas Engelbart [Eng62] wrote about a "writing machine that would permit you to use a new process of composing text... You can integrate your new ideas more easily, and thus harness your creativity more continuously" [DG17]. The idea behind is that creativity is a social process that can be augmented through technology. Currently, this field of study is called Computational Creativity and it builds software that exhibits behavior that would be deemed creative in humans. People who research Computational Creativity generally start from the definition that "a creative idea is a novel and valuable combination of known ideas". This means that physical laws, theorems, musical pieces can be generated from a finite quantity of knowledge (or data). The idea is that creativity could be model as an advanced form of problem solving that might involve memory, analogy, learning, and reasoning under constraints as well as other capabilities, and it is possible to replicate by means of computers. But there are degrees of creativity, and Computational Creativity refers to a degree that can not be called geniality nevertheless it can generate something that did not exist before, because it has combined existing things in a novel way. For instance, there are so many different combinations of notes that it is possible to find new combination of notes that no one had done before. This new found combination can be considered creative in the sense that it is original. Computational Creativity uses and combines different kind of techniques from case base reasoning to robotics [DG17].

Assuming that when Klets refers to “creativity to find deviations”, he refers to the different kind of combinations that deviations might take, computers might find as many or even more combinations than humans just because its processing capabilities may be able to find new combinations that might not have been thought before. However, with respect to the creativity needed for “possible methods of prevention and control”, it seems that Kletz refers to intuition and flair within an expertise. Within the field of knowledge management, there are two kinds of knowledge representation: tacit knowledge and explicit knowledge [AS10].

- Tacit Knowledge is knowledge that is hard to encode and communicate. It is personal, context-specific, and hard to formalize [AS10]. Polanyi [Pol62], who was the first to coin the name, explains that rules of art can be useful, but they do not determine the practice of an art; they are maxims which can serve as a guide to the art only if they can be integrated into the practical knowledge of the art, they cannot replace this knowledge. Tacit knowledge is a kind of knowledge that might need different kinds of representation. Yet, reasoning about this kind of knowledge falls close to the common sense intelligence which will be discussed later on (see Section 3.7.3).

- Explicit knowledge is the kind of knowledge that can be encoded and is transmittable in language. Explicit knowledge is possible to capture, acquire, create, leverage, retain, codify, store, transfer, and share [AS10].

3.7.2 A different kind of knowledge

The second point raised by Kletz is that the knowledge used in a Hazop is “broad and deep”, while expert systems are suitable only for “narrow and deep” knowledge. The knowledge used in HAZOP can be divided into four types: plant-specific (or system specific) knowledge, general process engineering knowledge, general scientific knowledge, and everyday or commonsense knowledge.

This affirmation can be further divided into three things. First, regarding what can be represented with respect to the (digital) physical space and processing time. Second, the kind of knowledge that currently AI is able to represent. Third, what kind of reasoning AI is able to make with this knowledge up to now.

Massive data-driven AI is currently under a lot of research and leading AI scientists, such as Ramon Lopez de Mantaras, think that lots of research effort will be focused on such research topics. Data-driven AI studies the possibility to access large quantity of data and being able to process it with ever-faster hardware with the aim to discover relationships among them, detect patterns, make inferences, and develop learning through probabilistic models, such as deep question-answering system such as watson or deep-learning systems among others. Broad and deep knowledge is a current research topic (with promising results) if it refers to the massive quantity of data that needs to be processed but it remains elusive if it refers to common sense knowledge or even intuitive knowledge [LdM15].

3.7.3 Common sense

A final statement by Klets was that "in HAZOP we are concerned with mundane matters as well as purely technical ones". This seems to be the fundamental objection to full automation of HAZOP or another current, broadly used, hazard analysis technique. Current AI has two different visions: Weak AI and Strong AI. Weak AI is the science and engineering that allow us to design and program computers in such a way that they can perform jobs/tasks that require intelligence. Strong AI is the science and engineering that searches to replicate human intelligence with machines [LdM15]. In 1992, Markin Minsky, Turing Award Winner, pointed out that scientists have not yet developed any AI software that uses "common sense". In order to use "common sense", first of all, there should be developed a common sense knowledge base [Min92]. This knowledge base will have to contain information about "obvious" knowledge, the kind of knowledge that even a child has. Also, these common sense software will need to understand the "functions" and "uses" of this obvious knowledge. For example, consider a string – a child could tell dozens of ways of how to use or not to use a string but computers do not have this knowlegde. The same applies for many other words [Min92].

Additionally, this common sense knowledge needs to have a representation, this is, a data structure or any other way to build that knowledge into the memory. Up to 1992, there were different kind of visions and there was not common agreement on this. Minsky concludes that the best way is to find out how to make"unified theories" about how to combine non-unified theories [Min92]. Years later, there were high hopes that common sense representation would be soon discovered. In 1999, an article [Win99] stated that we can discover the computational basis of natural intelligence (common sense) during the next ten years or so. This did not happen. However, the author pointed out that more research should be put on I/O channels because it must be that in order to understand intelligence, we must understand the contributions of vision, language, and motor faculties. Recent research is growing quickly in this area.

Nevertheless, up to now, common sense intelligence is still elusive to machines, as Marvin Minsky pointed out in a recent interview [dM] (2016). Even more, he also stated that the most interesting progress in AI was made between the early 60s and the late 70s. There are new currents of research in AI such as Developmental Robotics that is inspired in children's cognitive development that are promising and they consider I/O channels as part of their intelligence representation [LdM15]. Currently, a very important problem to solve in AI is how to integrate the different elements of intelligence: perception, representation, reasoning, action, and learning. Those integrated systems are a fundamental prerequisite for achieving artificial intelligence of a general nature (common sense) [LdM15].

3.7.4 Human-machine synergy

Industry is broadly giving much trust into AI. There are programs that buy and sell stocks at a high speed, there are judges who use AI to dictate judgments, there are banks and insurance companies that make decisions based on AI, there are police services that use AI to decide where to put more forces to law enforcement. Decision making is passing from humans to computers. It is not necessary the right thing to do because algorithms can also be biased because they work according to the way they are coded and are data given, which, at the same time, can be biased [Ter17, Cor]. For example, New York police department discovered that their application for guiding their patrols was making them more prone to arrest black people, just because the parameters and data were biased, which was later on corrected [Cor]. Ramon Lopez de Mantaras defends that AI should always be a complement of the human being. Humans and computer working together are better than either of them working separately. Nevertheless, the last decision should be always be made by a person, and this person should be able to analyse if it is coherent what the machine says [Ter17].

3.7.5 Thesis author's view

Hazard analysis requires different kind of knowledge, expertise and creativity. Therefore, the knowledge, expertise, creativity and common sense of the people involved is key. Nevertheless, hazard analysts can benefit greatly from applications that can assist them in particular tasks. Logic is not able to represent every different kind of knowledge and expertise of the safety analyst. No single AI will be able to represent the wide hazard analysis knowledge, a combination of different AI techniques might help with this representation but a way to integrate those techniques and made them interact among themselves will be needed. One of the aims of this thesis is to represent the knowledge that has been previously captured by a hazard identification technique and system description. What is wanted is to be able to find relationships, which could have not been seen, and point out the relevant information that might have been hidden because of the quantity of information taken. In addition, another aim is to understand these relationships and transform them into hazard paths which could warn the analyst about the development of certain hazards. Finally, this proposal aims to help the safety analyst in the decision making process.

4 A New Ontology-Based Analysis Method

This thesis introduces a structured analysis method that enables computer-aided assessment and improvement of hazard analysis. The ultimate goal is to help safety analysts and other stakeholders revise in a systematic way hazard analysis results, and check that hazards or relationships between hazards have not been accidentally overlooked this way improving the results. This systematic analysis could lead to the better understanding of hazards and causes, and contribute to creation of scenarios that explore corner cases that may otherwise be not considered in the analysis.

The presented method provides:

- An ontology suitable to describe the structure and behaviour of the system.
- A process to link hazards to structural and behavioural aspects of the system.
- Rules that can be used by the analysts to check consistency and dependency relations between hazards, causes, and consequences.

This chapter is organised as follows, Section 4.1 provides an overview of the method, Section 4.2 explains the ontology that has been built and on which the method is based. Section 4.3 explains the first part of the method called Systematic exploration of dependencies and Sections 4.3.1, 4.3.2, 4.3.3, 4.3.4 explain the various stages of this first step. Section 4.4 describes the second part of method called state based hazard model analysis and sections 4.4.1, 4.4.2, 4.4.3 explain the stages of this part. Section 4.5 describes the technical transformations done in order to get a state based hazard model and analyse using the model checker MCMAS [LQR09].

4.1 Overview of the analysis method

The ontology-based analysis method consists of two steps and seven stages (see Figure 4.1):

- **Step 1: Carry out a systematic exploration of dependencies**. In this step, information provided in the hazard analysis and in the hazards worksheets (description in natural of the system under analysis, identified hazards, their causes in system design, and potential consequences) are collected and systematically analysed. The step is decomposed in four sequential stages:

  - **Stage 1**: Collect information about the system under analysis.
  - **Stage 2**: Revise the hazard analysis with regards to the collected information.
  - **Stage 3**: Develop structural and behavioural representations of the system under analysis.
  - **Stage 4**: Search implicit and overlooked relationships.

- **Step 2: Perform an automated search of hazard paths**. This step disambiguates information in hazard analysis and worksheets by constructing and analysing a state based model of hazard paths. The exploration of different relationships (paths) between hazards is carried out with the help of an automatic search engine. Overlooked hazard paths can be captured in the form of hazard scenarios highlighting common causes for different hazards, or hazards whose severity or frequency needs to be reclassified. The step can be decomposed in three sequential stages:

  - **Stage 5**: Generate a state-based hazard models from the ontology rules
  - **Stage 6**: Explore the state-based hazard model.
  - **Stage 7**: Visualize hazard paths.

In order to fully describe the proposed ontology based method, the foundational ontology on which the method is built on needs to be explained. The following section would tackle this.

4.2 Foundational Ontology

Ontologies are the foundation of our proposed method. This section gives an brief explanation to ontologies and explains the tools used to represent the ontology, as well as, describes the concepts and approach used later in the proposed method. Finally, it is going to be shown the structure of the proposed ontology and the reasoning capabilities that can be exploited from an ontological representation of a hazard analysis.

4.2.1 Developing the Ontological Model

This Section defines the overall structure of the ontology developed in this thesis and how it will be instantiate during a hazard analysis. In Artificial Intelligence, the term ontology has got two slightly different meanings. First of all, ontology is a representation of a hierarchy of concepts and their relations in a certain domain that it is intended to be captured. In its second sense, the term ontology is also referred to a body of knowledge or a collection of facts describing some domain using a hierarchy of concepts and relations. The distinction is that the former

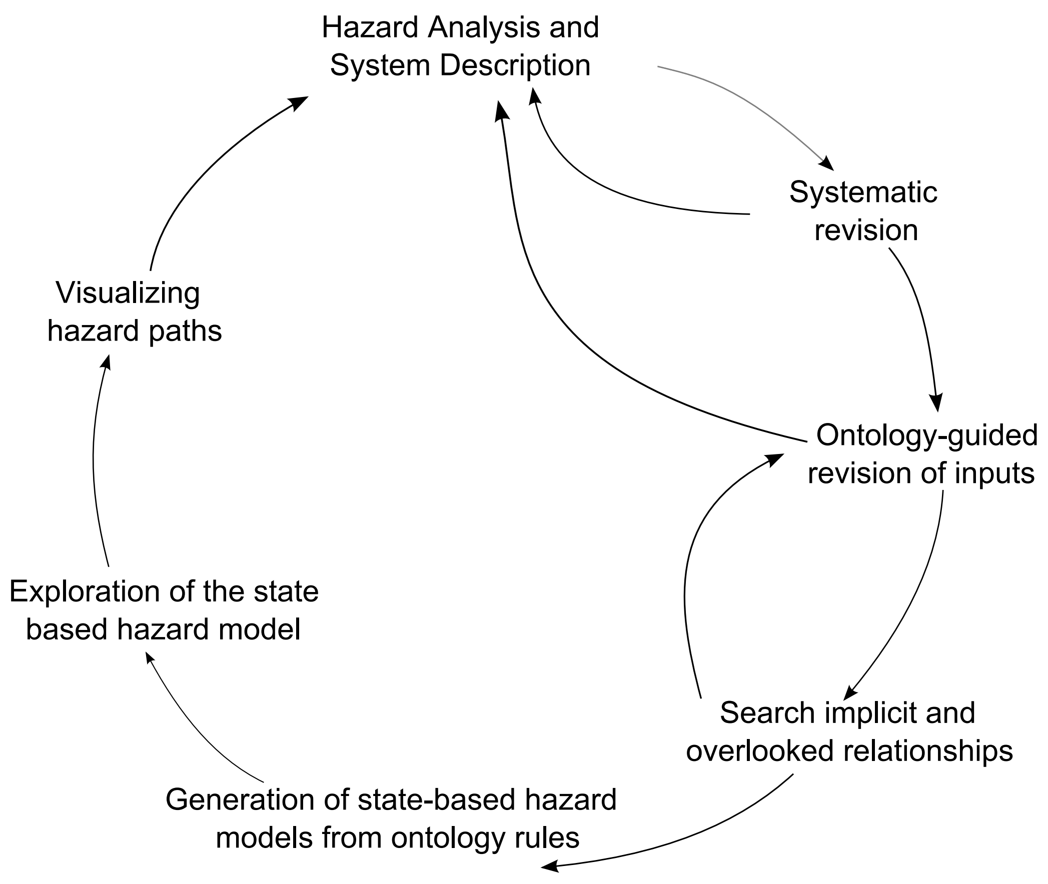

Hazard Analysis and System Description
Systematic revision
Ontology-guided revision of inputs
Search implicit and overlooked relationships
Generation of state-based hazard models from ontology rules
Exploration of the state based hazard model
Visualizing hazard paths

Figure 4.1: Diagram presenting an overview of the stages of the analysis method presented in this thesis.

emphasizes the use of ontology as a set of terms for representing specific facts in an instance of the domain, while the latter emphasizes the view of ontology as a general set of facts to be shared [CJB99]. At this moment, we want to put emphasis in the concepts that help us to represent our domain. Therefore, when we use the term ontology we refer to its first sense and when we use the term instance of the ontology we will refer to its second sense. When developing an ontology, we aim to formally and explicitly describe concepts in a domain, as well as their properties. Describing the concepts involves defining classes for these concepts and arranging them in a hierarchy. The first domain we work on is the hazard analysis and Hazard identification(HazID) worksheets, which consists of various hazards where each hazard might have multiple causes and each cause have one or more possible consequences.

The Protégé environment is used in this thesis for ontology development. The tool has been chosen not only because it is the leading ontological engineering tool but also it provides an integrated environment for the ontology development and it has different features supporting different tasks during the ontology life cycle. The knowledge representation language of choice was OWL-DL because it is the most used by the ontology development community. We have chosen OWL-DL and SWRL also because OWL-DL and SWRL offer a number of sophisticated reasoning capabilities. SWRL is the language for reasoning within the ontology. SWRL is a reasoning language for OWL-DL that uses rules of the form of an implication between an

Table 4.1: Basic example of HazID worksheet

Hazard	*Cause*	*Consequence*
$A - 5$ `Toxic gases in tunnel`	`Toxic gases enter in tunnel from alignment or station` `Maintenance personnel release toxic gas while performing work`	`Injury, death, service disruption`

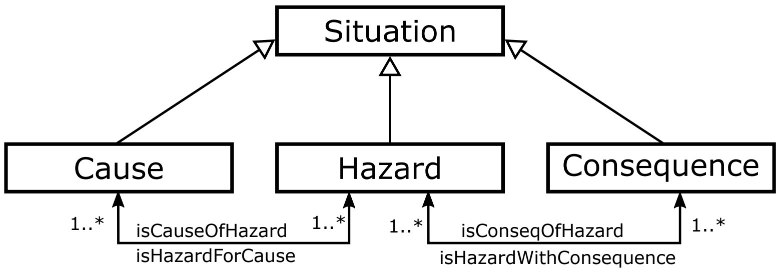


Figure 4.2: Concepts that are part of our ontology

antecedent and consequent. More information on this has been given in sections 2.4 and 2.4.2 of Chapter 2.

4.2.2 Core Ontology

Table 4.1 shows a basic example of the main part of a HazID worksheet. The example is taken from [EA06]. The hazards description for hazards $A - 5$ is `Toxic gases in tunnel`. The cause of this hazards is `Toxic gases enter in tunnel from alignment or station`. The alignment in the context of a railway is the ground plan of the railway, as well as the path that the train follows. The consequences for hazard $A - 5$ are `injury`, `death` or `service disruption`. Another cause of hazard `Toxic gases in tunnel` is `Maintenance personnel release toxic gas while performing work`.

Figure 4.2 shows the concepts of Hazard, Cause, and Consequence represented in our ontology. These concepts can be used to define: a set of hazards, H, consisting of $h_1 \dots h_n$ individual hazards; a set of causes, consisting of $c_1 \dots c_n$ individual causes; a set of consequences, consisting

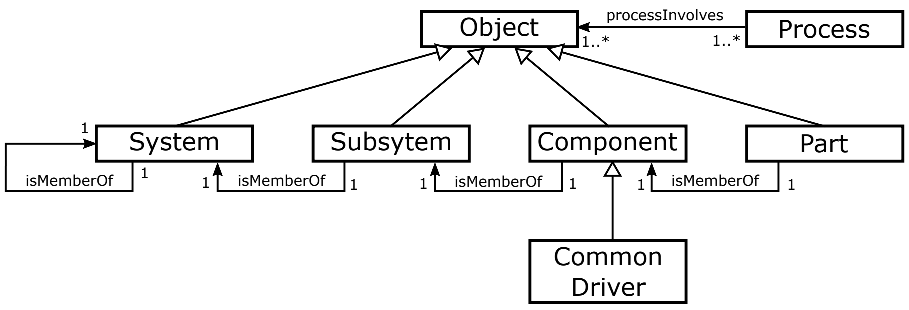


Figure 4.3: Part of Ontology

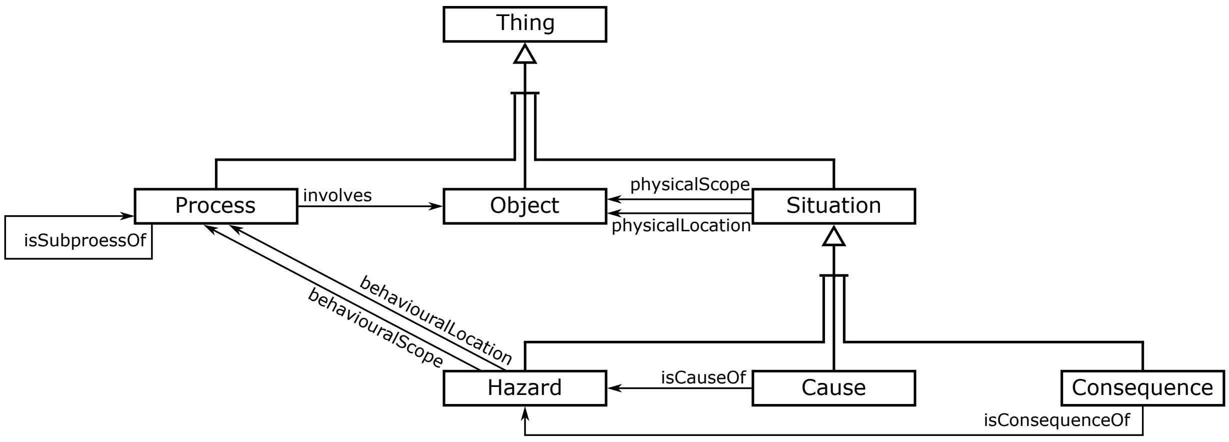


Figure 4.4: Core classes of the ontology

of $q_1 \dots q_n$ individual consequences. Figure 4.2 shows the relationships among these concepts. The relationships cause-hazard and hazard-consequence are many to many. That is, a cause might occur on different hazards, and a consequence might occur on several hazards. In the hierarchy of types, **Situation** is a higher node and the super type for **Hazard**, **Cause** and **Consequence**. Hazard Identification is performed within a system and its boundaries. In order to represent the basic ontology it is important to define two core concepts to represent the characteristics of a system and its behaviour. The basic concepts would be **objects** and **processes**. These concepts are used in order to represent the initial design of the system and its behaviour. The system structure can be represented as follows. First of all the concept *object* symbolises any part of the system structure including the *system.* An *object* can be a *system*, *subsystem*, *component* or a *part* of the *system.* This is an arbitrary cutting short of a possibly infinite hierarchy. It has been done for convenience, there are 4 levels of granularity and each level will be explained in the following paragraphs.

Generally, any system A is part of a larger system B. In addition, system A would have parallel systems working alongside. The parallel systems would be, if relevant, the environment of system A. The parallel systems of system A are subsystems of system B. System A is composed of subsystems that are interlinked and work together. The boundaries of system A will be the common frontier between the system and its environment. Figure 4.5 shows that subsystem A is a member of system B. They have different system boundaries and, as a result, concentrate on different hazards. System B provides the environment for system A. A hazard, normally, is described with respect to its system boundary so it is a relative term. The scope of system A will be localised out of the boundaries of the system. Depending on what characteristics of the system we are focusing on, we differentiate between behavioural or structural (physical) scope because it helps us to have more detailed information about the hazard we are describing.

System is an important concept needed to be represent in the ontology. The physical domain of the *system* is composed of *subsystems. Subsystems*, in turn, are composed of *components*, and

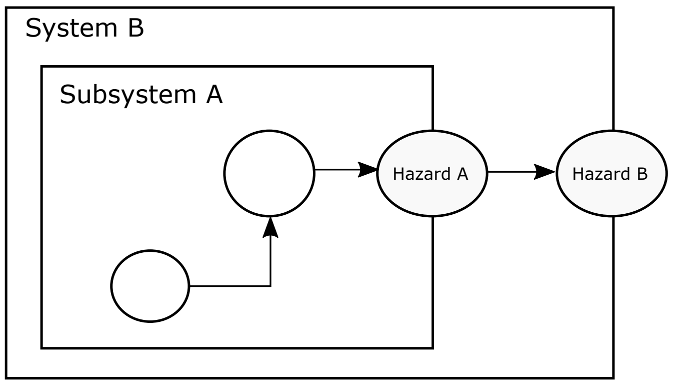


Figure 4.5: Hazard in Systems. Based on [Rai01b].

components are composed of *parts*. From the CLIOS formalism [SDM+09], we take the notion of *Common drivers*, which are shared *components* among *subsystems* of a *system*. Figure 4.3 shows the concepts of *Object*, *Process*, *System*, *Subsystem*, *Component*, *Common Driver* and *Part* modelled. The initial design of *system* set S, consists of different natural divisions such as, a *subsystem* set B, a *component* set M, a *part* set P. Where it is identified a set of *subsystems*, B, consisting of $b_1 \dots b_n$ individual *subsystems*; a set of *components* M, consisting of $m_1 \dots m_n$ individual *components*; and finally, a set of *parts* P consisting of individual *parts* $p_1 \dots p_n$. A *part* might occur within a *component*, so then the relationship is many to one. Similarly, the relationship between *component* and *subsystem* and between *subsystem* and *system* is many to one. In addition, a *system* has a recursive relationship, where a *system* can be part of another *system*, and the relationship will be again many to one.

The representation of the behaviour of a system is through the concept *process*. The behavioural description of a *system* can be described as a set of *objects* O consisting of $o_1 \dots o_n$ individual *objects*; a set of *processes* C consisting of $c_1 \dots c_n$ individual *processes* where an *object* may participate in different *processes* and a *process* may occur in different *objects*, so the relationship is many to many. A *process* can be decomposed into sub-processes.

Figure 4.4 shows fundamental concepts of the basic ontology. *Thing* is a generalisation of *objects*, *processes* and *situations*, it represents the class of all things, or the abstract objects that can be described by the criteria for being something. This concept is taken from class defined ontologies. In addition, Figure 4.4 shows four new relationships that we need to discuss. The hazard information within the ontology needs to include information such as the scope and location of the hazards. When we talk about the scope of the hazard, we refer to what affects the hazard directly, we therefore refer to two kind of scopes: *behavioralScope* and *physicalScope*. If a hazard affects a specific system, that system will be the scope of the hazards. When we refer to the *physicalScope* we refer to the boundary of the hazard. For example if *subsystem A* is a member of *system B*, then *system B* is the scope of *subsystem A*. When we talk about *location*, we refer to where the *hazard* is located in terms of *objects* and also in terms of *processes*, the

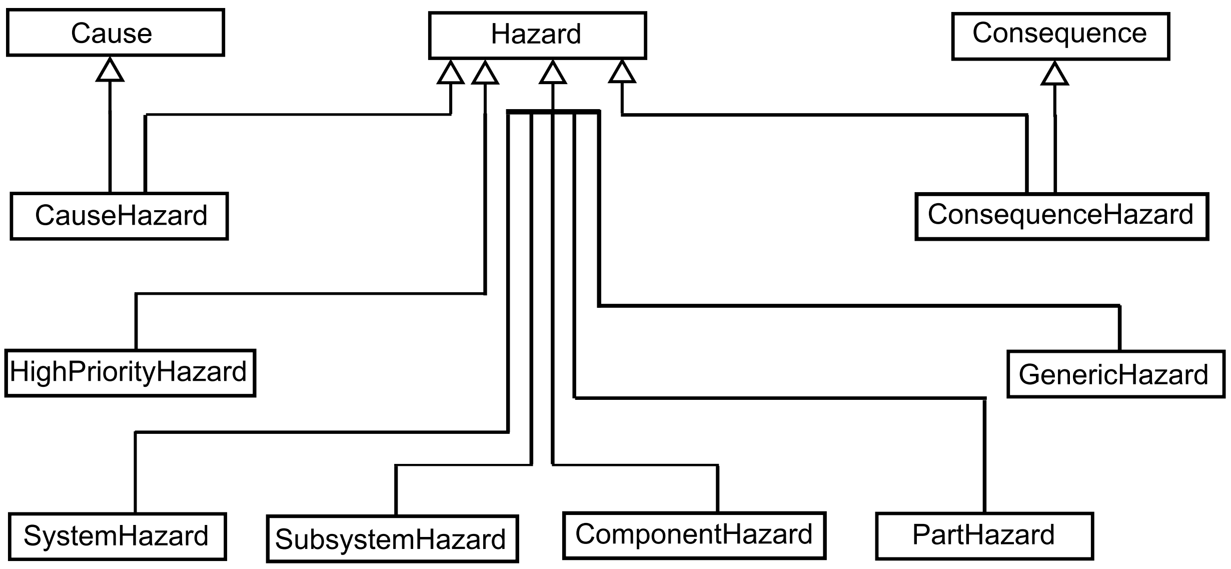


Figure 4.6: Class hierarchy of Hazards in the ontology

relationships that we will use will be called *physicalLocation* and *behaviouralLocation*.

Hazards, *causes* and *consequences* could be interlinked by any of these characteristics. This kind of knowledge is already known by the safety analyst or it can be deduced by reading the HazID description. The idea is that information provided by these relationships (object properties in Protégé) offer a natural way to establish associations among hazard meaning, or senses, in a certain hazard analysis process. This can be profitably used during, for example, the processing of the hazID worksheets to disambiguate the process. For instance, the hazard `virus affecting the system`, when referring to an infusion pump medical device, is ambiguous between its Biology and Computer Science senses and therefore system boundary of the hazard, and can be disambiguated by assigning the correct domains to the contexts where it actually occurs.

4.2.3 Extended Ontology

In the previous section, we have explained the core concepts of our ontological proposal. In this Subsection we explain the extended ontology. The extended ontology is formed using more specific types, i.e., lower nodes in the hierarchy. This provides a more granular organisation of the hazards, causes and consequences, and relations between them. Figure 4.6 shows a more detailed hierarchy of the classes in our ontology.

Hazards have a scope. In order to generate a hierarchy to hazards according to their scope, we create equivalent classes called *PartHazard*, *ComponentHazard*, *SubsystemHazard* and *SystemHazard*. When editing the ontology, the analyst can decide which of the *SystemHazards* belong to the *HighPriorityHazard*. These are the *hazards* that are deemed the most important for the analysis by the safety analyst. The class *GenericHazards* is useful to handle worksheets with generic hazards identified using, e.g., a Preliminary Hazard Analysis.

The class *ConsequenceHazard* is subclass of class *Hazard.* The classes *Consequence* and *CauseHazard* are subclasses of *Hazards* and *Cause.*

4.2.4 Using the Ontology to Reason about Hazards

Having established the conceptual structure and instantiation of the ontology, this can now be exploited to reason about the system and associated hazards. In this subsection, some concepts will be explained in order to understand the reasoning approach implemented within the ontology.

Synonymy, Entailment and Non-Direct Causality

As it has been discussed in Section 3.6, systems can be non deterministic due to different circumstances. Nevertheles, we can model structure of hazards and interaction using a deterministic logic because the properties we are interested in are captured by this abstraction. These are: definition of terms, equivalences and connectivity.

Hazard identification is a demanding task and relies on detailed descriptions. It is difficult to remember each and every hazard and cause in a hazard analysis and, as a consequence, the worksheets may refer to hazards, causes and consequences in different written forms. Sometimes this reflects real differences that should be captured, while in other situations the differences are only artificial, and should therefore be amended. For example, when analysing a road system, we could have the following hazards: $H-1$ `Vehicle drives on the wrong lane`; $H-2$ `Vehicle passes through the wrong lane`; $H-3$ `Vehicle on the wrong lane`. $H-1$ and $H-3$ could have the same meaning, or $H-3$ could mean that the vehicle actually stopped on the wrong lane. $H-1$ and $H-2$ could also have the same meaning if each just means that a vehicle temporarily drives on the wrong lane. An ontology could help the analyst to find and identify these hazards, as well as to disambiguate them, when needed, by means of logical rules applied to the structured knowledge or knowledge base.

When two situations (hazards, causes, consequences) have the same meaning, we will call them synonyms. Using only inference rules, we cannot determine whether a hazard is a synonym of another hazard, cause or consequence. To resolve the problem, we rely on different relationships established between situations that are part of the HazID entry. For example, two synonym hazards might have the same physical location, physical scope, behavioural location, or behavioural scope. Therefore, using rules to find hazards where multiple similar relationships occur can help to spot synonyms. In addition, a synonym is not the only kind of relationship that can be spotted.

In logic, entailment [Fel90], or strict implication, is properly defined for propositions; a proposition P entails a proposition Q if and only if there is no conceivable state of affairs that could make P true and Q false. Entailment is a semantic relationship because it involves reference to the states of affairs that P and Q represent. We generalise the term in order to talk about hazards. For us, in the context of the hazard descriptions for a certain system, entailment refers to the special relationship between two hazards, where a hazard, $H1$, of a more general hazard, $H2$, also entails $H2$. In addition, if $H1$ entails $H2$, then it can not be the case that $H2$ entails $H1$. In order to explain this, we introduce a brief example. We mentioned before that the alignment in the context of a railway is the ground plan of the railway, as well as the path that the train follows. The crossing is the crossroad between the train path and the motor road. Hazards $H-6$ `Motor vehicle on alignment` and $H-7$ `Road vehicle drives around crossing gate` refer to a motor vehicle inside the train path. Because the crossing is part of the path of the train, $H-7$ entails $H-6$, because while a motor vehicle drives around the crossing gate, the motor vehicle is on the alignment. In addition, $H-6$ does not entail $H-7$ because `Motor vehicle on alignment` does not necessarily mean that the vehicle is driving around the alignment. Using inference rules and a reasoner could therefore help the safety analyst to disambiguate the meaning of those hazards and also to find hazards interlinked by the causation relationships.

Rules

In this section, the rules that help the analysis to find non-direct or indirect causality between hazards and causes are shown. The property representing this relationship is *isIndirectCauseOf*. The idea is that when analising the current instance of the ontology, when a potential synonym, entailment, similarity (*mightRelateTo*) relationship has been found then this could also mean that there are some causal relationships to add to the results of this analysis.

$$\begin{aligned}
&\mathit{Object}(?l1),\\
&\mathit{Situation}(?s),\\
&\mathit{physicalLocationOf}(?s, ?l1),\\
&\mathit{scopeOf}(?s, ?c),\\
&\mathit{behaviouralLocationOf}(?s, ?p1),\\
&\mathit{Object}(?c),\\
&\mathit{Object}(?l2),\\
&\mathit{Situation}(?h),\\
&\mathit{physicalLocationOf}(?h, ?l2),\\
&\mathit{scopeOf}(?h, ?c),\\
&\mathit{behaviouralLocationOf}(?h, ?p2),\\
&\mathit{isSubProcessOfProcess}(?p1, ?p3),\\
&\mathit{behaviouralScopeOf}(?h, ?bs),\\
&\mathit{behaviouralScopeOf}(?s, ?bs),\\
&\mathit{isSubProcessOfProcess}(?p2, ?p3)\\
&\rightarrow \mathit{possiblyEntails}(?s, ?h)
\end{aligned} \tag{4.1}$$

Rule 4.1 is explained as follows. Every **situation** (this is a hazard or cause or consequence) happens within some object (based on the boundary of hazards), the property representing this is *physicalLocationOf* $(?s, ?l1)$. The behaviour of the system is described via **processes**, the property *behaviouralLocationOf* $(?s, ?p1)$ represents that process $p1$ is ocurring while the situation s is being active. Property *behaviouralScopeOf* (s, bs) represents that process bs would be directly affected by situation s and property *scopeOf* $(?s, ?c)$ represents that the structure of the system c would be directly affected by situation s. When two situations s and h happen to share the same behavioral and structural scope, it could be understood that these two situations could be affecting the system in some similar ways, but that is just an initial intuition. In addition, if these two situations s and h happen to have behavioural locations $p1$ and $p2$ that share same super process $p3$ then there might be an entailment relationship.

The rule 4.2 refers to two situations s and h that share the same structural and behavioural location as well as sharing the same structural and behavioural scope. Therefore, there is a real potential of s and h actually being the same hazard.

$$\begin{aligned}
&\mathit{Situation}(?s),\\
&\mathit{physicalLocationOf}(?s, ?l),\\
&\mathit{behaviouralLocationOf}(?s, ?p),\\
&\mathit{behaviouralScopeOf}(?s, ?q),\\
&\mathit{scopeOf}(?s, ?o),\\
&\mathit{Situation}(?h),\\
&\mathit{physicalLocationOf}(?h, ?l),\\
&\mathit{behaviouralLocationOf}(?h, ?p),\\
&\mathit{behaviouralScopeOf}(?h, ?q),\\
&\mathit{scopeOf}(?h, ?o),\\
&\rightarrow \mathit{possibleSynonyms}(?s, ?h)
\end{aligned} \tag{4.2}$$

The rule 4.3 tries to take advantage of generic hazards. A generic hazard would be a known hazard, or a hazard that is common to certain industry where the system under study operates. For example it could be argued that a generic hazard for a railway system would be "Fire". A more detailed hazard in the ontology can be related to a generic hazards through the property *hasGenericHazard* $(?p, ?g)$. For example a hazard such as "Fire on station platform" or "Fire on alignment" would have as generic hazard (*hasGenericHazard*) "Fire". Two situations p and q with shared generic hazard g, could be related if the process where hazard p occurs involves the structural location of hazard q. The behaviour of the system is described via processes; a process may involve various objects (*processInvolves*$(?a, ?o)$ property).

$$\begin{aligned}
&\mathit{Object(?o)},\\
&\mathit{behaviouralLocationOf(?p,\ ?a)},\\
&\mathit{Situation(?p)},\\
&\mathit{hasGenericHazard(?p,\ ?g)},\\
&\mathit{processInvolves(?a,\ ?o)},\\
&\mathit{Situation(?q)},\\
&\mathit{hasGenericHazard(?q,\ ?g)},\\
&\mathit{physicalLocationOf(?q,\ ?o)},\\
&\mathit{Process(?a)}\\
&\rightarrow \mathit{migthRelateTo}(?p, ?q)
\end{aligned} \tag{4.3}$$

These rules would be applied on Stage 4.3.4 of the proposed method. The following section will explain the ontology based method.

4.3 Step 1: Systematic Exploration of Dependencies

The first step of the proposed analysis method creates the ontological representation that will be used for the automatic analysis of dependencies in Step 2. The ontology produced in this step can be used as the source of new information that will help us in further analysis, like the hazard paths.

4.3.1 Stage 1: Hazard Analysis and System Description

In this initial stage, system description and hazards list are collected from different sources, including: current studies, hazard data collected in the past, organizational guidelines.

4.3.2 Stage 2: Systematic Revision

A systematic comparison is performed to find possible incoherences. The work is structured in two sequential tasks:

1. Task 1: Compare the hazard analysis against the description of the structure and behaviour of the system in order to find if it is coherent.

2. Task 2: Investigate the different diagrams already given in order to start the study. Investigate the current documents where specifications, descriptions, explanations of the system are given. The aim is to find possible incoherences. For example, hazard descriptions that might not reflect the system's structure.

4.3.3 Stage 3: Ontology-guided revision of the inputs

This stage makes use of the foundational ontology that has been created as part of the current thesis proposal. This ontology is described in Section 4.2. In this stage an instance of the foundational ontology is created. It includes two main steps:

1. Every hazard is associated to the structure and the behaviours of the system. This task is called *hazards localization*, and is useful to understand the scope of the hazards (e.g., system-level, or specific to a subsystem).

2. An ontological representation is created that gives a systematic way to a revision of the hazard analysis and the system description. The explicit relationships among all the different "sets" can tell more about the information we have in the analysis and the system.

Table 4.2: Hazard and Operability Studies Table

#	Guide Word	Deviation (Potential Hazard)	Causes (Potential Hazard)	Consequences (Potential hazard)

Redundant information can be revisited and polished and new relationships can also be created. All these new information is there but, often, difficult to grasp.

The produced ontology will explicitly show and record the following information, which enhances the organisation of the different information provided for the hazard analysis:

- System description: System, subsystem, component, part. Common driver.
- System behaviour: Processes and subprocesses.
- Relationships among the system description and the system behaviour.
- Hazards (Hazardous situations), Causes and Consequences.
- Physical location of Hazards, Causes and Consequences.
- Behavioural location of Hazards, Causes and Consequences.

In the ontology, the system is recorded according to structure, hierarchy and behaviour. Figure 4.3 in Section 4.2 remind us of the structure in this case. We considered that the fundamental entities to represent a system A were:

- System A
- Subsystems of system A
- Components of the subsystems.
- Parts of all the components.

There are 4 levels in the ontological hierarchy showed above. The class Part, as the final one in the hierarchy, can also be recursive, this means, a part P_1 can be part of part P. The generic class (or set) to which all of the mentioned set belong is *Object*, all of them are, ultimately, *objects*. We will need to enter the information in this way.

In addition to the structural representation of a system in the ontology we also needed to represent the behaviour of a system. The way we represented the behaviour of a system with respect to the structure of a system is as follows:

Table 4.3: Preliminary Hazard Analysis Table

#	Hazard	Causes (Potential Hazard)	Effects/Consequences (Potential Hazards)

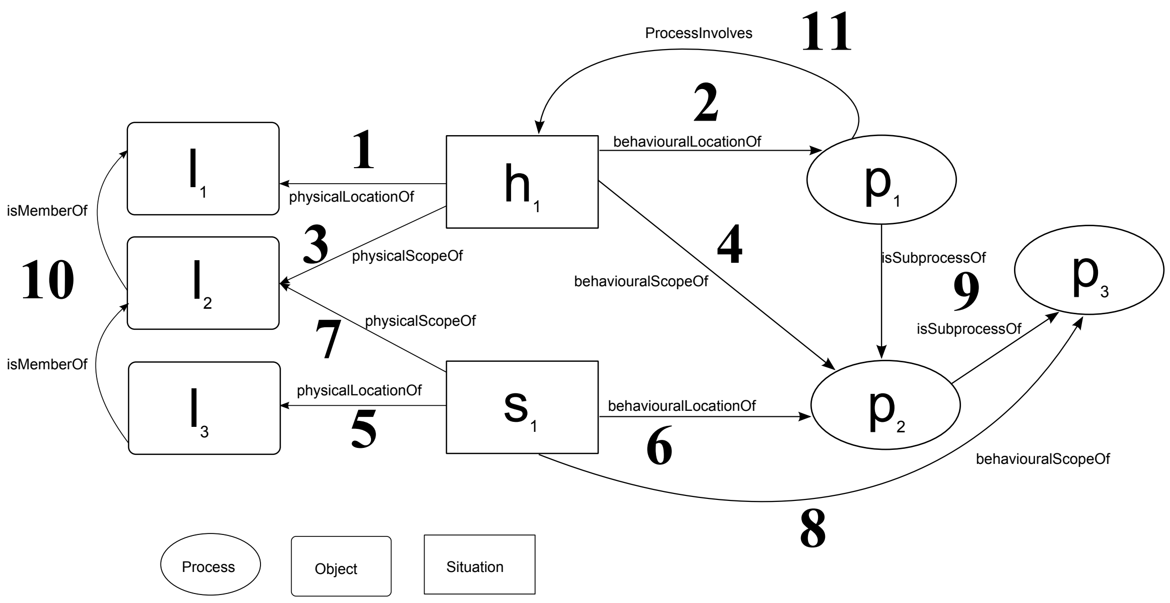


Figure 4.7: Different relationships represented in our Ontology

- A process Pr could be a subprocess of Process Pr_1 or superprocess of Process Pr_2.
- A subsystem's behaviour is its process which in turn is guided by its components behaviour through other processes.
- Similarly for the system's behaviour.
- A component's and part's behaviour is its process(es).

Table 4.2 is an example of a HAZOP worksheets taken from [Rai01a, Eri05]. The information that can be find: deviation; potential effects of the deviation; and potential causes of the deviation. To translate into the required format, the analyst can use the following mapping: the deviation is the potential hazard; the causes of this deviation could be at the same time deviations in another node of the HAZOP worksheet, some of the consequences could also be situations that at the same time are deviations (hazards) into another HAZOP node or deviation (hazard).

Table 4.3 is an example of a Preliminary Hazard Analysis worksheets. The worksheet contain the following information: hazards, causes, effects/consequences [Eri05, Rai01a]. Causes could be also a hazard, the consequences are situations that could be also hazards. Figure 4.7 shows the relationships which are implicit or explicit knowledge taken from the hazard analysis and system description. These relationships will be made explicit and will be named. Additionally, the following relationships can be seen numbered in Figure 4.7:

1. A relationship from a hazard to the System structure where the hazard is located.
2. A relationship from a hazard to the process that represents a certain behaviour of the system and where the hazard occurs.
3. A relationship from a hazard to the correspondent structural scope: System structure which will be the structural Scope of hazard.
4. A relationship from a hazard to the process that is going to be affected because of the occurrence of this hazard, it is called behavioural scope.
5. A relationship from a cause to the System structure where the cause is located.
6. A relationship from a cause to the corresponding process. A process that represents a certain behaviour of the system and where the hazard occurs.
7. A relationship from a cause to the System structure which is the structural Scope of it.
8. A relationship from a cause to the corresponding process. A process that represents the behavioural scope of the cause.
9. A relationship from a consequence to the System structure where it is located. Number 5 in Figure.
10. A relationship from a consequence to the process that represents a certain behaviour of the system and where the consequence is located. Number 6 in Figure
11. A relationship from a consequence to the System structure. The corresponding System structure which will be the structural scope of the consequence. Number 7 in Figure.
12. A relationship from a consequence to corresponding process. Process that represents the behavioural Scope of it. Numbered 8 in Figure.
13. A relationship from every process to a subProcess and a superProcess. Numbered 9 in Figure.
14. A relationship from every part of a System with respect to another in a hierarchical way. Number 10 in Figure.
15. A relationship from a process to all the parts of the System that play a role in that behaviour. Number 11 in Figure.

Making these relationships explicit and keeping a record of it will help the future hazard analysis of the system in the tasks of joining separate hazards analysis of the same system. In addition, the maintenance process of the resulting hazard analysis will be enhanced.

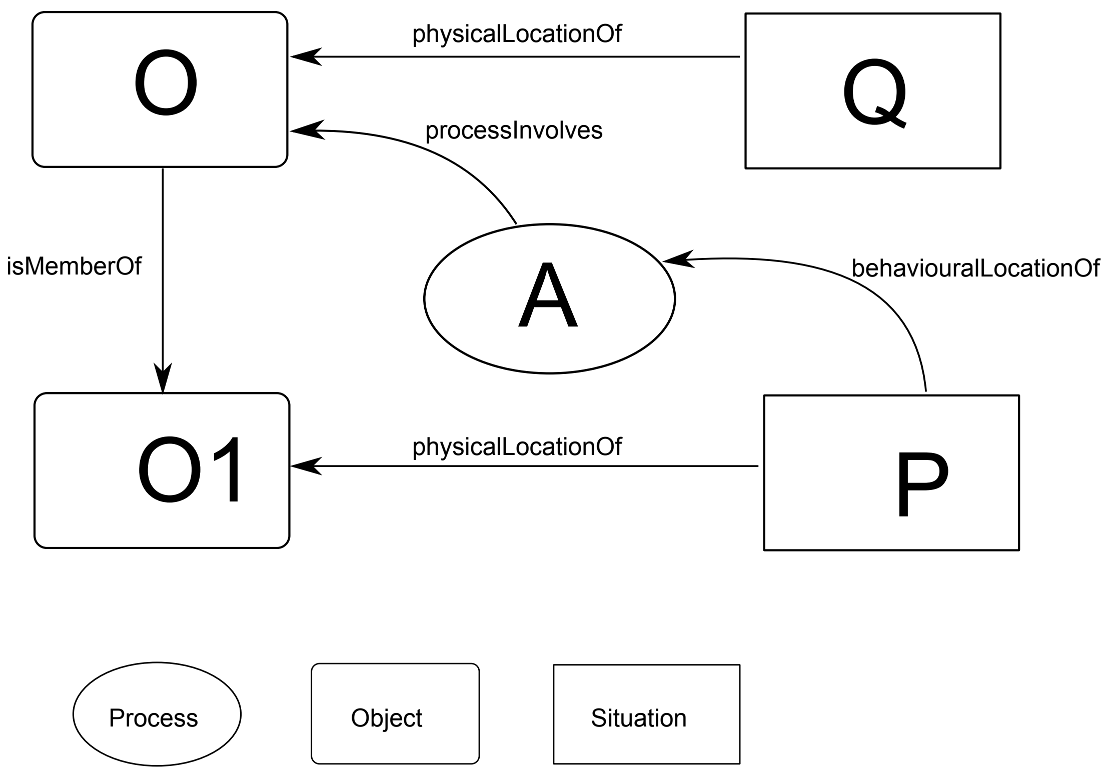


Figure 4.8: Similarity

4.3.4 Stage 4: Search implicit and overlooked relationships

This stage is dedicated to the identification of possible indirect causal and overlooked relationships. The intention is to use the structured and automated reasoning provided by tools for the analysis of ontologies. Ontologies can be used to reason about the dependencies of hazards, causes and consequences. The purpose is to help with the analysis of hazards with the aims of making them more precise, disambiguating causal relationships, and supporting the proper definition of system boundaries. Our analysis process is supported by a foundational ontology that can help analysts to find indirect causal relationships, and to disambiguate identified hazards and causes. The ontology, how has been created and what represent is described in Section 4.2 and this section uses different concepts described in there. Section 4.2.4 explains the ontology rules and the relationships that need to be found: Entailment, Synonymy and Similarity. When these relationships are found, the safety analyst could explore the possibility to interlink the results using non-direct causality (*isIndirectCauseOf()*).A non-direct causality is a causality relationship that has been discovered while analysing the hazard analysis with the ontology rules.

Figure 4.8 shows the relationships that are looked for in order to discover if two hazards (or causes, or consequences),in this case P and Q, are related: Similarity. Hazards, Causes and Consequences are considered subclasses (subsets) of the class Situation. The location of a Hazard, Cause or Consequence is represented by the *physicalLocationOf*(a, b) relationship or

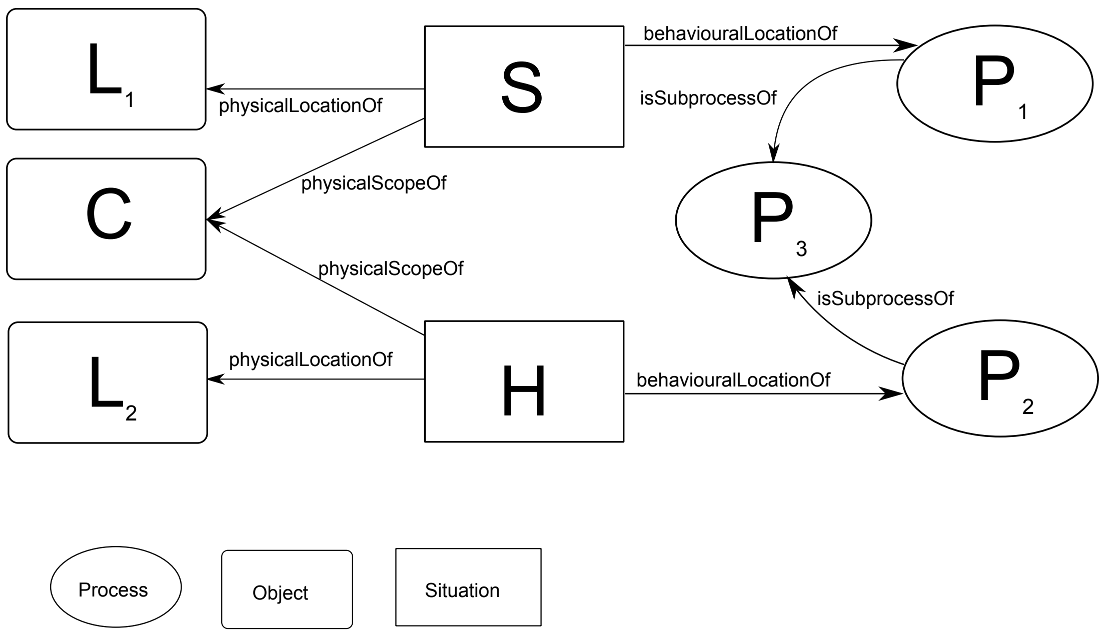


Figure 4.9: Entailment

property. In the ontology, the behaviour of the system is described via processes; a process may involve various objects (*processInvolves*(e, f) property). An object is anything that is used to describe a system and its parts (or members). The property *behaviouralLocationOf*(c, d) record the process(es) that could be directly affected by a hazard (or cause or consequence).

Figure 4.9 describes the relationships that are looked for in order to discover if S is an entailment of H or viceversa. The property *physicalScope* refers to the boundary of the hazard. As explained earlier, if a hazard affects a determined system, that system will be the scope of the hazard. The property *isSubProcessOf* exemplifies that a process can be decomposed into subprocesses. What we want to express is that if S and H have the same *physicalScope* and if their respective *behaviouralLocationOf* are subprocesses of the same process, one might entail the other. Section 4.2.4 shows Figure 4.9 in rule format.

Figure 4.10 refers to the synonymy relationship, this means a rule that is used to try to find synonyms in the ontological representation. The idea is to focus on all the possible relationships that our situations (hazards, cause, consequences) have and compare them in order to see if two situations have the same results when all these properties *physicalLocationOf*,behaviouralLocationOf, behaviouralScopeOf, *physicalScopeOf* are applied. Final decision is left to the Safety Analyst.

Processing the results

After a search has been performed in order to find synonymy, entailment and similarity relationships. Now, the results of these searches will be processed. In a nutshell, if two hazards, or a cause and a hazard, seem to have a similar, or close, meaning then it could be found out what

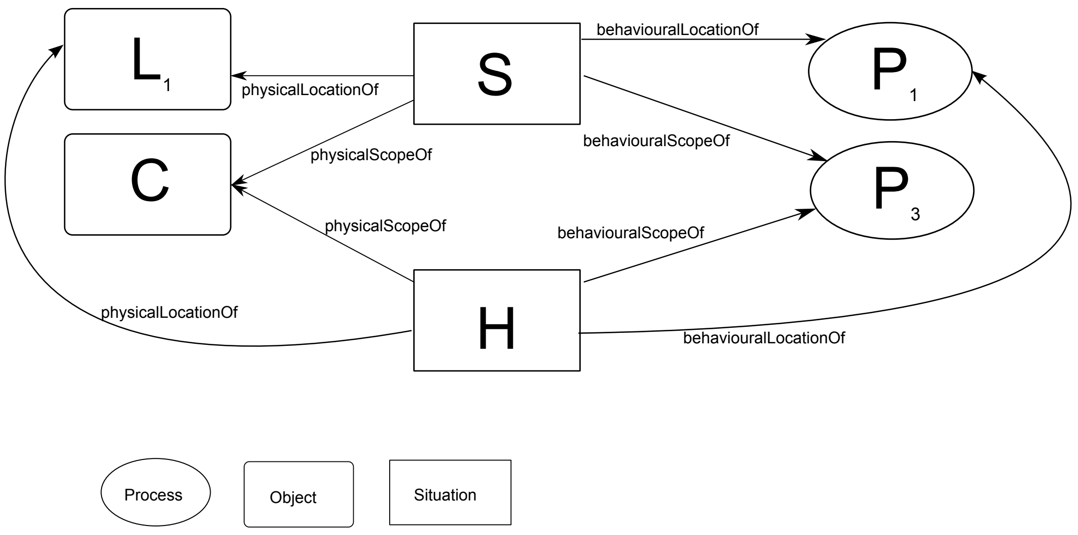


Figure 4.10: Synonymy

those hazards affect and where those hazards are allocated. If it is concluded that two hazards are actually the same hazard but expressed in different words then there are two alternatives, either we re-edit them and make just one hazard out of them with the changes that this will imply to the dependencies to these elements or we re-edit them following the algorithms given below.

In order to follow the algorithms, first, it is important to recall that a hazard occurs on a boundary. A hazard can be cause of another hazard if, for example, a hazard H_1 occurs in a certain boundary and it is cause of hazard H_2 that occurs in another boundary that is the environment of H_1. In addition when a hazard H_1 is found to be cause of another hazard, hazard H_1 would be added to the set $CauseHazard$ of the ontology. When a cause is classified as a hazard then it will be added to the set $CauseHazard$ as well.

Algorithm 1 describes the procedure of finding relationships induced by situations (hazards or causes) that are synonyms, i.e., situations linked via $isSynonymWith$ relationship. Assume that Hazard H_1 and Cause C_1 are synonyms, Hazard H_2 and Cause C_2 are also synonyms, and C_1 is a cause of Hazard H_2. Then cause C_1 will be transformed into a **CauseHazard**, all causes of H_1 will be indirect causes of C_1, the relationship is $isIndirectCauseOf$, and Hazard H_1 will be indirect cause of H_2 and C_2. Furthermore, assume that hazards H_1 and H_2 are synonyms then causes of H_1 will be indirect causes of H_2 and vice versa.

Algorithm 2 describes the procedure of finding relationships induced by situations (hazards or causes) where there exists an entailment. Assuming that H_1 entails S_1 and S_1 is a cause of Hazard H_2 then there is a chance that H_1 is an indirect cause of H_2. If so, the relationship $isIndirectCauseOf()$ should be added. In addition, if H_1 is a hazard then we need to add H_1 to the set of CauseHazards. Moreover, if S_1 is a Hazard, we still should analyse S_1 with relation to H_1 in order to understand if any relationship between them went missing.

Algorithm 3 describes the procedure of analysing if two situations (hazards or causes) that

Algorithm 1 `ProcessingSynonyms()`

```
1: $H_1$ is a Hazard
2: $C_1$ is a Cause
3: $H_2$ is a Hazard
4: $C_2$ is a CauseHazard
5: if $H_1$ isSynomymWith $C_1$ then
6:     $C_1$ is a CauseHazard
7:     Causes of $H_1$ are indirect causes of $C_1$
8:     if $C_1$ is Cause of $H_2$ then
9:         $H_1$ isIndirectCauseOf $H_2$
10:    end if
11:    if $H_1$ isIndirectCauseOf $H_2$ and $H_2$ isSynomymWith $C_2$ then
12:        $H_1$ isIndirectCauseOf $C_2$
13:    end if
14: end if
15: if $H_1$ isSynonymWith $H_2$ then
16:    Causes of $H_1$ are indirect causes of $H_2$
17:    Causes of $H_2$ are indirect causes of $H_1$
18: end if
```

seem to have similarities and analyse if they could be actually synonyms, or have an entailment relationship, or another overlooked relationship that the analysis could have overlooked. If an $isIndirectCauseOf()$ relationship needs be added then we might need to add H_1 or S_1 to the set of CauseHazards.

An example of algorithm 1 could be as follows, Figure 4.13 shows an example where Cause c_1 is cause of Hazard h_1 and Cause c_2 is cause of Hazard h_2. Cause c_2 and Hazard h_1 are linked together because of the relationship *isSynonymWith*, Cause c_1 and Hazard h_3 are linked together with $isSynonymWith$ as well, as shown in Figure 4.11. Following the algorithm 1, all the relationships shown in Figure 4.12 will be created. All these previous and new relationships (*isCauseOfHazard*, *isIndirectCauseOf*), together with the elements of sets Hazard, Cause and CauseHazard, are the source of an initial model that can be transformed in a state based transition system to be representing the next steps of the method.

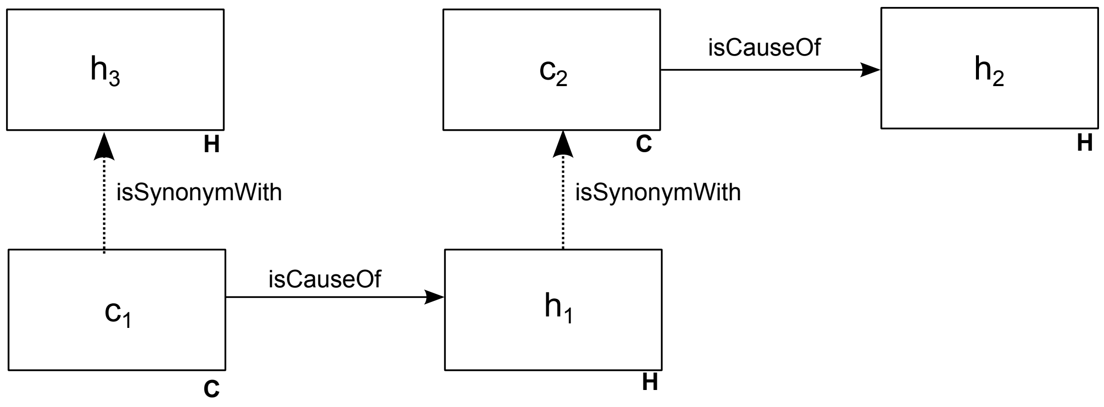


Figure 4.11: First: Synonymy Algorithm example

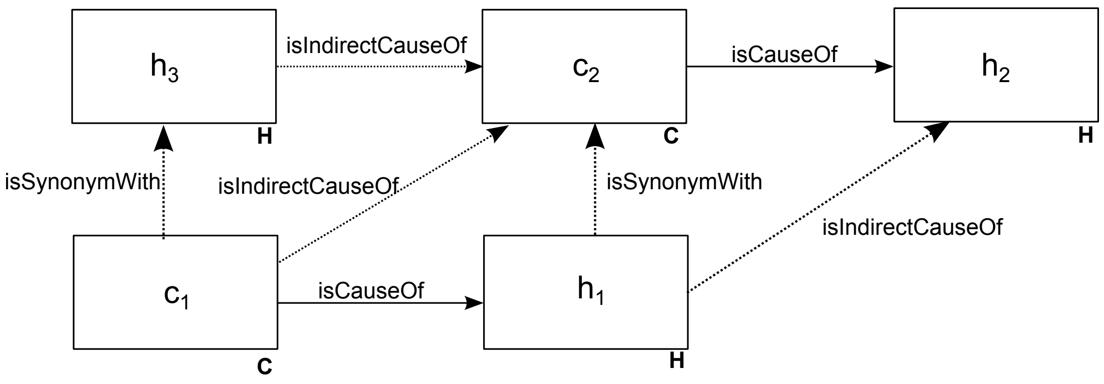


Figure 4.12: Second: Synonymy Algorithm example

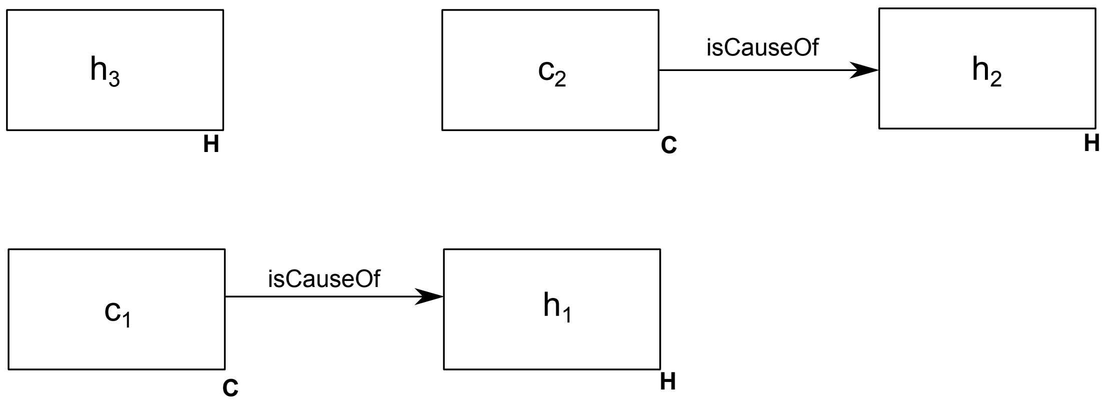


Figure 4.13: Third: Synonymy Algorithm example

Algorithm 2 `ProcessingEntailments()`

```
 1: $H_1$ is a Situation
 2: $S_1$ is a Situation
 3: $H_2$ is a Hazard
 4: if $H_1$ entails $S_1$ then
 5:     if $S_1$ isCauseOfHazard $H_2$ then
 6:         Analise $H_1$ might be indirect cause of $H_2$
 7:         if $H_1$ isIndirectCauseOf $H_2$ and $H_1$ is a Hazard then
 8:             $H_1$ is a CauseHazard
 9:         end if
10:     else
11:         if $S_1$ is a Hazard then
12:             Analise any possible relationship between $S_1$ and $H_1$
13:         end if
14:     end if
15: end if
```

Algorithm 3 `ProcessingSimilarity()`

```
 1: $H_1$ is a Situation
 2: $S_1$ is a Situation
 3: $H_2$ is a Hazard
 4: $C_2$ is a Situation
 5: if $H_1$ mightRelateTo $S_1$ then
 6:     Analise $H_1$ and $S_1$ to confirm relationship
 7:     if $S_1$ isSynomymWith $H_1$ then
 8:         ProcessingSynonyms()
 9:     end if
10:     if $H_1$ entails $S_1$ then
11:         ProcessingEntailment()
12:     end if
13:     if $H_1$ holds a particular relationship with $S_1$ and $S_1$ isCauseOfHazard $H_2$ then
14:         $H_1$ might be indirect cause of $H_2$
15:         if $H_1$ isIndirectCauseOf $H_2$ and $H_1$ is a Hazard then
16:             $H_1$ is a CauseHazard
17:         end if
18:     end if
19: end if
```

4.4 Second Part of the Method: State based hazard model analysis

Hazard paths represent causal relations among hazards: if hazard H_1 is cause of hazard H_2 and H_2 is cause of hazards H_3, then there is a hazard path from H_1 to H_3. These paths are found from causal relationships (*isCauseOfHazard*, *isIndirectCauseOf*) identified in the ontological representation built in step 1.

The following sections will explain the second part of the method which is an analysis of the state based hazard model. Section 4.4.1 will explain the steps to obtain a state based hazard model. More specifically, Section 4.4.1 explains the transformation of the ontology relationships into transitions and it then explains the translation of this intermediate result into a transition system. Finally, Section 4.4.2 explains the analysis that can be done from our generated model.

4.4.1 Stage 5: Generation of State-Based Hazard Models from Ontology Rules

This is the stage where a state-based model is constructed suitable for the mechanised analysis of causal paths among hazards. Up to now, we have been interested to classify the Hazard Analysis information into elements of the classes $Hazard$, $Cause$ and $CauseHazard$. Class $CauseHazard$ contains all those elements that belong to both classes $Hazard$ and $Cause$.

After the hazard analysis has been revisited and probably edited, new relationships can be created. In this context we will have elements in the Cause set that will belong to the Cause-Hazard set as well. The relationship between the elements in the Cause set with the elements in the Hazard set were given by the properties $isCauseofHazard$. The relationships among those sets will have a new property called $isindirectCauseOf$. In addition, the relationships considered to be extracted for us will be:

1. $CauseHazard\ isCauseOfHazard\ Hazard$

2. $CauseHazard\ isIndirectCauseOf\ Hazard$

3. $CauseHazard\ isIndirectCauseOf\ CauseHazard$

4. $Hazard\ isIndirectCauseOf\ Hazard$

These CauseHazard set and Hazard set and their causal relationships would be then transformed into a transition system. Figure 4.14 could be called a Transition System. Figure 4.15 shows a set of states and paths that is a transformation from Figure 4.14. Technical details of the transformations and translations in Section 4.5. The model is encoded in the modeling language ISPL [LQR09], and properties of interest are expressed in CTL Logic.

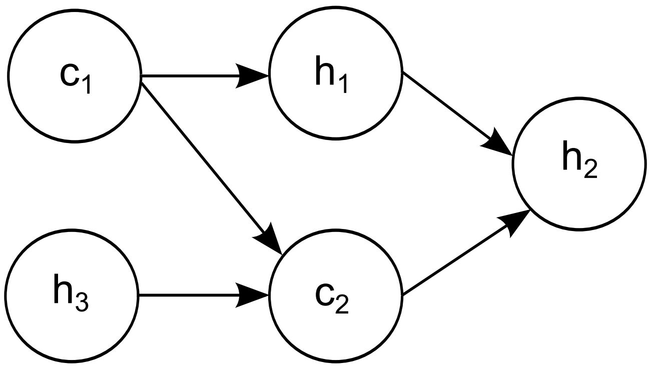


Figure 4.14: Transitions

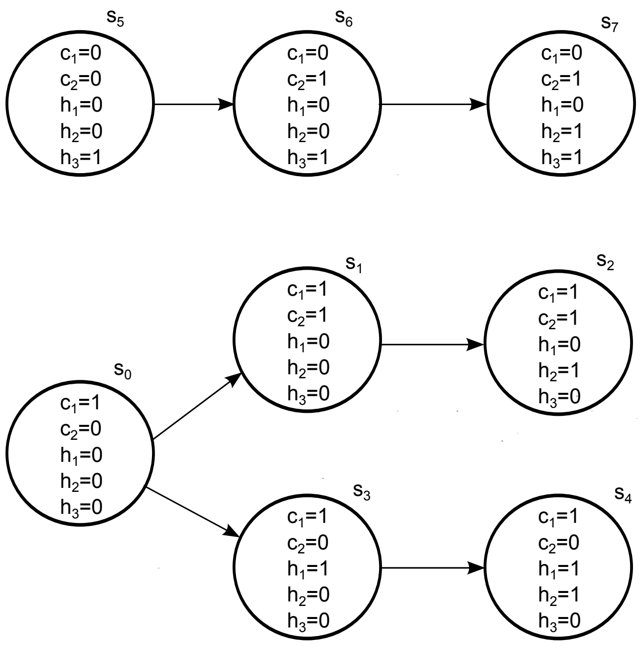


Figure 4.15: Paths

4.4.2 Stage 6: Exploration of the state-based hazard model

It is in this stage, where a formal methods tool, the MCMAS model checker [LQR09], is used for automatic analysis of the hazard model. Section 4.5.3 shows more technical detail about this search. Our technique aims to help with hazards analysis, specially, at the Hazard Identification stage, which is performed relatively early in the system development stage. The system representation we support in our ontology could perfectly be a representation of an early stage of the system representation. So, this stage contributes to more exploration of the hazard analysis and provides more feedback to the hazard analysis in very early stages. So the hazard path is a way to identify where we need to know more. This "to know more" means to explore the ontology in order to get more knowledge, such as what failed and how failed. For example, we could answer questions like:

- How a hazard was triggered.

- Which is the furthest cause.
- Where this furthest cause is located.

4.4.3 Stage 7: Visualizing Hazard Paths

In this stage the hazard paths can turn into visual hazard paths with more detail about the name of the hazards and the way they develop. In doing so, the hazard paths turn into scenarios that give more input to the hazard analysis under study. For example, a common cause for different hazards could be highlighted, or a hazard could be reclassified according to severity or frequency. The various case studies that are going to be presented will provide examples about this stage.

4.5 Developing a Prototype

This Section provides the detail of the transformation from the set of hazards, causes that are also hazards ($CauseHazard$) and the causal relationships that relate them, which are initially in OWL format, into the modeling language ISPL, in order to use the MCMAS model checker and find hazard paths. It is important to remark that the first step of this data transformation, which extracts from the ontology only the hazards, cause hazards and their causal relationships into an intermediate format(see Table 4.4) is mechanised but needs to be implemented in a tool.

4.5.1 Overview

Ontologies are used to represent the system under study and its hazards analysis. This seems a natural way to represent all the information collected from the Hazard Identification Process in a structured manner. In addition, model checking is used to analyse how hazards develop by finding hazards paths. The information, captured by previous Hazard Identification techniques and safety management processes, is saved in an ontological structure, which allows the safety analyst to create relationships and from these relationships perform inferences. In turn, these inferences help the analyst to refine the information that is already there. When the refinement process has finished a transformation from these ontological information is performed in order to get a transition system than later can be explored using a model checker.

4.5.2 Translating from the intermediate format into a Transition System

Up to now, the interest was to classify the information from the hazard analysis into elements of the classes $Hazard$, $HazardCause$ and their causal relationships. Class $HazardCause$ contains all those elements that belong to both classes $Hazard$ and $Cause$. Table 4.4 shows the sets of $Hazard$, $Cause$ and $CauseHazard$ of an ontology O. These elements are an intermediate representation, in the real ontological representation there are the textual description of the hazards, for purpose of the exploration of the hazard model the IDs of the hazards would be use. Table 4.5 shows only elements of the sets $Hazard$ and $CauseHazard$ linked by a causal relationship, extracted from Table 4.4.

These raw results are a set of tuples T such that $(a, b) \in T$ where $a \in Z$ and $b \in W$, W is the set of hazards and Z is the set of cause hazards that belongs to the results. In addition, $Z \subseteq W$. All the elements of T are shown bellow:

$$\begin{aligned} T = \{&(STA-4, STA-1), (STA-1, A-2), (STA-1, A-1), (A-2, A-1), \\ &(A-2, A-3), (A-1, A-3)\} \end{aligned}$$

4.5.3 Hazard Model

At this point, sets T and W are the initial results but we need to give semantics to these results. They need to be translated into a Hazard Model. In order to do so, first, it needs to be clarified that all hazard elements and causes (that are hazards) are part of every state of the system. This means that a single state of the Hazard Model is defined by every single value of every hazard in set W.

To reach the final Hazard Model there will be some intermediate translations. The first translation is a translation from T and W to the conceptual level, by conceptual level is meant the abstract representation of the system, then we there will be another translation to an implementation level of the transition system.

Conceptual level

The first translation is guided by the following algorithm, where the input for the algorithm would be the set of hazards W, the set of pairs T and the set of cause hazards Z and the result would be an unlabelled transition system M.

The result of this translation is M. The unlabelled transition system M can be seen in Figure 4.16. The explicit representation of M is as follows:

1. First, the Atomic Propositions:

$$AP_{TS} = \{STA-4, STA-1, A-1, A-2, A-3\}$$

2. Second, a finite set of states of the System:

$$S_{TS} = \{STA-4, STA-1, A-1, A-2, A-3\}$$

Algorithm 4 `GettingUnlabelledTransitionSystem()`

```
1: Data Set of Hazards $W$, Set of Pairs $T$, Set of CauseHazards $Z$
2: Result An unlabelled transition system $M$
3: Let $AP_{TS}$ be the set of Atomic Propositions
4: Let $S_{TS}$ be the set of States of the System
5: Let $I_{TS}$ be the set of Initial States
6: Let $R_{TS}$ be the Transition Relation
7:
        $$AP_{TS} := W$$
8:
        $$I_{TS} := \{x \in W \mid \nexists y \in Z : (y, x)\}$$
9:
        $$R_{TS} := T$$
10:
        $$M := (AP_{TS}, I_{TS}, S_{TS}, R_{TS})$$
```

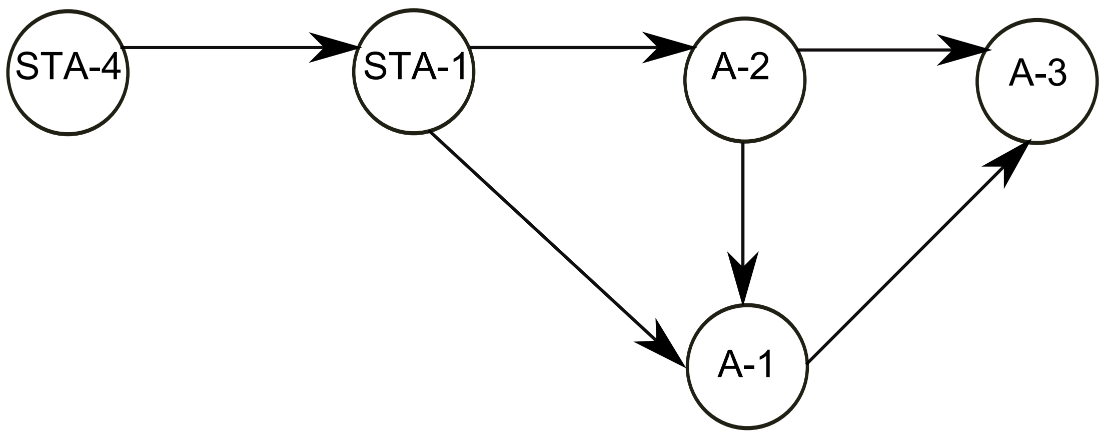


Figure 4.16: Unlabelled Transition System

3. Third, the set of initial states $I_{TS} = STA - 4$

4. Fourth, the transition relation:

$$R_{TS} = \{(STA-4, STA-1), (STA-1, A-2), (STA-1, A-1), (A-2, A-1), (A-2, A-3), (A-1, A-3)\}$$

Implementation level

At the implementation level, the unlabelled transition system M is translated into a kripke structure K. A Kripke structure is a variation of nondeterministic automaton used in model checking to represent the behavior of a system. A Kripke structure consist of a set of atomic propositions AP, a set of states S, a set of initial states I, a transition relation R and a labelling function L.

The translation from M to K is as follows

1. First, the set of atomic propisitions is defined. An atomic proposition can be either true or false. A hazard is represented with an atomic proposition, if the atomic proposition is true means that a hazard is active. The set of atomic propositions is $AP := AP_{TS}$.

2. Second, a state in K is defined. A state s in K is an $n - tuple$ of boolean variables where $n := |AP_{TS}|$, $s = (b_0, \ldots, b_{n-1})$, and for all $0 \leq i \leq n-1 : b_i \mapsto 1 \iff h_i \in AP_{TS}$ evaluates to true and where $h_0, h_1, \ldots, h_{n-1}$ are distinct.

3. Third, the set of states is defined. The approach to represent the states of K is to use the cartesian power of values of all the atomic propositions in any state s of the Kripke structure K. See bellow:

$$S := \{(b_0, \ldots, b_{n-1}) : b_i \in Boolean \text{ for all } i = 0, \ldots, n\}$$

4. Fourth, the set of initial states is defined. See bellow:

$$I := \{(b_0, \ldots, b_{n-1}) : b_i \mapsto 1 \iff h_i \in I_{TS} \; and \; h_i \notin I_{TS} \iff b_i \mapsto 0, 0 \leq i \leq n-1\}$$

5. Fifth, a transition relation is $R \subset S \times S$ where $\forall s \in S, \exists s' \in S$ such that $(s, s') \in R$. The transition relation is defined as follows:

$$R := \{(s,t) : \exists b_i \in s, b_j \in t \; and \; (h_i, h_j) \in R_{TS} \; and \; (\forall b_k \in t : b_k \mapsto 1 \iff b_k \mapsto 1 \in s)\}$$

6. Finally, the labelling function L and kripke structure K is shown in Figure 4.17.

Figure 4.17 visualize the sets of K. In this case:

1. The set of initial states is $I = s_0$

2. The set of reachable states $S_1 \subseteq S$:

$$S_1 = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$$

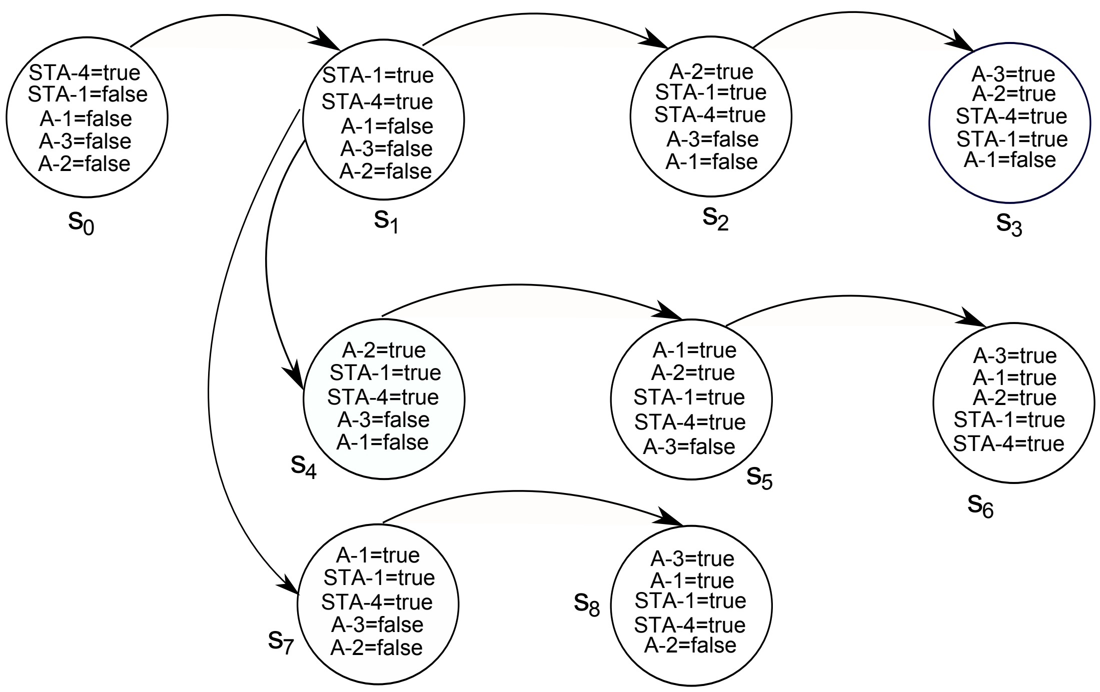


Figure 4.17: Transition System for our Example

3. In addition, the transition relation R:

$$R = \{(s_0, s_1), (s_1, s_2), (s_1, s_4), (s_1, s_7), (s_2, s_3), (s_4, s_5), (s_5, s_6), (s_7, s_8)\}$$

ISPL language representation

The code 4.1 shows an example of how the transitions will look when translated into the ISPL language [LQR09]. This translation is needed in order to search for hazard paths. First, it needs to be determined the $Hazards$ and $CauseHazards$ that are part of the initial state space, then the transitions need to be specified. In the example, this will be given by the following code $h_1 = true\ if\ h_0 = true$ which means, in the current state if variable $h_0 = true$ then there will be a transition to another state where variable $h_1 = true$. The initial state of the system is when when all variables are not active, i.e., $false$ but h_1. The idea is to search a path to h_2, this is specified in the Formuale and Evaluation sections of the code.

Listing 4.1: Example of an ISPL file

```


file  Transitions  {
 Agent: Environment;
 vars:  h_0:bool
        h_1:bool
        h_2:bool
```

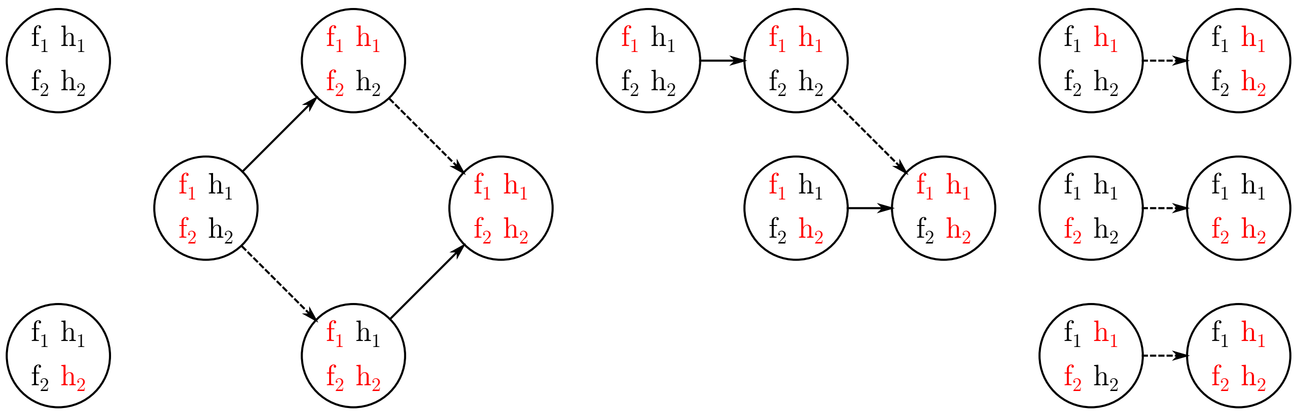


Figure 4.18: State space of causes *f1* and *f2* and hazards *h1* and *h2*

```

 Evolution:
         h_1=true if h_0=true;
         h_2=true if h_1=true;
 end Evolution
end Agent
Evaluation:
    target if Environment.h_2= true;
end Evaluation
InitStates
    Environment.h_0= true and Environment.h_1=false and Environment.h_2=false
end InitStates

Formulae

EF target;

end Formulae


};
```

Figure 4.18 illustrates a small subset of the state space of a system. We consider only hazards *h1* and *h2* with their causes (if they are CauseHazards). The black letters indicate that a causehazard or hazard is not active in in the state and the red letters indicate that they are. The arrows then represent transitions between the states when certain actions occur in the system. This way, a hazard will be active in the following state and in the way a hazard path is generated.

Searching for paths

Once there is a Hazard Model, it is possible to make use of the knowledge encoded by it.

- To start with, it is possible to ask questions such as:

  is there a way, that I am not aware of, to reach hazard $A - 3$ (which I recognise it has a great impact on the system)?

- A more formal way to write the previous question is:

  Is there a path in the Hazard Model that leads to a state where hazard $A - 3$ is active?

- The questions will be represented in Temporal Logic (TL). So, this question needs a translation into TL, an intermediate translation is:

  Is there a path where the following assertion $System.state.H = true$ holds?

  Where $System.state.H = true$ represents the state of the system that being is being verified, in this case, the state of the system where hazard $A - 3$ is active($A - 3 = true$); this will be expressed it in the following way `hazard1 if Environment.h = true;`

- The whole translation from the previous statement in Computational Tree Logic (CTL) is:

  $$\mathsf{EF}P \text{ where } P \equiv (System.state.H = true)$$

  The operator E stands for Exist and F stands for $\mathsf{Finally}$. If there is the following atomic proposition "I like chocolate" which will called P then $\mathsf{EF}P$ stands for "It is possible I may like chocolate some day, at least for one day".

  Back to the current example, $\mathsf{EF}P$ means that "It is possible that there is at least one way to hazard A-3".

- In order to use Linear Temporal Logic (LTL), a translation from CTL to LTL is needed. This is, $\mathsf{EF}h \equiv \neg\mathsf{AG}\neg h$ and it has to be verified whether $\mathsf{G}\neg h$ is not satisfied in the model.

A path obtained, applying this property or question to the Hazard Model is:

$$S_0 \longrightarrow S_4 \longrightarrow S_5 \longrightarrow S_6$$

Table 4.4: Classes and Elements of its class

Element	Class
STA-1	Hazard, HazardCause
STA-4	Hazard, HazardCause
A-2	Hazard, HazardCause
A-1	Hazard, HazardCause
A-3	Hazard
Ca-1	Cause
Ca-2	Cause
Ca-3	Cause
Ca-4	Cause
Ca-5	Cause
Ca-6	Cause
Ca-7	Cause
Ca-8	Cause

Table 4.5: Hazards and CauseHazards for modeling

`?x`	`?y`
STA-4	STA-1
STA-1	A-2
STA-1	A-1
A-2	A-1
A-2	A-3
A-1	A-3

5 Evaluation Framework

In this chapter, we reflect on the proposed method, its case studies and the evaluation of the method. The evaluation framework for the proposed methods is presented and a discussion is given about why empirical studies of how hazard analyses are performed in practice has not been part of the current research.

5.1 Summary

The way the method was evaluated is the following:

- The inputs were the various documents that helped to construct the hazard analysis as well as the documentation of the systems under study.
- The outputs were the stages of the method that were performed by each case study.
- The outcome was the contribution to the proposed method, each Stage of it, and to the refinement of the hazard analysis and the description of the system. In addition, new graphical representation of the system under study has been proposed. Moreover, various ways to document the results of the refined hazard and the system in table forms have been rehearsed.
- The potential impact is the improvement of the hazard analysis results and the documentation of the hazard analysis performed. The intense review of the system under study that might result in a better understanding of it.

The inputs are going to be presented more explicitly in the following chapters (see 6, 7, 8, 9, 10) where the case studies are going to be introduced. The potential impacts are going to be discussed more extensively in chapter 11. The outputs and the outcome are going to be discussed in the next sections.

5.2 Outputs

Outputs refer to tages that were applied in each case study. Various case studies have been used to assess the utility of the developed method. It has been an outgoing work where there has been a lot of feedback in order to improve the proposed method.

Chapter 6 refers to a Medication System Case Study where the hazard analysis has been developed from various sources of research about safety analysis in hospitals. A system structure and a behavioural model of the system has been created as well.

Chapter 7 refers to the Generic Infusion Pump User Interface (UI) Case Study. A Preliminary Hazard Analysis (PHA) has been developed for this case study.

Chapter 8 refers to the Prêt á Voter Case Study where a rigorous HAZOP study was already performed and where a system description was already provided. We have used these inputs in the proposed method.

Chapter 9 refers to a Railway System Case Study where the Hazard Analysis was taken from a Preliminary Hazard Analysis (PHA) already performed. The system structure needed to be created from the information given in the system documentation that was available.

Chapter 10 refers to the ITP Case Study where the In-Trail Procedure (ITP) procedure is analysed.

In most of these case studies the system description was not detailed. There were no complete and detailed models attached to the hazard analysis. In the Railway system, the initial study and the documentation did not consider a detailed system description but an enumeration of the different subsystems of the railway system and their respective description in text form. Prêt á Voter sources considered a more detailed description of the system and tasks, however, some components were not analysed in deep but in a superficial way. The Infusion Pump UI had a decription of the UI protoype in the early stages. All of these system descriptions had different ways to be represented, either graphically, textually, or both. The Prêt á Voter and Infusion Pump UI Case Studies provided a graphical model. The rest of the case studies only had textual system descriptions. Table 5.1 shows the case studies that have been presented and which stages of the proposed method have been applied in each case study.

Table 5.1: Stages of the method and what was applied in each case study

Stages vs Case Studies	Medication System	Infusion Pump UI	Prêt á Voter System	Railway System	ITP
Inputs: Hazard Analysis, System Description	Done	Done	Done	Done	Done
Systematic revision of inputs	Done	Done	Done	Done	Done
Ontology guided revision of inputs	Done	Done	Done	Done	Done
Search for implicit and overlooked relationships	Not Done	Not Done	Done	Done	Done
Generation of the State Based Hazard Model	Done	Done	Not Done	Done	Done
Exploration of the State Based Hazard Model	Done	Done	Not Done	Done	Done
Significant Hazard Paths	Done	Done	Not Done	Done	Done

5.3 Outcome

Here we discuss contribution of the case studies to the proposed method and to the refinement of the hazard analyses and systems. The method has been developed and improved together with the case studies. The case studies have provided feedback that contributed to the development of the proposed method. The various results provided feedback that helped to settle the various stages and arrive to a final version of the proposed method.

The Medication System Case Study (see Chapter 6) and the Generic Infusion Pump User Interface (UI) Case Study (see Chapter 7) explored the suitability of the ontological representation in order to generate a state based hazard model that would be able to find paths to determined hazards. This way, scenarios would be generated. In addition to that, those case studies helped to contribute with the constructions of the notation of the graphical representation of the system under study.

The Prêt á Voter Case Study (see Chapter 8) explores a voting system that mixes electronic vote with traditional voting for elections. The hazard analysis for this system has been done in HAZOP and it presents a refined version. The behavioural documentation of the system is quite detailed in diagrams and in text form but the physical representation is not detailed enough with regards granularity but it is possible to get a physical representation from the documentation. This case study is important because the hazard analysis is not a preliminary version but a is a final refined version. In addition, it was performed by outside parties, no related to the thesis author or for the purposes of this thesis.

Railway System Case Study (see Chapter 9) explores a railway system. This Case Study is important for two reasons. First, it is one of the case studies where all the stages of the method were applied. Second, it was a case study that explores a Preliminary Hazards Analysis done by a safety group with the intention of developing the railway system. So, this hazard analysis was not performed by author of the thesis.

With the Prêt á Voter Case Study and the Railway System Case Study the ontological representation of the systems under study and its respective hazard analyses were reinforced. In addition, the respective graphical representation of the systems with the proposed notation was strengthen. This because of the versatile way in which the hazard analyses and the systems were represented with the ontology. But more importantly, because of the results obtained from the Stage 4 of the method where, in the case of the Prêt á Voter Case Study, synonym hazards were found and consequently the hazard analysis and the hazard descriptions were refined. In Stage 4 of the method applied in the Railway System Case Study, synonym and entailment relationships have been found; therefore, this initial PHA could also be refined.

One of the difference between the Prêt á Voter Case Study and the Railway System Case Study is that the former counts with a refined version of its hazard analysis which is performed with HAZOP and the later counts with an initial version of a hazard analysis, performed with the PHA technique.

In the case of the Railway System Case Study all the stages of the method were applied. This means that Stages 5, 6, 7 were also applied and a state based model was created which could be explored and hazard paths to selected hazards were found.

Chapter 10 refers to the ITP Case Study. This Case Study explores a procedure in controlled airspace ITP. The hazard analysis has been performed by the author of the thesis, it was done in PHA, and the initial results were compared with the results obtained from the STPA Hazard Analysis to show the level of detail at which PHA was performed. In the same way than the Railway System Case Study, this is another case study where all the stages of the proposed method were applied. All the documentation in which the representation of the system was based are detailed in stage 1 of the method described in 10. Stage 4 of the the method applied to this case study showed a few entailment relationships were found and therefore the PHA was also able to be refined. In addition, the following stages were also performed where a state based model was created which could be explored and present hazard paths to selected hazards.

Moreover, in this 10 case study various tables are presented and where the infornation is based on the ontology representation of the system, the hazard analysis and their respective relationships. Those tables have the STPA table form. This was to show how based on the knowledge represented on the ontology various ways to present the information to the safety analyst and for documentation purposed can be rehearsed in order to improve the analyst comprehension of the hazard analysis and its results and what they might imply for the safety of the system.

In case of the Medication System Case Study it was a socio-technical system and in the case of the Generic Infusion Pump User Interface (UI) Case Study what was studied was a a software user interface, one belongs to the healthcare industry and the other to thesoftware for medical devices industry. So, various industries are being tackled.

5.4 Empirical Studies

Empirical studies of how hazard analysis are carried out in practice have not been considered into the research being conducted. This is because the research that has been conducted was aiming to, first of all, consolidate an ontology for the representation of the system, the hazard analysis and their dependencies. In addition to that, the ontology itself needed to be evaluated by means of the various instantiations of it. Together with this came the consolidation and validation of the graphical representation of the system in the structural and behavioural forms, this again, by means of the various case studies done and where the graphical notation was applied.

Second of all, the translation of the ontological representation of the hazard analysis needed to be translated into the state based hazard model. This also needed to be applied and tried and it has been. Results have been found and have been explained in the case studies. Finally, the whole method need to get feedback, through the case studies, in order to arrive to the current version.

Nevertheless, empirical studies about how hazard analysis are performed in practice must take part in future work in order to contribute to the feedback given to the proposed method as a whole. Moreover, it can also contribute to improvement of the framework in which the proposed method can be evaluated.

6 Medication Prescription System Case Study

This case study considers a typical workflow used in hospitals to prescribe the intravenous injection of a medication to a patient. This chapter is organised as follows. First, a system overview is given. Second, the hazard analaysis of the Medication Prescription System is detailed. Third, the stages of the method that have being applied to the case study are explained. Finally, a discussion about this chapter is given.

6.1 System Overview

Prescribing a medication to a patient involves the following steps:

- The medical practitioner writes a prescription for the patient and gives it to a nurse.
- The nurse takes the prescription to the pharmacy.
- The pharmacy produces a labelled medicine.
- The nurse brings the prescription and the labelled medicine to the patient's room.
- The nurse uses the prescription to program an infusion pump, connects the pump to the patient's veins, and starts the infusion.

During the last step, calculations may be performed by the nurse to convert information indicated in the prescription in a format suitable for the infusion pump. For example, this is necessary when the units used in the prescription differ from those used by the medical device.

6.1.1 Hazard Analysis of the Medication Prescription System

Table 6.1 shows the Preliminary Hazard Analysis for the Medication Prescription System. The first column shows the ID of the hazard, the second column shows the hazard description, the third column shows the potential causes of the hazard (note that a hazard can also be the cause of another hazard), and the last column are the possible consequences of the hazards described.

Table 6.1: Preliminary Hazard Analysis

Item	*Hazard Description*	*Potential Cause*	*Consequences*
H-1	Overdose due to wrong dosage	H-2 Pump programmed in accordance with miscalculation C-1 Correct calculated flow rate of pump introduced wrongly C-2 Nurse unadvertedly programmed pump with wrong volume of drug	Patient's life at risk
H-2	Pump programmed in accordance with miscalculation	H-3 Miscalculation	Overdose
H-3	Miscalculation	H-4 Ommited step in calculation C-3 False confirmation on label	Pump programmed in accordance with miscalculation
H-4	Ommited step in calculation	H-5 Complex calculation C-4 Nurse workload is stressful at all times	Miscalculation
H-5	Complex calculation	C-5 Not standarized information in this chemotherapy treatment C-6 Volume and rate of infusion not given by the general practitioner	Mistakes during calculation
H-6	Wrong delivery of drugs from pharmacy	H-8 Orders do not include administration process detail H-7 Mistake in name of drug fluid	Wrong drug dues to be administered to patient
H-7	Mistake in name of drug fluid	C-7 Close similarity in name of drugs	Wrong delivery of drug from pharmacy
H-8	Orders do not include administration process detail	C-8 Limited integration between disciplines and sites	Wrong delivery of drug from pharmacy

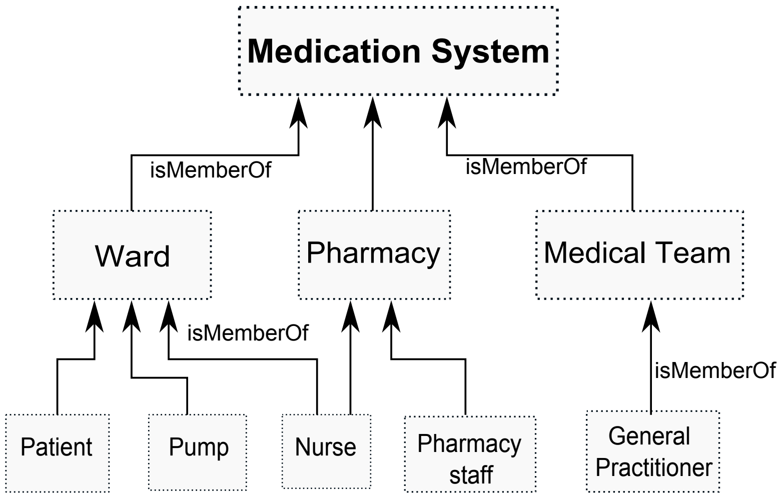


Figure 6.1: Medication Prescription System Structure

6.2 Application of the Ontology-Based Method to the Medication Prescription System Case Study

6.2.1 Step 1: Carry out a systematic exploration of dependencies

The first three steps of the analysis process are considered here. They are instantiated as follows:

- Stage 1: Collect information about the medication system we are describing.
- Stage 2: Revise the hazard analysis with regards to the information analysed.
- Stage 3: Here we develop the diagrams with the structural and behavioural representation of the Medication System.

Stage 1: Hazard Analysis and System Description We use, as source of information, documentation on infusion pumps publicly available on the website of the US Food and Drug Administration (FDA). We also used an accident investigation about a death by a sequelae from overdose. The technique used was incident root cause analysis that was done in a hospital in Canada [fSMPTC07a, fSMPTC07b].
Stage 2: Systematic Revision The hazard analysis was informed and checked with the results of a safety risk analysis carried out by others and publicly available in [WLAHR+09].
Stage 3: Ontology guided revision of inputs The behavioural and structural representations of the Medication Prescription System are in Figure 6.1 and 6.2.

6.2.2 Step 2: Perform an automated search of hazard paths

To perform the automated analysis of hazard paths, information about the hazard analysis and information contained in the hazard analysis spreadsheets is transformed into a formal model for the MCMAS model checker.

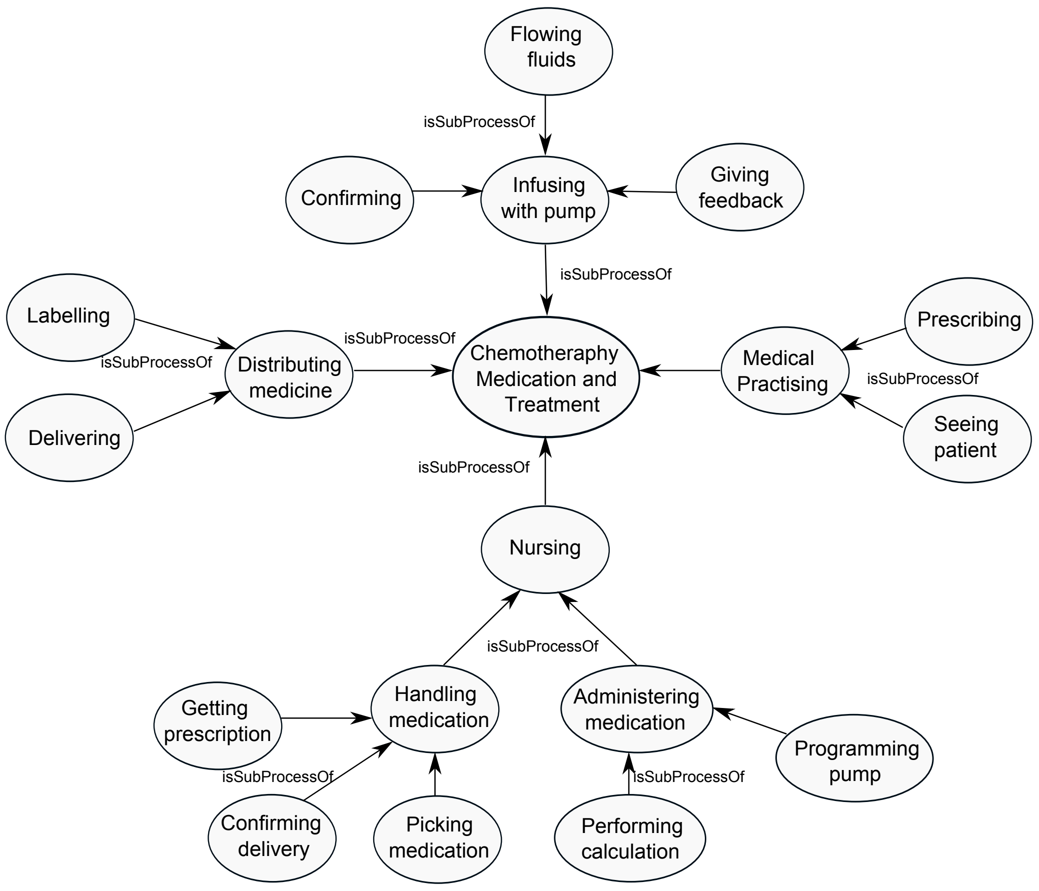


Figure 6.2: Medication Prescription System Processes

Stage 5: Generation of State-Based Hazard Models from Ontology Rules

In this stage the state based hazard model is developed. The possible relationships are:

1. $CauseHazard\ isCauseOfHazard\ Hazard$
2. $CauseHazard\ isIndirectCauseOf\ \ Hazard$
3. $CauseHazard\ isIndirectCauseOf\ \ CauseHazard$
4. $Hazard\ isIndirectCauseOf\ \ Hazard$

For the sake of simplicity, the focus here is only on relationships of the kind $isCauseOfHazard$. The same analysis process applies to the other relationships. The model can be built directly from Table 6.1. The initial results can be seen below:

- H-2 $isCauseOfHazard$ H-1
- H-3 $isCauseOfHazard$ H-2
- H-4 $isCauseOfHazard$ H-3

- H-5 $isCauseOfHazard$ H-4
- H-7 $isCauseOfHazard$ H-6
- H-8 $isCauseOfHazard$ H-7

Stage 6: Exploration of the state-based hazard model

At this stage, a formal methods tool is used for automatic analysis of the model. The MCMAS model checker is used. To do this, the model is encoded in the modeling language of the MCMAS model checker, and properties of interest are expressed in CTL Logic. The full MCMAS model can be found in Appendix A.

As an example, let us consider here the case where we want to check if there is a path to hazard *H-1: Overdose due to wrong dosage*. The CTL property necessary to perform this analysis can be expressed formally by the following CTL formula: $\mathsf{EF}\ h1$, where $h1$ is an atomic proposition representing hazard H-1.

Note that the property of interest was expressed in CTL rather than LTL because modal operators in LTL are implicitly universal over all paths and so existence of a path cannot be checked. However, this problem can partly be alleviated by considering the negation of the property in question, and interpreting the result accordingly. To check whether there exists a path from s satisfying the LTL formula φ, we check whether all paths satisfy $\neg\varphi$; a positive answer to this is a negative answer to our original question [HR04]. In our case, $\mathsf{EF}h1 \equiv \neg\mathsf{AG}\neg h1$ and we verify whether $\mathsf{G}\neg h1$ is not satisfied in the model.

Stage 7: Visualizing Hazard Paths

The analysis performed with MCMAS provided a path to H-1. The witness identified by the model checker is depicted in Figure 6.3, where circles represent system states, and arrows represent the transitions between states. The hazards that are active in each state are indicated in red. When there is a transition and a hazards becomes active, the colour of the hazard label changes from black to red.

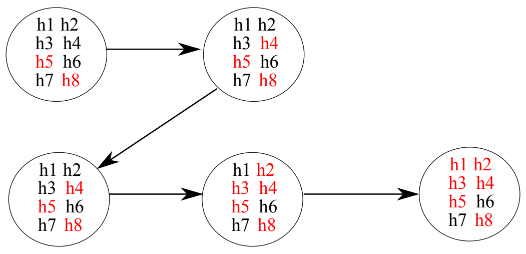


Figure 6.3: Data Entry Hazard Development

6.3 Discussion

This case study shows a simplified medication prescription system. The hazard analysis shown in Table 6.1 has been performed by the author of the thesis. An ontological representation of the prescription system and the hazard analysis has been done. The main motivation for the execution of this case study was to apply and evaluate the second step of the proposed method. It was aimed was to explore the utility of a state based hazard model that makes use of model checking tools such as MCMAS. A state based hazard model has been produced and it has been explored by a tool and it has produced hazard paths that can easily be transformed into scenarios that can add more value to the initial hazard analysis. Stage 4 of the method has not been presented here because the scale of the example is rather small for this stage and it does not provide significant results in this stage.

7 Infusion Pump Case Study

Infusion pumps are medical devices used to deliver fluids into a patient's body in a controlled manner. There are many different types of infusion pumps, which are used for a variety of purposes and in a variety of environments. For example, some infusion pumps are designed mainly for stationary use at a patient's bedside. Others, called ambulatory infusion pumps, are designed to be portable or wearable. These infusion pumps also work on different kind of environments or contexts (physical, social, technological), these contexts influence directly or indirectly on the failures that these devices can have and consequently the harm that they can cause to a patient. This chapter is organised as follow, Section 7.1 presents a system overview of the data entry of the infusion pump as well as the hazard analysis performed. Section 7.2 presents the use of the ontology based method for this case study. The final section of this chapter discussed the results from the application of the method.

7.1 System Overview

The specific focus of the analysis is the *data entry system* of the infusion pump, which is used by nurses to program the therapy to be delivered with the pump. A generic version of the User Interface Software of an infusion pump [ZMJT19] includes three main components:

- The **UI Logic** handles input events and interaction tasks.
- The **UI Output Status Manager** represents status variables used by the software to store information the user can interact with and communicates with the renderer.
- The **UI Renderer** updates the output every time a state update is notified by the UI output configuration manager.

7.1.1 Hazard Analysis of the Data Entry Subsystem

A preliminary hazard analysis is shown in Table 7.1. The main hazards are:

1. Misprogramming of quantity of medicine

2. Misprogramming of flow or rate of medicine

3. Misprogramming of parameter settings

Table 7.1: Preliminary Hazard Analysis

Item	*Hazard*	*Cause*	*Consequence*
H-1	Misprogramming of quantity of medicine	H-4 Confirmation button pressed with wrong data H-5 Wrong edition of number by decimal point	Overinfusion or underinfusion
H-2	Misprogramming of flow or rate of medicine	H-6 Pump performs a big jump H-4 Confirmation button pressed with wrong data	Patient receiving medicine to a rate or flow damaging to his health
H-3	Wrong selection of parameter settings	H-17 Lack of competence about current pump H-16 Design of menues confusing to user	Wrong edition of parameter settings
H-4	Confirmmation button pressed with wrong data	H-10 Confirmation messages display unclear and not substantive information	Pump programmed wrongly
H-5	Wrong edition of number by decimal point	H-7 Decimal point misplaced at editing due to general lack of visibility H-8 Lack of visibility of the decimal point	Pump programmed wrongly
H-6	Pump performs a big jump	H-9 Data entry system records keys pressed for too long and more strokes recorded H-18 Lack of knowledge about pump functionality w.r.t. big jumps H-17 UI Logic allows "big jumps" with single user actions during data entry	Editing of settings contains error
H-7	Decimal point misplaced at editing due to general lack of visibility	H-12 Characters and size of characters designed causes lack of visibility of UI	Wrong edition of number by decimal point
H-8	Lack of visibility of the decimal point	H-11 UI Renderer renders point of innapropiate size	Wrong edition of number by decimal point

Table 7.1: Preliminary Hazard Analysis

Item	*Hazard*	*Cause*	*Consequence*
H-9	Data entry system records keys pressed for too long and more strokes recorded	c-1 Keys malfunction c-2 Unintended slip caused by user	Pump performs a big jump
H-10	Confirmation messages display unclear and not substantive information	H-15 Design of messages to be displayed do not offer enough feedback	Confirmation button pressed with unadverted mistakes on data
H-11	UI Renderer renders point of innapropiate size	H-14 Point designed to be rendered smaller than minimum size recommended by HF75:2009	Lack of visibility of the decimal point
H-12	Characters and size of characters designed causes lack of visibility of UI	H-13 Characters designed to be rendered eithout considering size and format recommended by HF75:2009	Wrong edition due to cause of visibility of UI
H-13	Characters designed to be rendered without considering size and format recommended by HF75:2009	H-21 Lack of use of UI design principles for the infusion pump	Characters and size of characters designed causes lack of visibility of UI
H-14	Point designed to be rendered smaller than minimum size recommended by HF75:2009	H-21 Lack of use of UI design principles for the infusion pump	UI Renderer renders point of innapropiate size

Table 7.1: Preliminary Hazard Analysis

Item	*Hazard*	*Cause*	*Consequence*
H-15	Design of messages to be displayed do not offer enough feedback	H-21 Lack of use of UI design principles for the infusion pump	Confirmation messages display unclear and not substantive information
H-16	Design of menues displayed confusing to user	H-21 Lack of use of UI design principles for the infusion pump	Wrong selection of parameter settings
H-17	UI Logic allows ”big jumps” with single user actions during data entry	H-21 Lack of use of UI design principles for the infusion pump	Pump performs big jumps
H-18	Lack of knowledge about pump functionality w.r.t big jumps	H-19 Lack of competence about current pump H-20 Pump user manual omits a number of instruction of some menues and interactions	
H-19	Lack of competence about current pump	c-3 Different kind of pumps in hospital c-4 Pump currently in use omitted in training	User performs different mistakes while programming pump
H-20	Pump user manual omits a number of instruction of some menues and interactions	c-5 Printed manual designed without any particular method or standard	Lack of knowledge about pump functionality

Table 7.1: Preliminary Hazard Analysis

Item	*Hazard*	*Cause*	*Consequence*
H-21	Lack of use of UI design principles for the infusion pump	c-6 No particular standards for infusion pump design c-7 Lack of interest from the UI design team c-8 Time dedicated ro UI design was too short	Design of UI might contain flaws
H-22	Wrong edition of parameter settings	H-3 Wrong selection of parameter settings	Misprogramming of parameter settings
H-23	Misprogramming of parameter settings	H-22 Wrong edition of parameter settings	Patient getting the wrong treatment

7.2 Application of the Ontology-Based Method to the Data Entry Subsystem of the Infusion Pump

7.2.1 Step 1: Carry out a systematic exploration of dependencies

The first three steps of the analysis process are considered here. They are instantiated as follows:

- Stage 1: This stage involves collecting all information about the Data Entry Subsystem.
- Stage 2: This stage involves revising the hazards identified for the Data Entry Subsystem.
- Stage 3: In this stage involves developing the structural and behavioural representation of the Data Entry Subsystem.

Stage 1: Hazard Analysis and System Description The main source of information was documentation on infusion pumps publicly available on the website of the US Food and Drug Administration (FDA).
Stage 2: Systematic Revision The hazard analysis was informed and checked with hazards and causal factors described in a detailed incident investigation report involving data entry errors with an infusion pump [fSMPTC07a] and a generic version of the User Interface Software of an infusion pump [ZMJT19].
Stage 3: Ontology guided revision of inputs Figure 7.1 shows the structural representation of the Data Entry subsystem. This subsystem is part of the User Interface System of the infusion pump and it has three main components which are the *UI Logic*, the *UI Renderer* and the *UI Output Status Manager* (see Section 7.1)

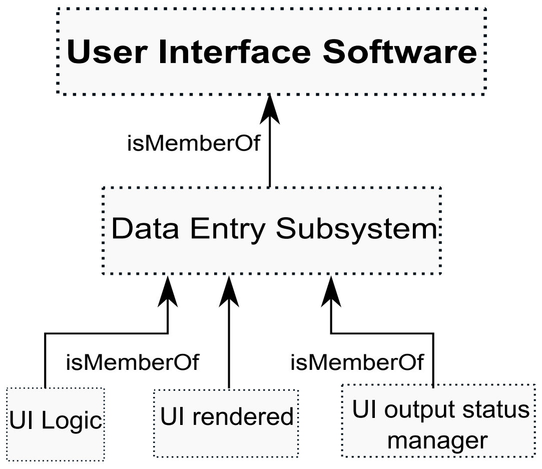


Figure 7.1: User Interface System, Data Entry Subsystem Structure

The behaviour of the Data Entry Subsystem is described in Table 7.2. It includes three main processes: *Select pump settings or infusion parameters*, *editing settings*,*confirmation and submission.*

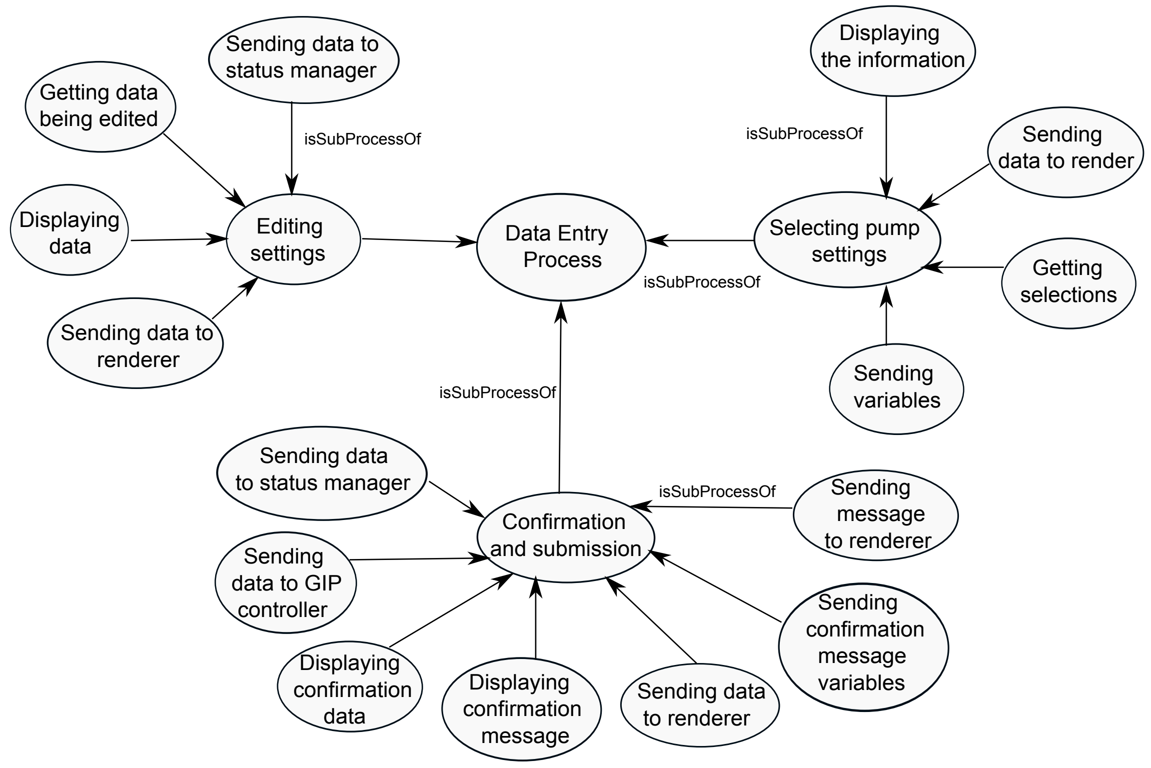

Sending data to status manager
Getting data being edited
Displaying data
Editing settings
Sending data to renderer
isSubProcessOf
Data Entry Process
Displaying the information
Sending data to render
Selecting pump settings
Getting selections
Sending variables
Sending data to status manager
Confirmation and submission
Sending data to GIP controller
Displaying confirmation data
Displaying confirmation message
Sending data to renderer
Sending confirmation message variables
Sending message to renderer

Figure 7.2: Processes of the Data Entry Subsystem

Figure 7.2 provides additional details on the processes. It shows that process *Select pump settings or infusion parameters* has the following subprocesses:

1. Displaying the information
2. Sending data to render
3. Getting selections
4. Sending variables

The name of these subprocesses have been reduced in Figure 7.2 but the whole names are written in Table 7.2. They read as follow: Input driver gets selections, UI logic sends variables that have being modified to the status manager, status manager sends data to renderer, renderer displays information. The description of the subprocesses of *Editing settings* and *Confirmation and submission* are also described in Table 7.2.

An ontology is thus created that includes information about processes and the hazards from the Preliminary Hazard Analysis. The ontology will be used in the next section for the automated analysis of hazards paths.

Table 7.2

Process	Subprocesses	Subprocess and description
Data Entry Subsystem	Select pump settings or infusion parameters	1. Input driver gets selections 2. UI Logic sends variables that have being modified to the status manager 3. Status manager sends data to renderer 4. Renderer displays the information
	Editing settings	1. Input driver gets data being edited 2. UI Logic sends data to status manager 3. Status manager sends data to renderer 4. Renderer displays data
	Confirmation and Submission	1. UI Logic sends confirmation message variables 2. UI status manager sends the message to the renderer 3. Renderer display message 4. UI logic sends data to GIP Controller 5. UI logic sends data to the status manager 6. Status manager sends data to the renderer 7. Renderer displays data

Table 7.3: Hazards and Cause Hazards with relationship $isCauseHazardOf$

Cause Hazard	$isCauseHazardOf$ Hazard	Cause Hazard	$isCauseHazardOf$ Hazard
H-3	H-22	H-15	H10
H-4	H-1	H-16	H-3
	H-2	H-17	H-3
H-5	H-1		H-6
H-6	H-2	H-18	H-6
H-7	H-5	H19	H-18
H-8	H-5	H20	H-18
H-9	H-6	H-21	H-13
H-10	H-4		H-14
H-11	H-8		H-15
H-12	H-7		H-16
H-13	H-12		H-17
H-14	H-11	H-22	H-23

7.2.2 Step 2: Perform an automated search of hazard paths

Table 7.1 shows the hazard analysed and its causes. We can notice that some hazards are also causes of other hazards. These relationships are analysed in the remainder of this sub-section.

Stage 5: Generation of State-Based Hazard Models from Ontology Rules Table 7.3 shows all the possible relationships of the type $isCauseHazardOf$ that can be encountered from the hazard analysis for the Data Entry subsystem. The first and third column are hazards that have been found to be causes of other hazards, these other hazards are detailed in the second and fourth column respectively.

The states and transitions obtained from these dependencies constitute a model. This model is a transition system which is written in a model checking specification language. The whole MCMAS model can be found in Appendix A.

Stage 6: Exploration of the state-based hazard model The focus of the analysis here is on identifying all possible hazards paths that lead to the following subsystem hazards:

- H-1 Misprogramming of quantity of medicine
- H-2 Misprogramming of flow or rate of medicine
- H-23 Misprogramming of parameter settings

For example, what we want to ask is: *Is there any path to a state where the hazard* $h1$ *"Misprogramming of quantity of medicine" is part of the system state?*.

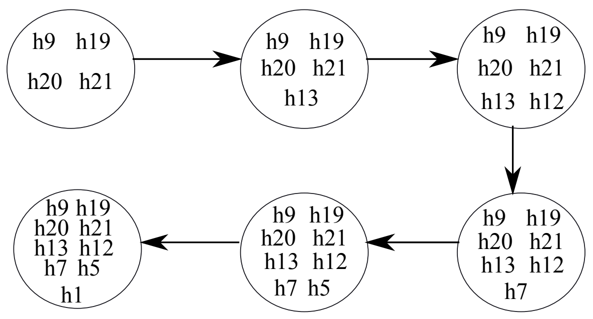


Figure 7.3: Data Entry Hazard Development

The following property, expressed in CTL Logic, has been used to find these kind of paths: $\mathsf{EF}h1$ (or $\exists\Diamond h1$). In MCMAS the atomic proposition $h1$ would be represented in MCMAS like `hazard1 if Environment.h1 = true` and then we vefiry the following property `EF hazard1`.

Stage 7: Visualizing Hazard Paths Figure 7.3 exemplifies a hazard path found from the hazard model represented. The circles represent system states and, in particular, the circle at the left bottom represent the state where hazard *H-1 Misprogramming of quantity of medicine* has been found and therefore the path finishes there. The arrows represent the transitions. The initial hazards state should include all the hazards found in our model but in Figure 7.3, for simplicity, we are only representing the hazards which are marked to be true during the initial state, which means, those are the ones occurring initially, then the hazard path will evolve from that.

7.3 Discussion

This case study focus on the data entry system of a generic infusion pump. Table 7.1 shows the hazard analysis done for this system. The data entry system of a generic infusion pump is a more complex case study than the medication prescription system. That is also why this hazard analysis is larger. Stage 4 of the method has not applied in this case study either. This case study mainly explores the second part of the proposed method. One of main ouputs of this case study would be an ontological representation of the infusion pump UI and the hazard analysis, another main output is the corresponding state based hazard model. The stated based hazard model has been proved useful because with the help of the MCMAS model checker it was possible to find a path to important system level hazard *H-1 Misprogramming of quantity of medicine.*

8 Voting System Prêt á Voter Case Study

Prêt á Voter [RBH+09] is a booth-based electronic voting system developed by Peter Ryan, David Chaum and Steve Schneider. The system is designed to guarantee *ballot privacy*, i.e., the voter's choice remains secret, and *auditability of the vote*, i.e., auditors can check that the election votes are counted correctly, and the voter can check that her vote has been counted [Lea11, PWB+].

8.1 System Overview

Prêt á Voter is a socio-technical system. It combines technical aspects such as saving a vote in encrypted form in a server, and human-related action such as a voter filling in a ballot form.

The system works as follows. A voter is issued a ballot form that contains all election candidates listed in a randomised order. The ballot form has two sides divided at the center by a clear divisory line along (see Figure 8.1). The left-hand side (LHS) of the form contains the names of the candidates. The right-hand side (RHS) contains white spaces the voter can use to mark her choice, as well as a representation of the candidate ordering in an encrypted form such that only those who tally the election can decrypt it. After marking a choice, the voter detaches the LHS and destroys it (Figure 8.2). This way, the order of the candidates can be accessed only by authorised election officers. The RHS remains. The polling station staff scans the RHS into the voting system to record the voter's choice. The voter keeps the RHS as a receipt of the performed vote. The vote is stored in the system using a key, called *cryptographic onion*. The key is encrypted with the public keys of several different authorised election officers, and will require that all these officers cooperate in order to decrypt the information.

The following sub-sections provide additional details on four core aspects of the system: Ballot Form Generation, Vote Casting, Audit of Ballot Forms, and Audit of Casted Votes.

8.1.1 Ballot Form Generation

The Prêt á Voter ballot form is secret because the list of candidates cannot be obtained from the RHS without a decryption key. However, before casting the vote, the two sides of the ballots form are still attached to each other, and the LHS side contains the candidate list in plain text.

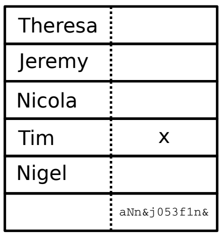


Figure 8.1: Ballot form of the Prêt á Voter System

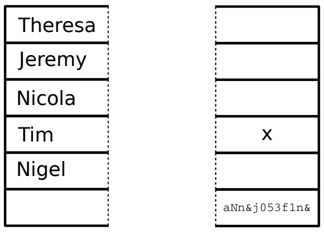


Figure 8.2: Detached sides of the Prêt á Voter Ballot form

Therefore, there should be a trusted custody of the ballot forms from its generation to its use in the polling station .

8.1.2 Vote Casting

The right-hand side of the ballot form contains a mark capturing the voter's choice and the order of the candidates in encrypted form. When scanning the RHS, what the scanning system needs to do is to extract the position of the vote and the encryption of the list ordering. This is all the system needs to record to correctly store and process the vote.

8.1.3 Audit of Ballot Forms

In a Prêt á Voter system, ballot forms can be checked by voters and vote auditors. For example, voters can check that the order of candidates encrypted on the right-hand side of a form corresponds to the list printed on the left-hand side. In order to do so, the voter selects a ballot form at random, then remove the left-hand side of the ballot form, and ask the system to decrypt the candidate list from the cryptographic onion on the right-hand side. The voter can then verify that actually the decrypted list matches the list of candidates printed on the left-hand side. This gives voters a way to double check the correct generation of the ballot forms. The tested ballot form needs of course to be discarded then, otherwise privacy of the vote would be lost.

8.1.4 Audit of Casted Votes

After scanning a vote in the system, the voter keeps the RHS as a receipt. It contains information necessary to check that the vote recorded in the system has not been altered. Votes that have been cast are in fact published on a Web Bulletin Board. Voters can audit their vote by comparing the information on theirs receipts and the information shown in the Web Bulletion Board. If their vote has not been recorded as a cast vote or if the information does not match their receipt by any means, the voter can challenge the election.

8.2 Modelling and Analysis of Prêt á Voter in HAZOP

The purpose of the Prêt á Voter HAZOP analysis is to identify potential problems in the different processes or subprocesses of the voting system. The analysis was performed by the following team: two experts of the HAZOP analysis method (one expert acted as chair of the team, the other as facilitator); and three experts in the area of cryptographic voting systems, who acted as experts in the design of the system. No single member of the team was a representative user, but each member had experience acting as a voter in a governmental election [Lea11].

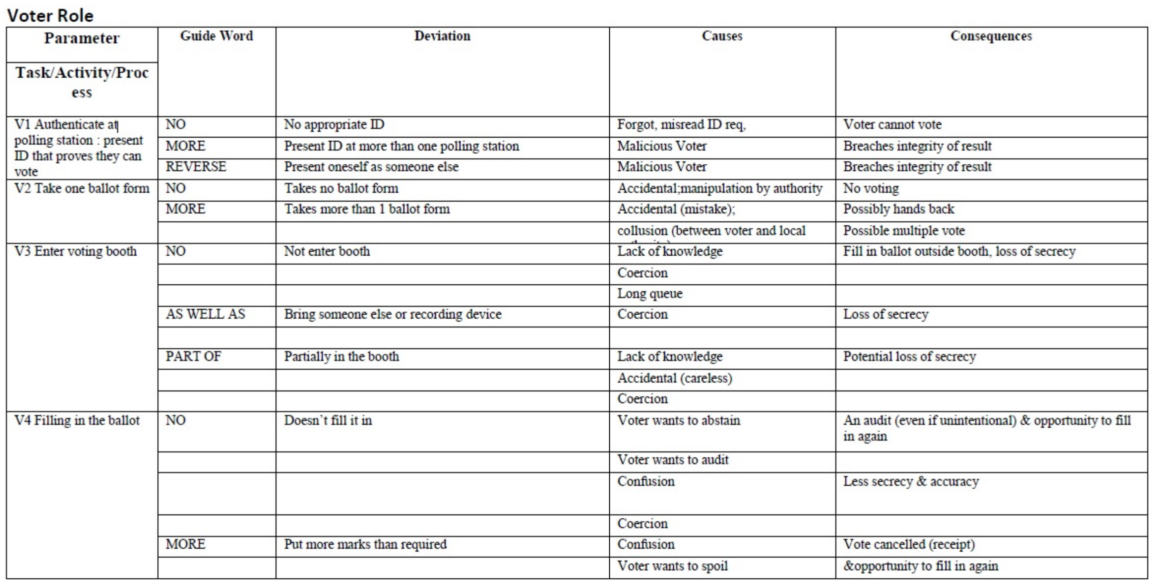
The Prêt á Voter HAZOP considered here is a refined version of an initial HAZOP analysis developed as part of the AROVE-V [PWB+] project. This original HAZOP had 136 deviations. The refined version counts additional 68 general deviations.

8.2.1 Modelling

In [PWB+], the Concurrent Task Tree [KRW12] (CTT) notation was used to create a model the system suitable for the HAZOP analysis. CTT is a graphical representations for specifying tasks and interactions between tasks. The notation is hierarchical, allowing larger tasks to be decomposed into smaller tasks. Specifically, CTT tasks were created to capture the activities carried out by different elements of the Prêt á Voter System. For example, tasks performed by a Voter include:

- Take one ballot form
- Destroy candidate list
- Enter voting booth
- Filling the ballot
- Remove candidate list

These tasks are listed in the HAZOP spreadsheet (see Figure 8.3). Each individual task determined during the creation of the CTT models became a task in the HAZOP analysis. The



Voter Role

Parameter / Task/Activity/Process	Guide Word	Deviation	Causes	Consequences
V1 Authenticate at polling station : present ID that proves they can vote	NO	No appropriate ID	Forgot, misread ID req,	Voter cannot vote
	MORE	Present ID at more than one polling station	Malicious Voter	Breaches integrity of result
	REVERSE	Present oneself as someone else	Malicious Voter	Breaches integrity of result
V2 Take one ballot form	NO	Takes no ballot form	Accidental;manipulation by authority	No voting
	MORE	Takes more than 1 ballot form	Accidental (mistake);	Possibly hands back
			collusion (between voter and local	Possible multiple vote
V3 Enter voting booth	NO	Not enter booth	Lack of knowledge	Fill in ballot outside booth, loss of secrecy
			Coercion	
			Long queue	
	AS WELL AS	Bring someone else or recording device	Coercion	Loss of secrecy
	PART OF	Partially in the booth	Lack of knowledge	Potential loss of secrecy
			Accidental (careless)	
			Coercion	
V4 Filling in the ballot	NO	Doesn't fill it in	Voter wants to abstain	An audit (even if unintentional) & opportunity to fill in again
			Voter wants to audit	
			Confusion	Less secrecy & accuracy
			Coercion	
	MORE	Put more marks than required	Confusion	Vote cancelled (receipt)
			Voter wants to spoil	&opportunity to fill in again

Figure 8.3: First page of the Prêt á Voter HAZOP Analysis

HAZOP has been performed over the roles of the participants in the Prêt á Voter system. The rest of this section presents the roles and a description of the tasks.

Voter. In an election, the Voter is the person who has the right to vote. In the Prêt á Voter system, the voter is the person casting a vote at the polling station and who is also able to verify if her vote has been properly recorded at the Web Bulletion Board (database). Before the day of the election, the voter identifies herself to be eligible. On the day of the vote, the voter goes to a polling station and authenticates. After successful authentication, the voter receives two ballot forms and enters in a voting booth. After marking the choice on the ballot form, the voter takes off the left part of ballot form and destroys it. The right part of ballot form is scanned by an official, and the voter keeps it as a receipt. When she receives her receipt, she should check that it was sign by the registration authority. When the vote is closed, this receipt can be used to check the bulletin board (which will be available on a specific website). During the election, a voter can challenge one ballot form. For example, she can ask the ballot generation authority to show how the ballots are constructed, and also check that the ballots can be correctly decrypted.

Voter Auditor. The role of the Vote Auditor is to audit the election via auditing a vote. During the voting phase, both voters and third independent parties can play the role of voter auditors [Lea11]. The Voter Auditor's task was described earlier on in this chapter, in Section 8.1.3.

Teller. The Teller is partially responsible for the ballot generation and for the vote processing task. The Teller also needs to proof ownership of private keys to the Web Bulleting Board. The Teller generates the cryptographic public/private keys used to encrypt the candidate lists. The public keys are published and they will be used by the Ballot Generation Authority to create ballot forms. The private keys, will be kept by the Teller in order to use it during the vote processing (decrypt the votes).

Teller Auditor. The Teller auditor is part of the authorities that are responsible for ensuring that certain parameters of the election are accomplished. During the Vote Processing, records should be auditable at each stage, at the same time, no link can be made from a set of decrypted votes from a Teller to the next one [Lea11]. The Teller Auditor is responsible for auditing decrypted sets of votes, at any stage, revealed by the Teller during the decryption phase.

Ballot Generation Authority. The Ballot generation authority is responsible for construction of ballot forms. The main task is to generate the ballot forms for the election. In order to do so, they need to receive, from the Teller, the different public keys. The Ballot Generation Authority creates the cryptographic onion and prints the ballot forms with the list of candidates in a random order. The Ballot Generation Authority is also responsible for destroying all data that relates the candidate list with the encrypted votes, in order to ensure secrecy of the voter's choice.

Polling Station Staff. The Polling Station Staff represents the local authorities during the voting phase [Lea11]. It includes: Scanner Attendants and Registration Desk Helpers.

Scanner Attendants are in charge of verifying that the LHS of the ballot form has been destroyed, confirming the ballot forms have beend properly filled, scanning filled ballot forms given by the voters (in the presence of the voters), signing the receipt and giving the RHS of the ballot form to the voter and a receipt (it is possible that the RHS is also the receipt) [Lea11].

The Registration Desk members are in charge of confirming voters are eligible to vote, issuing them a ballot form and registering that the voters have been issued a ballot form. The Registration Desk is also responsible for issuing a ballot form for the purpose of auditing the election(see Section 8.1.3).

Scanning System. The Scanning System is composition of hardware, software and people needed to scan the ballot forms and send the data to the Web Bulletin Board [Lea11]. It performs the following tasks: scan the ballot forms; transmit the encrypted data of the ballot forms to the Web Bulletin Board; print receipt of the process bein completed.

Web Bulletin Board. The Web Bulletin Board is the database where the recorded encrypted votes are stored. It is publicly viewable and append only [Lea11]. It is responsible for: storing all those records sent; ensuring that stored records are available for the voter's auditing process;

ensuring that stored records are available for decryption and tallying at the end of the election.

8.2.2 Results of the HAZOP analysis

The HAZOP analysis performed in [PWB+] identified 68 general deviations for the tasks carried out in the Prêt á Voter system. The HAZOP Guide Words were applied to the list of tasks derived from the CTT models. These results are captured on 11 pages of worksheets. A summary of the results are presented here. The full results are available in [Lea11].

- The role Voter has 14 activities (tasks or processes) analysed with 37 deviations.
- The role Voter Auditor has 6 activities (tasks or processes) analysed with 14 deviations.
- The role Teller has 7 activities (tasks or processes) analysed with 14 deviations.
- The role Teller Auditor has 3 activities (tasks or processes) analysed with 7 deviations.
- The role Ballot Generation Authority has 7 activities (tasks or processes) analysed with 11 deviations.
- The role Scanning Staff has 7 activities (tasks or processes) analysed with 9 deviations.
- The role Registration Staff has 6 activities (tasks or processes) analysed with 14 deviations.
- The role Scanning System has 6 activities (tasks or processes) analysed with 16 deviations.
- The role Web Bulletin Board has 5 activities (tasks or processes) analysed with 14 deviations.

8.3 Application of the Ontology-Based Method to the Prêt á Voter HAZOP Analysis

This section demonstrates how the ontology-based method presented in this thesis can be used to enhance the HAZOP analysis of the Prêt á Voter system. As discussed in Section 4.1, the ontology-based analysis method includes two steps and seven stages. It is not a hazard analysis but it is a method to provide a structure (or path) from which different initial hazard analysis can be evolved to a more refined versions. Also, it aims to provide scenarios about how the hazards evolve. This information can help the analysts gather a deeper understanding of how certain hazards could happen or originate. These results are presented further below, in Section 8.3.2.

In this particular case study, the second step of the ontology-based analysis method is not applied. Given that the causes of this HAZOP analysis were too general (*Coercion*, *Confusion*,*Accidental*), it was initially thought it was going to reduce the ocassions of finding *indirect causalities*.

8.3.1 Step 1: Carry out a systematic exploration of dependencies

This step has 4 stages. In this particular case study, the steps are instantiated as follows:

- Stage 1: Collect information about the Prêt á Voter system.
- Stage 2: Revise the HAZOP for Prêt á Voter with respect to all the information that has been collected.
- Stage 3: Develop two diagrams. One diagram describes describes the processes of the Prêt á Voter system. The second diagram describes the structure of the system. These diagrams are also represented in the ontology.
- Stage 4: Search implicit and overlooked relationships.

The following sub-sections describe these stages in more detail.

Stage 1: Hazard Analysis and System Description The main sources of information were used to build the Prêt á Voter Ontology representation were the system descriptions provided in the HAZOP analysis [Lea11] and in the AROVE-V project [PWB$^+$].

Stage 2: Systematic Revision The description of the Prêt á Voter system provided in [Lea11] is given in terms of tasks. It also makes a distinction between three distinct phases of operation: Pre-voting, Voting and Pos-voting. The description, however, does not explain which task is performed in which phase. Hence, this mapping is created here as part of the systematic revision of the description of the system.

Tables 8.11, 8.13, 8.17show the phases, with their respective steps, mapped against all the different tasks enumerated and analysed in the HAZOP study. By building and checking these tables, it was possible to discover some gaps in the description of the Voting phase. Specifically, the following tasks are not mentioned:

1. V13 Detect mismatch.
2. V14 Report a mismatch.
3. LS2 Verify left part destroyed.
4. LS3 Verify correct syntax of right part.
5. LR5 Check cancelled or audited receipts.
6. LR6 Provide replacement ballot.

In addition, it has also been discovered that the following task is not mentioned in the description of the Pos voting phase of Pret a Voter:

1. AT3 Report any irregularities flagged by computer system.

In spite of these tasks (processes) not being mentioned in the description of the system, given this description and the order it has been analyse in HAZOP, it is possible to understand the sequence with regards the whole description and steps of the system and therefore they could be represented in our ontology. This was possible, except for one case which is task (process) *LR6 Provide replacement ballot*.

In addition, the whole role *Voter Auditor* is not mentioned at all in the description of the system, and therefore, nor its associated tasks and it has not been represented in the ontology.

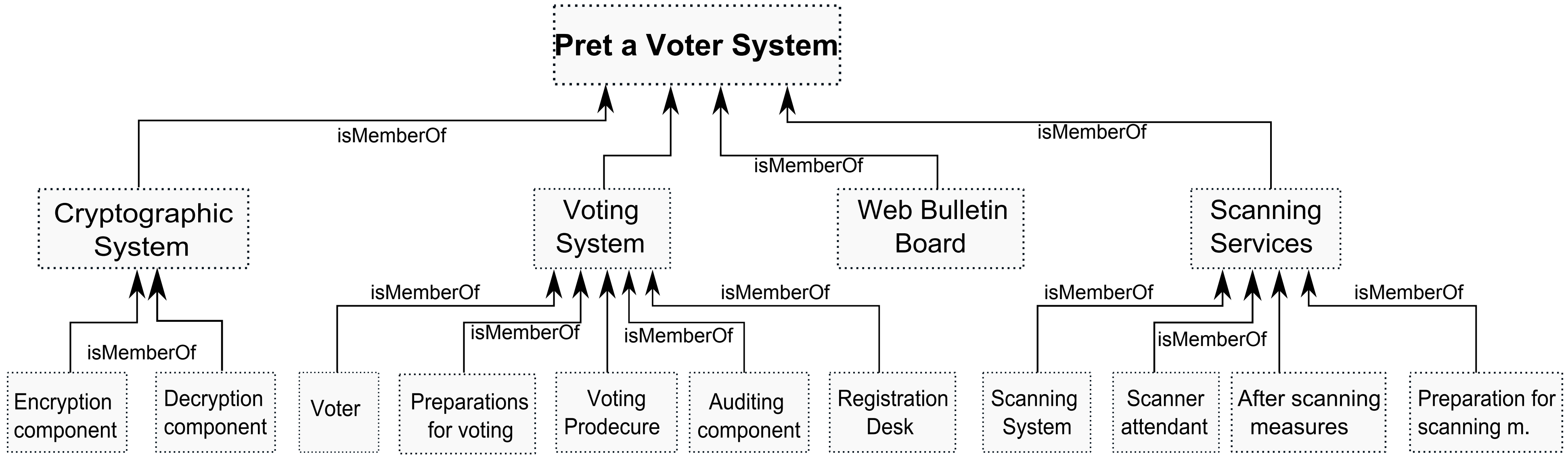


Figure 8.4: Pret A Voter Structure

Stage 3: Ontology-guided revision of inputs

Revising the inputs requires the development of a representation of the behaviour of the system (with processes) and a representation of the structure of the system.

Structural Representation Figure 8.4 shows the structural representation of the system. It includes the following subsystems: Cryptographic system, Voting System, Scanning Services, Web Bulletin Board. These subsystems were also identified as the main structures within the HAZOP analysis and the original project description of the Prêt á Voter system.

- The *Cryptographic System* is divided in two components (see Figure 8.4). Each component represents a part of the cryptographic architecture of Prêt á Voter: the *encryption component*, and the *decryption component*. Table 8.1 shows these two components are subsequently divided in the different structures that conform and outline them.
- The *Scanning Services* subsystem represents all the components that intervene in the scanning process. Starting from the protocols before and after the scanning of half of the ballot forms. It also represents the current scanning system and its submodules. Finally, it also represents the scanner attendant in charge of scanning and verifying that the protocols are followed before the scanning and after the scanning takes place. Table 8.2 represents the given description.
- The *Voting System* comprises the different actors who play a role in when a vote is being cast by a voter. The components representing protocols or procedures before casting a vote are *Preparation for voting*, *Voting Procedure* and *Auditing component*. The actors (components) within the voting system are *Voter*, *Voter Auditor* and *Registration Desk*.
- The *Web Bulleting Board* does not have component representation. It represents the database system that stores the votes and retrieves them to the Tellers and also retrieves queries about their votes to the voters.

The tasks performed by each actor are specified in tables 8.3– 8.10.

Behavioural Representation The behavioural representation of the overall system includes each task performed by each actor of the system. Additionally, some new processes were also added in order to follow more orderly and structurally the different steps in the Pre-Voting, Voting and Post-Voting phases.

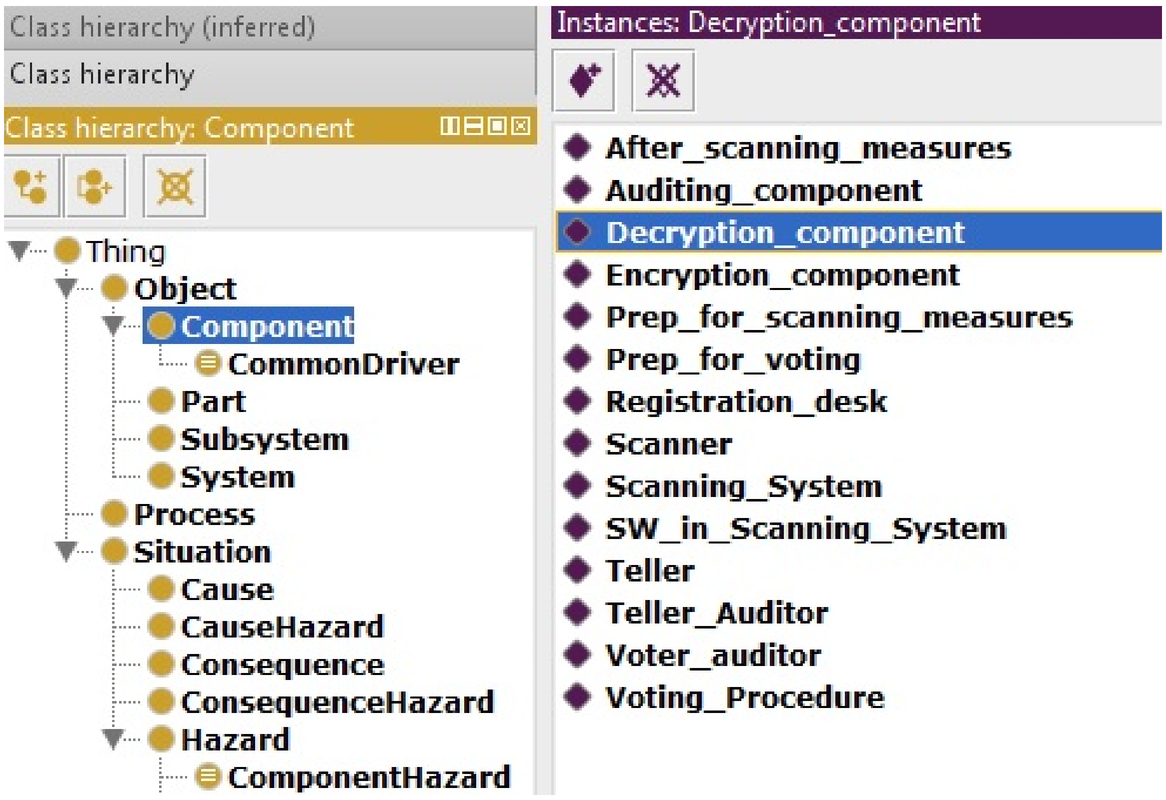
- *Pre-Voting Phase.* The actors involved in the Pre-voting phase are Teller and Ballot Generation Authority. The tasks associated to this phase are specified in Tables 8.11 and 8.12. The first table (Table 8.11) establishes a link between steps in the pre-voting phase and tasks carried out by actors. The second table (Table 8.12) further decomposes the tasks into a series of sub-tasks, following the CTT task specification provided in [Lea11].

- *Voting Phase.* The actors involved in the Voting phase are: Voter, Registration Staff, Scanning Staff, and Scanning System. The tasks associated to this phase are in Tables 8.13. Tables 8.14, 8.15, 8.16 further decompose the task into sub-tasks.
- *Post-Voting Phase.* The actors involved in the Voting phase are: the different Tellers (they decrypt the encrypted ballots and publish them on the Web Bulletin Board), and the Teller Auditors (they check the correctness of the voting system). The tasks performed by these actors in this phase are in Figure 8.16 and Table 8.18.

Stage 4: Search implicit and overlooked relationships

In this stage, a systematic exploration of dependencies between hazards (deviations) is performed. The analysis is carried out with the Protégé [SCfBIR15] tool. To this aim, the structural and behavioural representations of the Prêt á Voter system need to be modelled in Protégé. Figure 8.5 shows example structural representations I have created in Protégé. Figure 8.6 shows a sample of the processes represented in the ontology. Example hazards represented in the ontology are in Figure 8.8, and example causes of hazards are in Figure 8.7. It is important to mention that the relationships of every single hazard with respect to the structure and behaviour of the system is also represented. For example, Figure 8.9 shows the hazard ”Teller does not generate keys”, as well as the relationships related to that hazard that are part of the knowledge in the ontology. In this example, the relationship are concerned with the location of the hazard in the Prêt á Voter system, and information related to what aspects of the system will be affected by the hazard. Finally, Figure 8.10 shows the rules of inference that I have devised for searching implicit and overlooked dependencies.

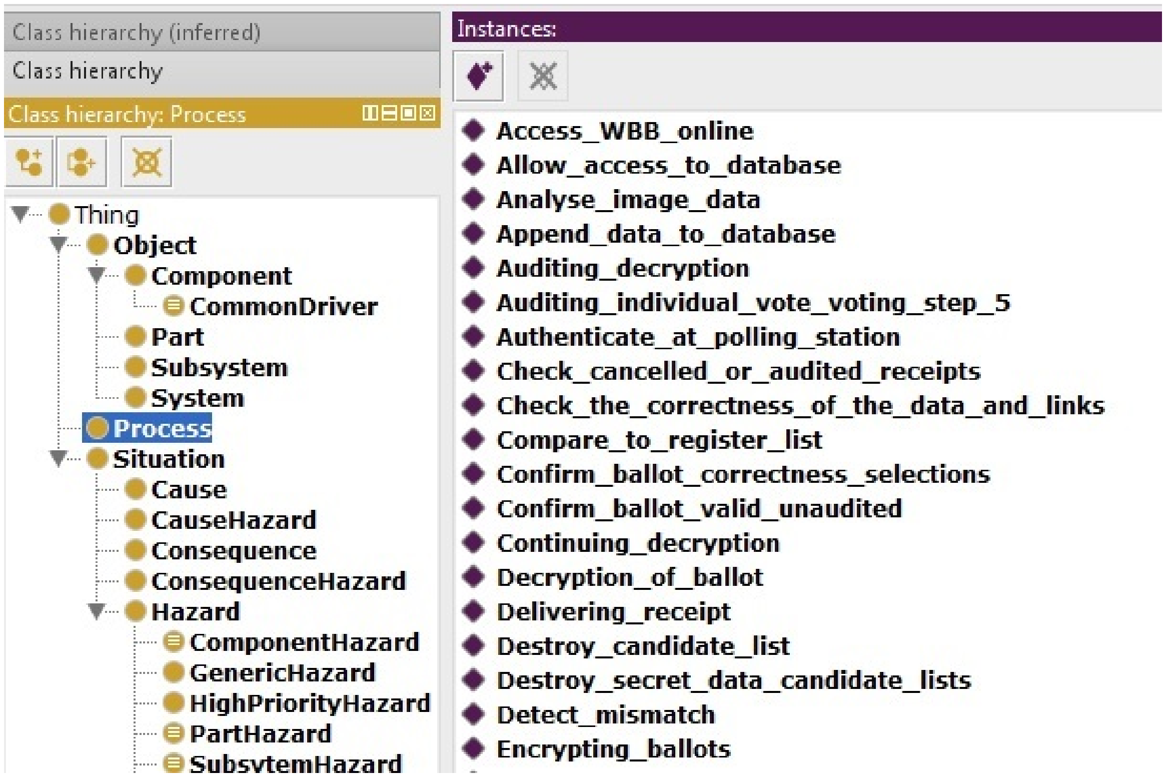
Once all the knowledge is represented in the ontology, the next step is search for implicit and overlooked relationships. Figure 8.11 shows a screenshot of the analysis performed using Protégé's reasoning engine and the inference rules. Figure 8.12 shows one of the results from our search. The main results obtained with the tool are presented in the following Section 8.3.2.


Class hierarchy (inferred)
Class hierarchy
Class hierarchy: Component
Instances: Decryption_component

```
Thing
    Object
        Component
            CommonDriver
        Part
        Subsystem
        System
    Process
    Situation
        Cause
        CauseHazard
        Consequence
        ConsequenceHazard
        Hazard
            ComponentHazard
```

- After_scanning_measures
- Auditing_component
- Decryption_component
- Encryption_component
- Prep_for_scanning_measures
- Prep_for_voting
- Registration_desk
- Scanner
- Scanning_System
- SW_in_Scanning_System
- Teller
- Teller_Auditor
- Voter_auditor
- Voting_Procedure

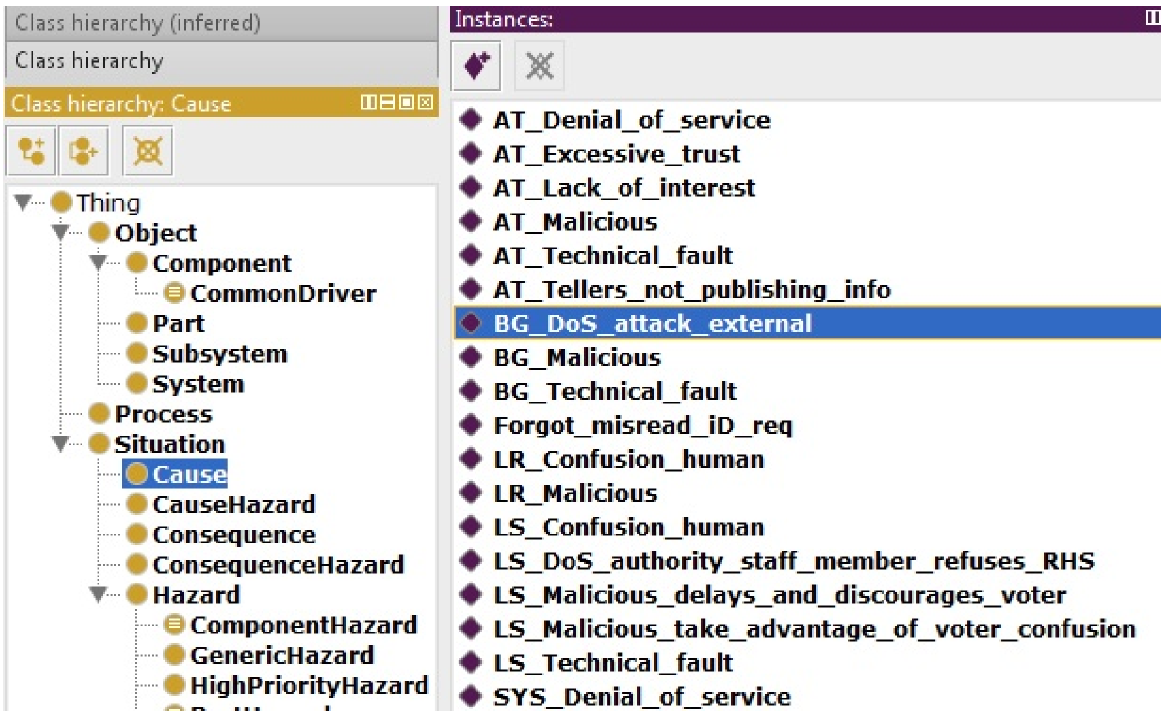
Figure 8.5: Protégé:Sample of components represented



```
Thing
    Object
        Component
            CommonDriver
        Part
        Subsystem
        System
    Process
    Situation
        Cause
        CauseHazard
        Consequence
        ConsequenceHazard
        Hazard
            ComponentHazard
            GenericHazard
            HighPriorityHazard
            PartHazard
            SubsytemHazard
```

- Access_WBB_online
- Allow_access_to_database
- Analyse_image_data
- Append_data_to_database
- Auditing_decryption
- Auditing_individual_vote_voting_step_5
- Authenticate_at_polling_station
- Check_cancelled_or_audited_receipts
- Check_the_correctness_of_the_data_and_links
- Compare_to_register_list
- Confirm_ballot_correctness_selections
- Confirm_ballot_valid_unaudited
- Continuing_decryption
- Decryption_of_ballot
- Delivering_receipt
- Destroy_candidate_list
- Destroy_secret_data_candidate_lists
- Detect_mismatch
- Encrypting_ballots

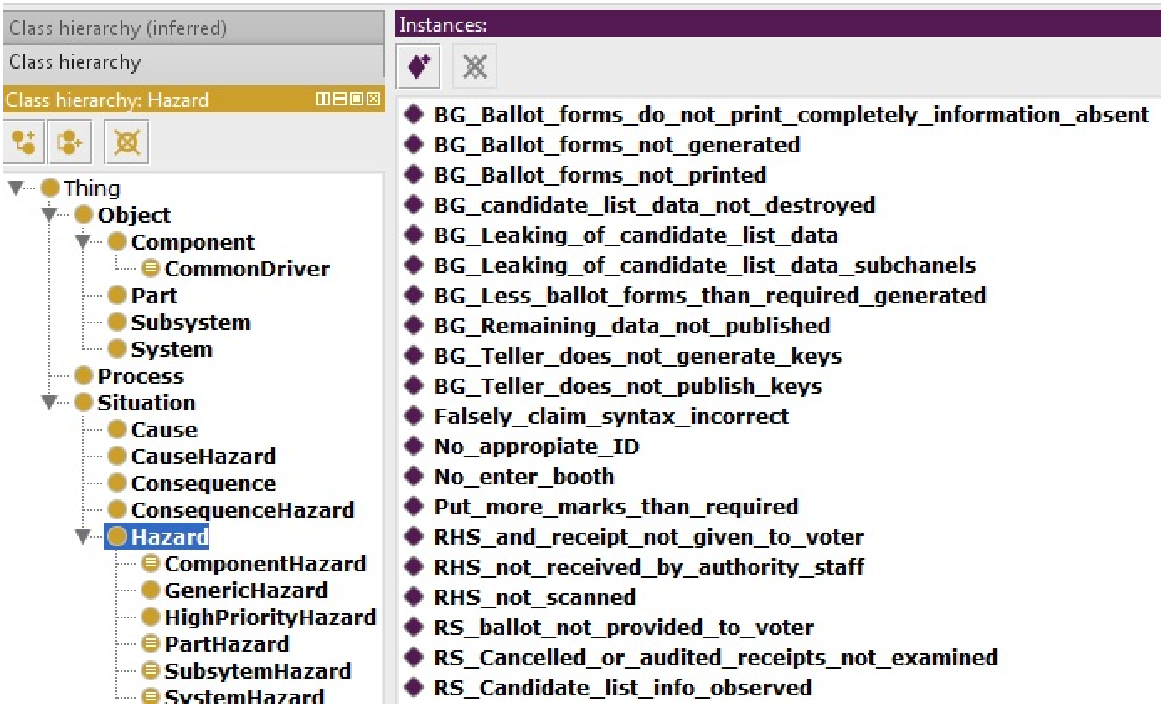
Figure 8.6: Protégé:Sample of processes represented



Thing
- Object
  - Component
    - CommonDriver
  - Part
  - Subsystem
  - System
- Process
- Situation
  - Cause
  - CauseHazard
  - Consequence
  - ConsequenceHazard
  - Hazard
    - ComponentHazard
    - GenericHazard
    - HighPriorityHazard

AT_Denial_of_service
AT_Excessive_trust
AT_Lack_of_interest
AT_Malicious
AT_Technical_fault
AT_Tellers_not_publishing_info
BG_DoS_attack_external
BG_Malicious
BG_Technical_fault
Forgot_misread_iD_req
LR_Confusion_human
LR_Malicious
LS_Confusion_human
LS_DoS_authority_staff_member_refuses_RHS
LS_Malicious_delays_and_discourages_voter
LS_Malicious_take_advantage_of_voter_confusion
LS_Technical_fault
SYS_Denial_of_service

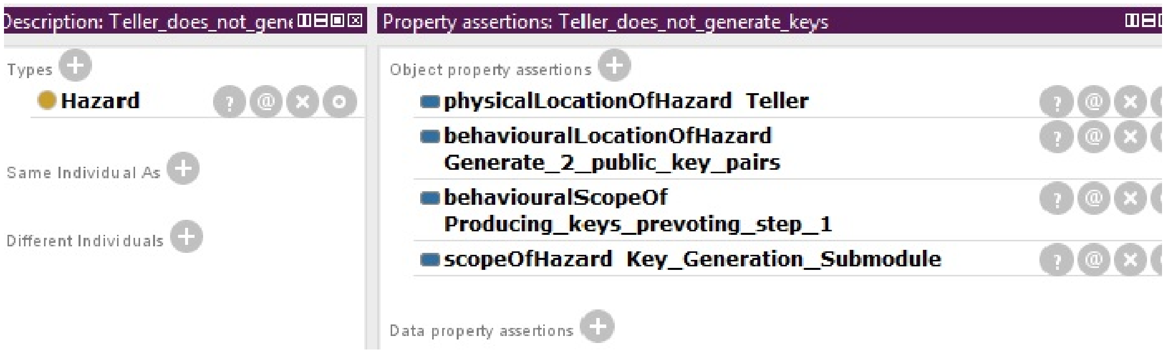
Figure 8.7: Protégé:Sample of causes represented



Thing
- Object
  - Component
    - CommonDriver
  - Part
  - Subsystem
  - System
- Process
- Situation
  - Cause
  - CauseHazard
  - Consequence
  - ConsequenceHazard
  - Hazard
    - ComponentHazard
    - GenericHazard
    - HighPriorityHazard
    - PartHazard
    - SubsytemHazard
    - SystemHazard

BG_Ballot_forms_do_not_print_completely_information_absent
BG_Ballot_forms_not_generated
BG_Ballot_forms_not_printed
BG_candidate_list_data_not_destroyed
BG_Leaking_of_candidate_list_data
BG_Leaking_of_candidate_list_data_subchanels
BG_Less_ballot_forms_than_required_generated
BG_Remaining_data_not_published
BG_Teller_does_not_generate_keys
BG_Teller_does_not_publish_keys
Falsely_claim_syntax_incorrect
No_appropiate_ID
No_enter_booth
Put_more_marks_than_required
RHS_and_receipt_not_given_to_voter
RHS_not_received_by_authority_staff
RHS_not_scanned
RS_ballot_not_provided_to_voter
RS_Cancelled_or_audited_receipts_not_examined
RS_Candidate_list_info_observed

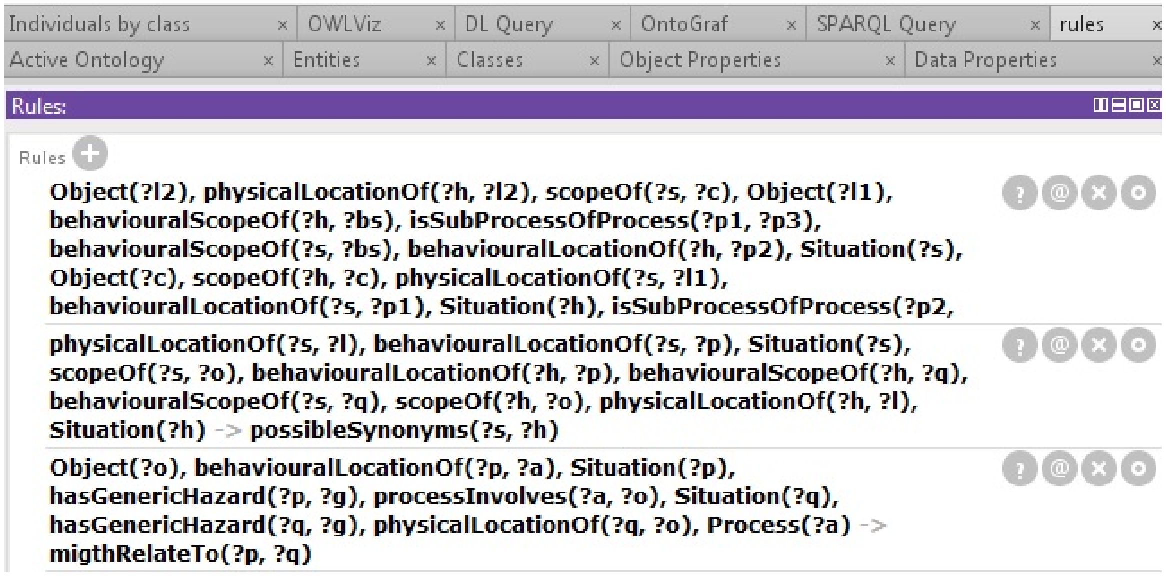
Figure 8.8: Protégé: Sample of hazards represented


Description: Teller_does_not_gen
Property assertions: Teller_does_not_generate_keys
Types
Same Individual As
Different Individuals
Object property assertions
Data property assertions

Hazard

physicalLocationOfHazard Teller
behaviouralLocationOfHazard Generate_2_public_key_pairs
behaviouralScopeOf Producing_keys_prevoting_step_1
scopeOfHazard Key_Generation_Submodule

Figure 8.9: Protégé:Sample of components represented



```
Object(?l2), physicalLocationOf(?h, ?l2), scopeOf(?s, ?c), Object(?l1),
behaviouralScopeOf(?h, ?bs), isSubProcessOfProcess(?p1, ?p3),
behaviouralScopeOf(?s, ?bs), behaviouralLocationOf(?h, ?p2), Situation(?s),
Object(?c), scopeOf(?h, ?c), physicalLocationOf(?s, ?l1),
behaviouralLocationOf(?s, ?p1), Situation(?h), isSubProcessOfProcess(?p2,

physicalLocationOf(?s, ?l), behaviouralLocationOf(?s, ?p), Situation(?s),
scopeOf(?s, ?o), behaviouralLocationOf(?h, ?p), behaviouralScopeOf(?h, ?q),
behaviouralScopeOf(?s, ?q), scopeOf(?h, ?o), physicalLocationOf(?h, ?l),
Situation(?h) -> possibleSynonyms(?s, ?h)

Object(?o), behaviouralLocationOf(?p, ?a), Situation(?p),
hasGenericHazard(?p, ?g), processInvolves(?a, ?o), Situation(?q),
hasGenericHazard(?q, ?g), physicalLocationOf(?q, ?o), Process(?a) ->
migthRelateTo(?p, ?q)
```

Figure 8.10: Protégé:Rules in the ontology

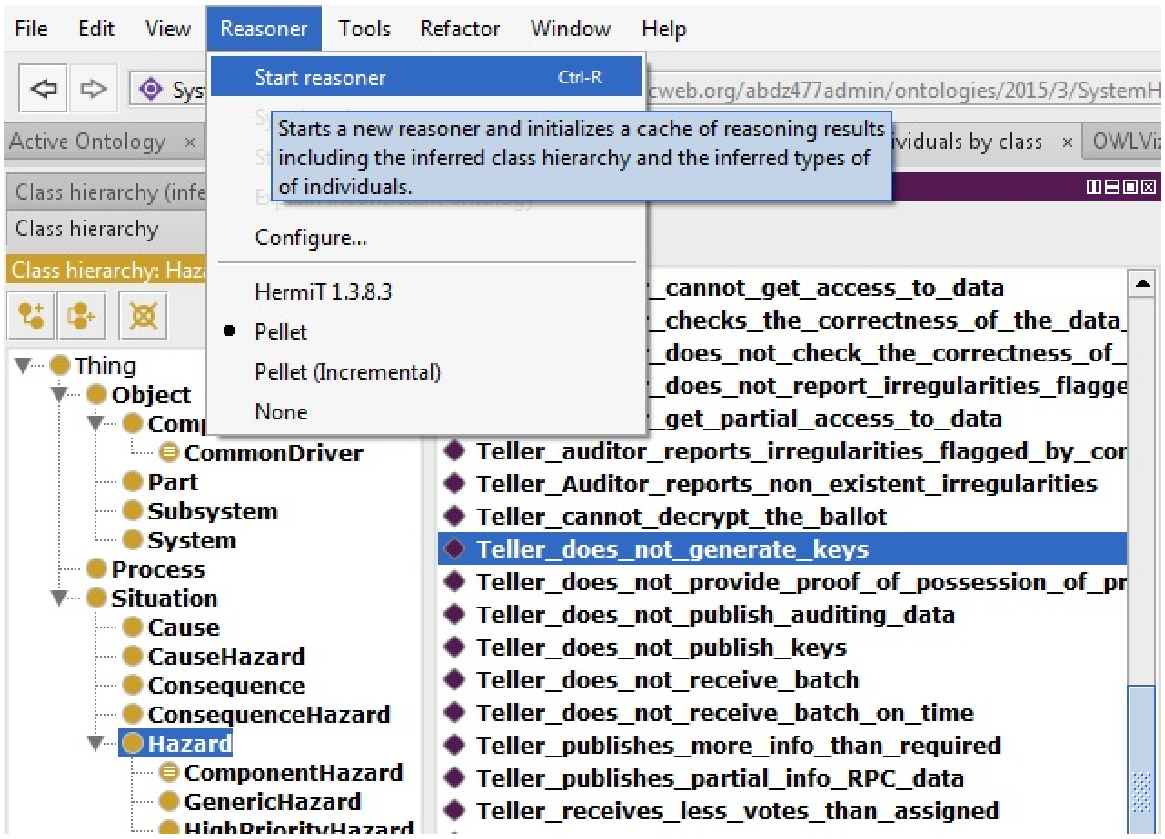
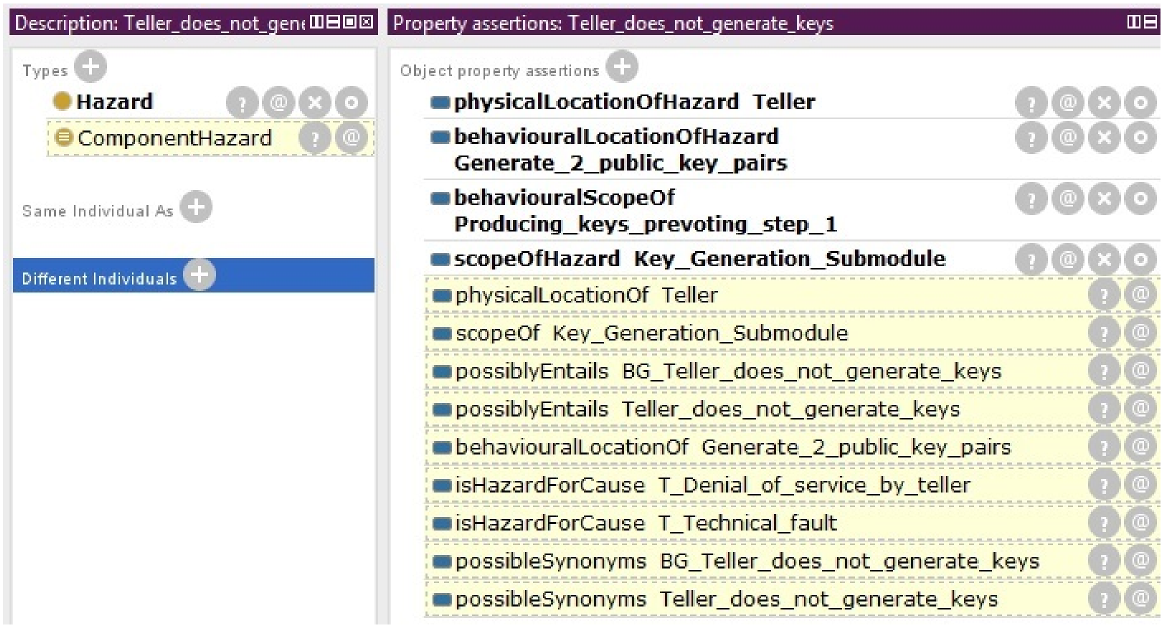

File
Edit
View
Reasoner
Tools
Refactor
Window
Help
Start reasoner Ctrl-R
Starts a new reasoner and initializes a cache of reasoning results including the inferred class hierarchy and the inferred types of of individuals.
Configure...
HermiT 1.3.8.3
● Pellet
Pellet (Incremental)
None
Active Ontology
Class hierarchy
Thing
Object
CommonDriver
Part
Subsystem
System
Process
Situation
Cause
CauseHazard
Consequence
ConsequenceHazard
Hazard
ComponentHazard
GenericHazard
Teller_auditor_reports_irregularities_flagged_by_cor
Teller_Auditor_reports_non_existent_irregularities
Teller_cannot_decrypt_the_ballot
Teller_does_not_generate_keys
Teller_does_not_provide_proof_of_possession_of_pr
Teller_does_not_publish_auditing_data
Teller_does_not_publish_keys
Teller_does_not_receive_batch
Teller_does_not_receive_batch_on_time
Teller_publishes_more_info_than_required
Teller_publishes_partial_info_RPC_data
Teller_receives_less_votes_than_assigned

Figure 8.11: Protégé:Starting the reasoner


Description: Teller_does_not_gen

Types
Hazard
ComponentHazard

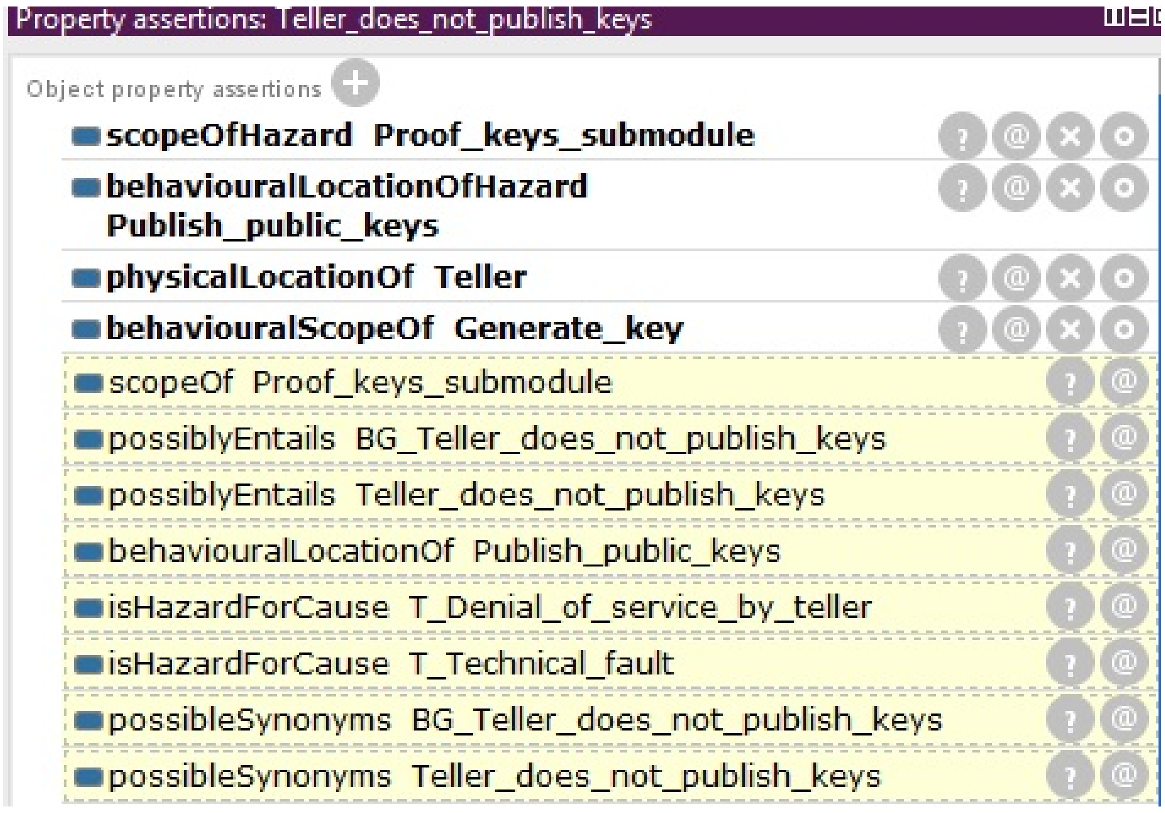
Same Individual As

Different Individuals

Property assertions: Teller_does_not_generate_keys

Object property assertions
physicalLocationOfHazard Teller
behaviouralLocationOfHazard Generate_2_public_key_pairs
behaviouralScopeOf Producing_keys_prevoting_step_1
scopeOfHazard Key_Generation_Submodule
physicalLocationOf Teller
scopeOf Key_Generation_Submodule
possiblyEntails BG_Teller_does_not_generate_keys
possiblyEntails Teller_does_not_generate_keys
behaviouralLocationOf Generate_2_public_key_pairs
isHazardForCause T_Denial_of_service_by_teller
isHazardForCause T_Technical_fault
possibleSynonyms BG_Teller_does_not_generate_keys
possibleSynonyms Teller_does_not_generate_keys

Figure 8.12: Protégé:An example of the results



Property assertions: Teller_does_not_publish_keys

Object property assertions

scopeOfHazard Proof_keys_submodule

behaviouralLocationOfHazard Publish_public_keys

physicalLocationOf Teller

behaviouralScopeOf Generate_key

scopeOf Proof_keys_submodule

possiblyEntails BG_Teller_does_not_publish_keys

possiblyEntails Teller_does_not_publish_keys

behaviouralLocationOf Publish_public_keys

isHazardForCause T_Denial_of_service_by_teller

isHazardForCause T_Technical_fault

possibleSynonyms BG_Teller_does_not_publish_keys

possibleSynonyms Teller_does_not_publish_keys

Figure 8.13: Process: Teller does not publish keys

Table 8.1: Structural representation of Cryptographic system

Subsystem	Component	Part	Part
Cryptographic system	Encryption component	Encrypting module	Key generation submodule
			Key generation measures
			Proof submodule (includes publication)
			Create ballots submodule
		Print ballots module	
		Destroy links measures	
		Publication module	
	Decryption component	Interface module	Receive submodule
			Send submodule
			Publish RPC submodule
		Decrypting module	
		Mix module	RPC submodule
		Tallying module	
		Verification decryption measures	

Table 8.2: Structural representation of Scanning Services

Subsystem	Component	Part
Scanning Services	Preparation for scanning measures After scanning measures Scanner attendant	
	Scanning System	Interface module Analysis module Printing module Sending (communication) module

Table 8.3: Task (or processes) vs Phases

Task of Voter	Phases
V1 Authenticate at polling station : present ID that proves they can vote V2 Take one ballot form V3 Enter voting booth V4 Filling in the ballot V5 Remove candidate list (LH part of ballot) V6 Destroy candidate list (LH part of ballot) V7 Present RHS for scanning V8 Take receipt V9 Access WBB online V10 Enter Correct Reference V11 View Record V12 Look for mismatch between Receipt and Electronic Record V13 Detect Mismatch V14 Report a mismatch	Voting Phase

Table 8.4: Teller Tasks

Task of Teller	Phases
T1 Generate 2 public key pairs T2 Publish public keys T3 Provide Proof of corresponding secret keys	Pre-voting phase
T4 Receive batch of encrypted votes (protected receipts) T5 Send batch on T6 Decryption of ballot T7 Publication of RPC data	Post-voting phase

Table 8.5: Teller Auditor Tasks

Task of Teller Auditor	Phases
AT1 Obtain access to tally auditing data AT2 Check the correctness of the data and links (mixes) AT3 Report any irregularities flagged by computer system	Post-voting phase

Table 8.6: Ballot Generation Authority Tasks

Task of Ballot Generation Authority	Phases
BG1 Generate Key BG2 Publish the public key BG3 Prove possession of private key BG4 Generate ballot forms BG5 Print ballot forms BG6 Destroy secret data (candidate lists) BG7 Publish remaining data (onions, ballot form serial numbers, etc.)	Pre-voting phase

Table 8.7: Scanning Staff Tasks

Task of Scanning Staff	Phases
LS1 Receive RHS LS2 Verify left part destroyed LS3 Verify correct syntax of right part LS4 Scan the right part LS5 Give right part and receipt	Voting phase

Table 8.8: Registration Staff Tasks

Task of Registration Staff	Phases
LR1 Review voter's ID LR2 Compare to register list LR3 Provide ballot to voter LR4 Mark voter as having been issued a form on register LR5 Check cancelled or audited receipts LR6 Provide replacement ballot	Voting phase

Table 8.9: Scanning System Tasks

Task of Scanning System	Phases
SYS1 Receive Image data	Voting Phase
SYS2 Analyse image data	
SYS3 Confirm ballot valid, unaudited	
SYS4 Confirm ballot correctness (selections)	
SYS5 Output Receipt	
SYS6 Transmit Receipt data to WBB	

Table 8.10: Web Bulletin Board Tasks

Task of Web Bulletin Board	Phases
WBB1 Receive data for posting	Voting phase
WBB2 Append data to DB	
WBB3 Allow access to DB	
WBB4 Locate searched item	
WBB5 Return appropriate data	Post-voting phase

Table 8.11: Prevoting phases vs Task and Processes

Steps of Prevoting phase		
Step	Description	Task (or Processes)
1.1	Teller creates 2 sets of public/private keypairs	T1 Generate 2 public key pairs
1.2	Teller publishes public key along with proof of ownership of private key to Web Bulletin Board	T2 Publish public keys T3 Provide Proof of corresponding secret keys
11.3	Ballot Generation Authority creates public/private keypairs (for signature purposes)	BG1 Generate Key
1.4	Ballot Generation Authority publishes public key along with proof of ownership of corresponding private key to Web Bulletin Board	BG2 Publish the public key BG3 Prove possession of private key
2	Ballot Generation Authority generates ballot form data, encrypting onions with Teller's public keys retrieved from Web Bulletin Board. This data includes:Random candidate ordering Cryptographic 'onion'	BG4 Generate ballot forms
3	Ballot Generation Authority prints ballot forms using the data generated in Step 2	BG5 Print ballot forms
4	Ballot Generation Authority destroy all data corresponding to the candidate ordering of ballot forms to ensure it cannot be used to reveal how individuals have voted	BG6 Destroy secret data (candidate lists)
5	Ballot Generation Authority publishes the remaining data in order to allow ballot forms to be confirmed as valid	BG7 Publish remaining data (onions, ballot form serial numbers, etc.)

Table 8.12: Prevoting phases vs Task and Processes

Steps of Prevoting phase				
Step	Description	Process	Subprocess	Subprocess
1.1	Teller creates 2 sets of public/private keypairs	Producing keys	T1 Generate 2 public key pairs	
1.2	Teller publishes public key along with proof of ownership of private key to Web Bulletin Board		Producing key Teller	T2 Publish public keys T3 Provide Proof of corresponding secret keys
1.3	Ballot Generation Authority creates public/private keypairs (for signature purposes)		BG1 Generate Key	
1.4	Ballot Generation Authority publishes public key along with proof of ownership of corresponding private key to Web Bulletin Board		Producing key BG	BG2 Publish the public key BG3 Prove possession of private key
2	Ballot Generation Authority generates ballot form data, encrypting onions with Teller's public keys retrieved from Web Bulletin Board. This data includes:Random candidate ordering Cryptographic 'onion'	BG4 Generate ballot forms		
3	Ballot Generation Authority prints ballot forms using the data generated in Step 2	BG5 Print ballot forms		
4	Ballot Generation Authority destroy all data corresponding to the candidate ordering of ballot forms to ensure it cannot be used to reveal how individuals have voted	BG6 Destroy secret data (candidate lists)		
5	Ballot Generation Authority publishes the remaining data in order to allow ballot forms to be confirmed as valid	BG7 Publish remaining data (onions, ballot form serial numbers, etc.)		

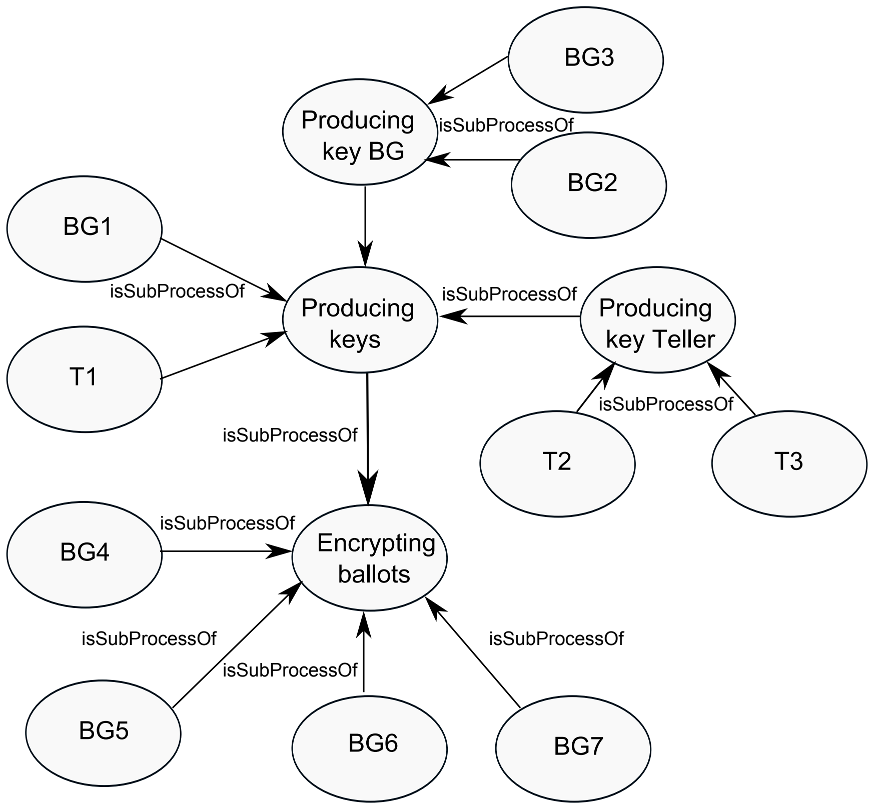


Figure 8.14: Pret A Voter Processes Prevoting

Table 8.13: Voting phases vs Task and Processes

Steps of Voting phase		
Step	Description	Task (or Processes)
1.1	The Voter presents their identification to the Polling Station Staff	V1 Authenticate at polling station: present ID that proves they can vote
1.2	Registration Staff confirm the Voter's identity, issue a ballot form and register the Voter as having been issued a ballot form	LR1 Review voter's ID LR2 Compare to register list LR3 Provide ballot to voter V2 Take one ballot form LR4 Mark voter as having been issued a form on register
2.1	The Voter enters the voting booth and fills in the ballot form	V3 Enter voting booth V4 Filling in the ballot
2.2	The Voter removes the candidate list from the ballot form and destroys it	V5 Remove candidate list (LH part of ballot) V6 Destroy candidate list (LH part of ballot)
3.1	The Voter hands the remaining section of the ballot form to the staff responsible for scanning	V7 Present RHS for scanning LS1 Receive RHS
3.2	The Scanning Staff enter the remaining section of the ballot form into the Scanning System	LS4 Scan the right part
3.3	The Scanning System scans the data on the ballot form and sends this to the Web Bulletin Board to be posted	SYS1 Receive Image data SYS2 Analyse image data SYS3 Confirm ballot valid, unaudited SYS4 Confirm ballot correctness (selections) SYS6 Transmit Receipt data to WBB WBB1 Receive data for posting WBB2 Append data to DB
4.1	The Scanning System prints out a paper receipt containing all the data which was printed upon the original ballot form	SYS5 Output Receipt
4.2	The Scanning Staff issue the paper receipt printed by the Scanning System to the Voter	LS5 Give right part and receipt V8 Take receipt
5.1	The Voter accesses the Web Bulletin Board via the Internet	V9 Access WBB online WBB3 Allow access to database
5.2	The Voter requests that their specific vote be shown by the Web Bulletin Board	V10 Enter Correct Reference
5.3	The Web Bulletin Board locates the Voter's record and displays it to them	WBB4 Locate searched WBB5 Return appropiate item
5.4	The Voter compares the paper receipt which they were issued at the polling station to the record displayed by the Web Bulletin Board to verify its correctness	V11 View Record V12 Look for mismatch between Receipt and Electronic Record

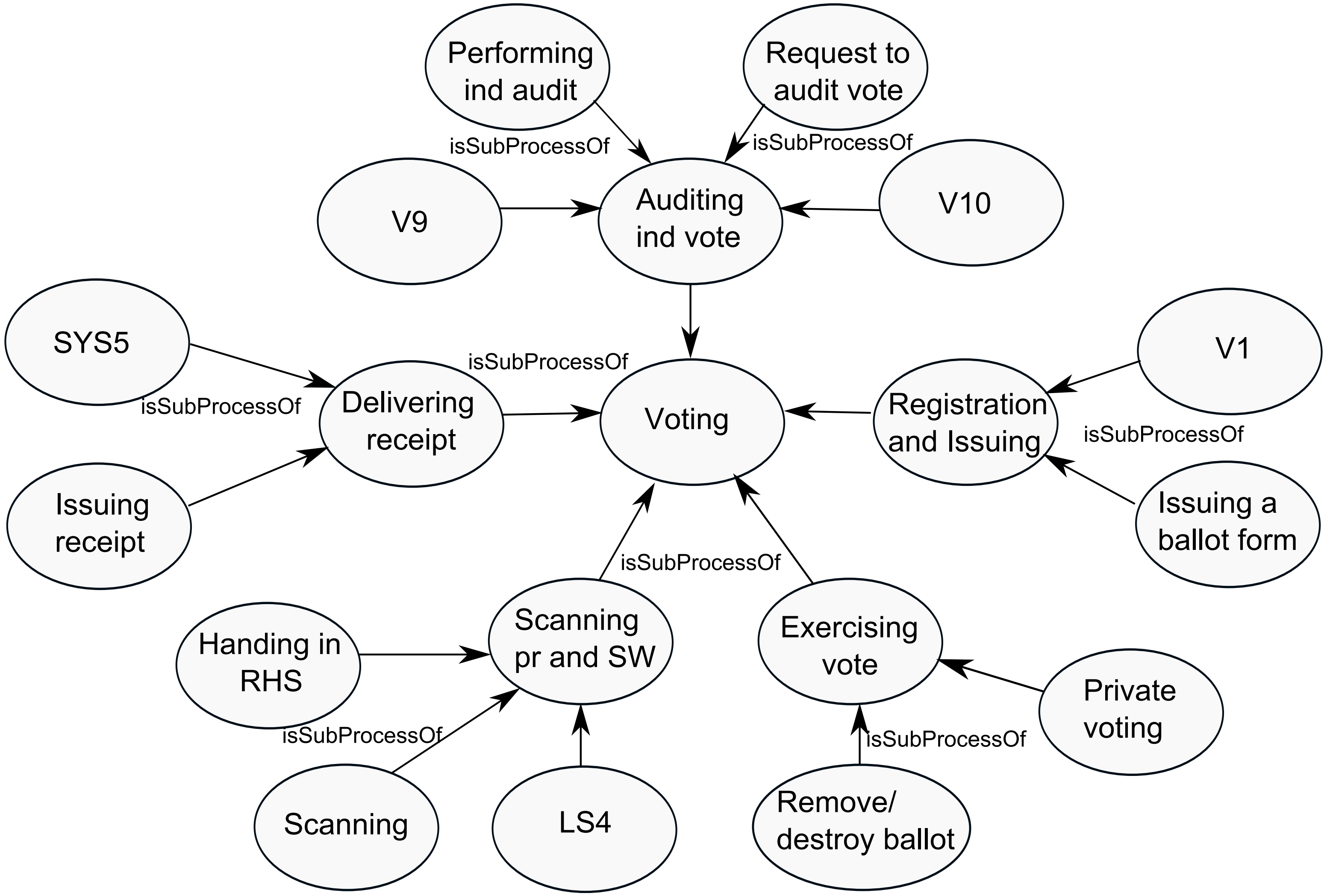


Figure 8.15: Pret A Voter Processes Voting

Table 8.14: Voting phases vs Task and Processes

Steps of Voting phase				
Step	Description	Process	Subprocess	Subprocess
1.1	The Voter presents their identification to the Polling Station Staff	Registration and issuing	V1 Authenticate at polling station: present ID that proves that they can vote	
1.2	Registration Staff confirm the Voter's identity, issue a ballot form and register the Voter as having been issued a ballot form		Issuing a ballot form	LR1 Review voter's ID LR2 Compare to register list LR3 Provide ballot to voter V2 Take one ballot form LR4 Mark voter as having been issued a form on register
2.1	The Voter enters the voting booth and fills in the ballot form	Exercising vote	Private voting	V3 Enter voting booth V4 Filling in the ballot
2.2	The Voter removes the candidate list from the ballot form and destroys it		Remove/destroy ballot	V5 Remove candidate list (LH part of ballot) V6 Destroy candidate list (LH part of ballot)

Table 8.15: Voting phases vs Task and Processes

Voting phase				
Step	Description	Process	Subprocess	Subprocess
3.1	The Voter hands the remaining section of the ballot form to the staff responsible for scanning	Scanning procedure and SW	Handling in RHS	V7 Present RHS for scanning LS1 Receive RHS
3.2	The Scanning Staff enter the remaining section of the ballot form into the Scanning System		LS4 Scan the right part	
3.3	The Scanning System scans the data on the ballot form and sends this to the Web Bulletin Board to be posted		Scanning	SYS1 Receive Image data SYS2 Analyse image data SYS3 Confirm ballot valid, unaudited SYS4 Confirm ballot correctness (selections) SYS6 Transmit Receipt data to WBB WBB1 Receive data for posting WBB2 Append data to DB
4.1	The Scanning System prints out a paper receipt containing all the data which was printed upon the original ballot form	Delivering receipt	SYS5 Output Receipt	
4.2	The Scanning Staff issue the paper receipt printed by the Scanning System to the Voter		Issuing receipt	LS5 Give right part and receipt V8 Take receipt

Table 8.16: Voting phases vs Task and Processes

Steps of Voting phase				
Step	Description	Process	Subprocess	Subprocess
5.1	The Voter accesses the Web Bulletin Board via the Internet	Auditing individual vote	V9 Access WBB online WBB3 Allow access to database	
5.2	The Voter requests that their specific vote be shown by the Web Bulletin Board		V10 Enter Correct Reference	
5.3	The Web Bulletin Board locates the Voter's record and displays it to them		Request to audit vote	WBB4 Locate searched item WBB5 Return appropiate item
5.4	The Voter compares the paper receipt which they were issued at the polling station to the record displayed by the Web Bulletin Board to verify its correctness		Performing individual audit	V11 View Record V12 Look for mismatch between Receipt and Electronic Record

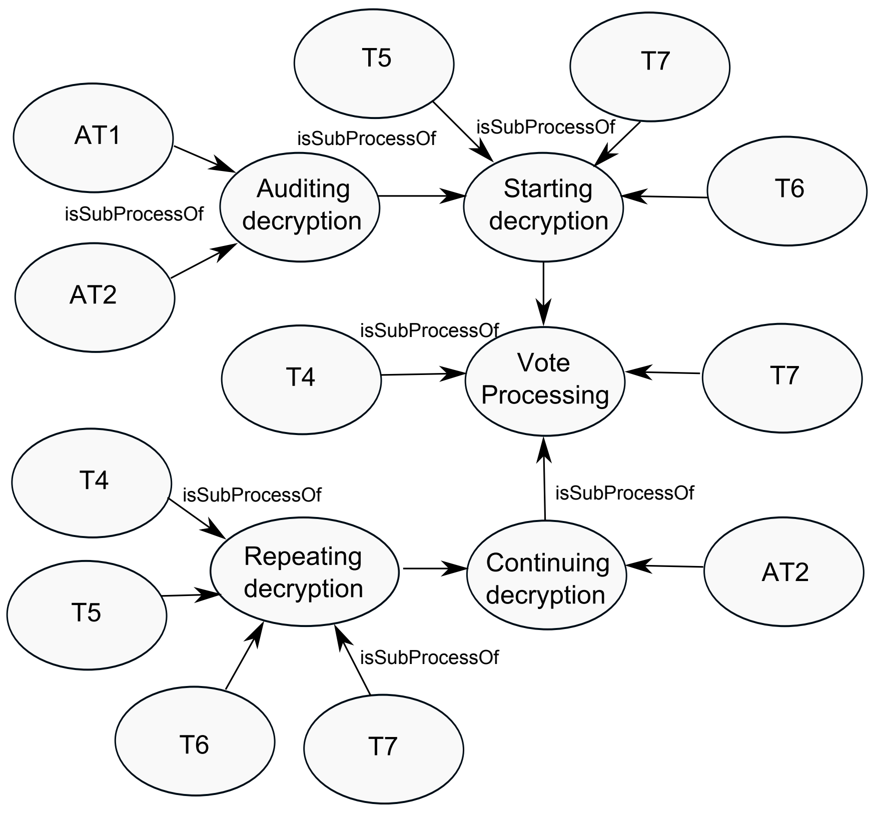


Figure 8.16: Pret A Voter Processes Posvoting

Table 8.17: Post-voting phase vs Task and Processes

Steps of Post-voting phase		
Step	Description	Task (or Processes)
1	First Teller retrieves encrypted voted records from Web Bulletin Board	T4 Receive batch of encrypted votes (protected receipts)
2.1	First Teller performs two sets of decryptions with private keys and performs the anonymising mix	T6 Decryption of ballot
2.2	First Teller publishes Random Partial Checking data to the WBB to allow the decryptions to be verified	T7 Publication of RPC data
2.3	First Teller sends the resulting decrypted records to the next Teller in line for decryption	T5 Send batch on
2.4	Teller Auditor uses Random Partial Checking data to verify that decryptions have been performed correctly	AT1 Obtain access to tally auditing data AT2 Check the correctness of the data and links (mixes)
3.1	Tellers following the first Teller perform receive, decrypt, mix and send on encrypted records and publish Random Partial Checking data in the same mane as the first Teller	T4 Receive batch of encrypted votes (protected receipts) T5 Send batch on T6 Decryption of ballot T7 Publication of RPC data
3.2	Teller Auditor uses Random Partial Checking data to verify that decryptions have been performed correctly	AT2 Check the correctness of the data and links (mixes)
4	The final Teller, having completed its decryptions and published the relevant Random Partial Checking Data, publishes the decrypted votes to the Web Bulletin Board allowing them to be tallied by any that wish to	T7 Publication of RPC data

Table 8.18: Post-voting phase vs Task and Processes

Steps of Post-voting phase				
Step	Description	Task (or Processes)	Subprocesses	Subprocesses
1	First Teller retrieves encrypted voted records from Web Bulletin Board	T4 Receive batch of encrypted votes (protected receipts)		
2.1	First Teller performs two sets of decryptions with private keys and performs the anonymising mix	Starting decryption	T6 Decryption of ballot	
2.2	First Teller publishes Random Partial Checking data to the WBB to allow the decryptions to be verified		T7 Publication of RPC data	
2.3	First Teller sends the resulting decrypted records to the next Teller in line for decryption		T5 Send batch on	
2.4	Teller Auditor uses Random Partial Checking data to verify that decryptions have been performed correctly		Auditing decryption	AT1 Obtain access to tally auditing data AT2 Check the correctness of the data and links (mixes)
3.1	Tellers following the first Teller perform receive, decrypt, mix and send on encrypted records and publish Random Partial Checking data in the same mane as the first Teller	Continuing decryption	Repeating decryption	T4 Receive batch of encrypted votes (protected receipts) T5 Send batch on T6 Decryption of ballot T7 Publication of RPC data
3.2	Teller Auditor uses Random Partial Checking data to verify that decryptions have been performed correctly		AT2 Check the correctness of the data and links (mixes)	
4	The final Teller, having completed its decryptions and published the relevant Random Partial Checking Data, publishes the decrypted votes to the Web Bulletin Board allowing them to be tallied by any that wish to	T7 Publication of RPC data		

8.3.2 Results

Table 8.19 reports the deviations (hazards) found as having an indirect relationship when applying our method. The table shows the deviations that are linked together, the subsystem where the deviation occur as well as the process and sub-process. The following dependencies between hazards emerged by analyzing this table.

1. **Dependencies between processes (1).** Consider the first two rows of Table 8.19, we find that the hazards (deviations) are described in exactly the same way. They have been found to have a relationship. They do occur in different processes and components. Figure 8.19 exemplifies the different connections that these two hazards have that lead us to suggest that they have a close relationship. In spite of belonging to different physical locations, here, the physical locations will be the roles where the hazards are being analysed. The tasks to which each hazard belong are written in slightly different ways: *Generate keys* and *Generate two public key pairs.* These could actually be the same process (or task). However, they have been described as different processes, and they are subprocesses of the same process *Producing keys.* These two hazards share the same behavioural scope which is *Producing keys* and they share the same physical scope *Key Generation Submodule.* Our inference mechanism can not tell that they are actually the same hazards repeated twice but suggests a close relationship between them. We believe the result gives the opportunity to the safety analyst to rename the hazards in order to offer more clarity, or correct the mistake, if any.

   Moreover, it can also occur, that during the process of populating our ontology, we could find that we are at the physical location *Ballot Generation Authority* but the description of the hazard (*Teller does not generate keys*) does not suggest an action done by the *Ballot Generation Authority* but rather an action performed by the *Teller*, the analyst could also decide to truly follow (or be guided by) the description of the hazard and relate the physical location of it to *Teller* rather than *Ballot Generation Authority*, the behavioural location would also need to changed. I have done this exercise, and the result will then suggest a rather synonym relationship between these two hazards, Figure 8.12 shows the results as given by the tool.

2. **Dependencies between processes (2).** A rather very similar case can be found in the third and fourth rows Table 8.19 we have two hazards (deviations) described exactly the same, *Teller does not publish key.* One of these hazards occur in the component *Teller* and the other in the component *Ballot Generation Authority*, also in different processes. The processes where these hazards belong are *Publish public keys* and *Publish the public key* respectively. The results show a close relationship (*possiblyEntails()*) between both. However, we could also follow same approach in previous example ,this is to be guided by the description of the hazards then we had to make these two hazards occur in the

component *Teller* and change behavioural location as well. The results as in the previous example would be a synonym relationship between both hazards. Figure 8.13 examplifies this.

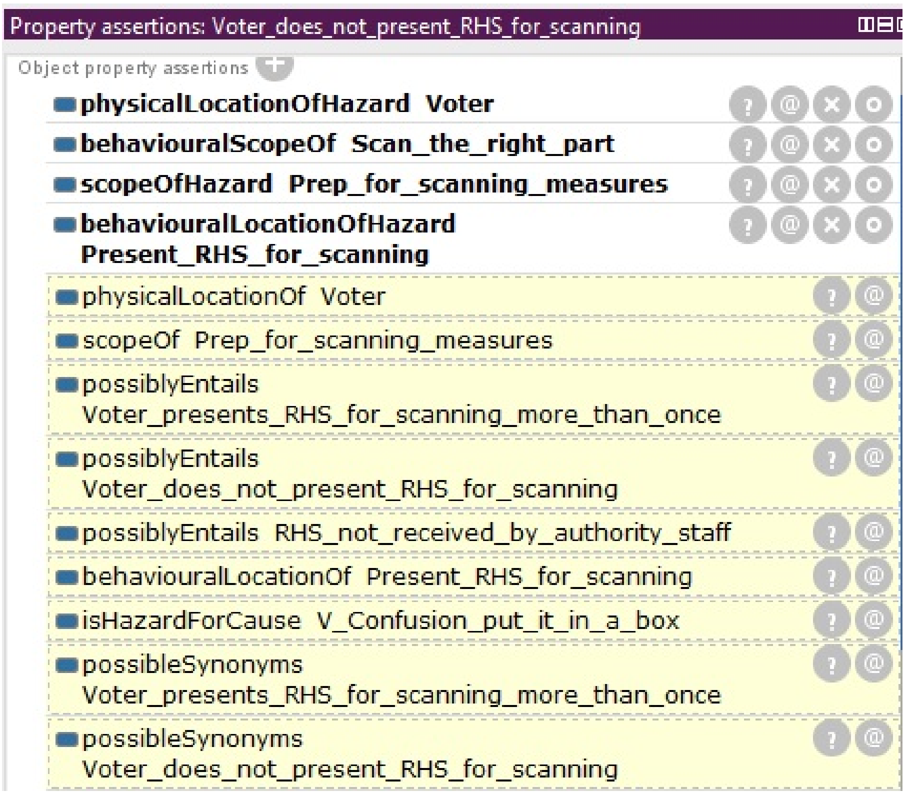
3. **Dependencies between error scenarios (1).** Consider the third, fourth and fifth rows of Table 8.19, it describes the following hazards: *Voter does not present RHS for scanning*, *Voter presents right hand side for scanning more than once* and *RHS not received by authority staff.* These group of three hazards are shown to have a possible entailment relationship. The first two hazards mentioned in this case, both occur within same component *Voter* and they both belong to same process *Present RHS for scanning*, this is because they both are deviations (in HAZOP) of same process but different guide words, these two hazards are also shown to have a synonym relationship between them. This is because being deviations (hazards) of same process in the HAZOP analysis, where the definitions are slightly different from each other it could occur that they might result in false positive cases.

   However, this group of three hazards still need further analysis (Figure 8.17 shows the result). Hazards *Voter does not present RHS for scanning*, and *RHS not received by authority staff*, they both have the same behavioural scope (process *Scan the right part*) but they have different original locations (*Voter* and *Registration Desk*, respectively). They both have the same structural scope *Preparation for scanning measures.* In my point of view, they both suggest an obvious relationship, as if one it is the cause of the other. This identified relationship gives the opportunity to the safety analyst to evaluate the refinement of both hazards or to make one cause of the other.

4. **Dependencies between error scenarios (2).** Consider the last three rows of Table 8.19, which captures the following hazards: *RHS and receipt not given to voter*, *Voter takes someone else's receipt* and *Voter does not take receipt.* Similarly to the previous result, hazards *Voter takes someone else's receipt* and *Voter does not take receipt* have also a synonym relationship between them and they are both deviations of same process but with different guide words. Figure 8.18 shows the result.

   However, hazards *RHS and receipt not given to voter* and *Voter does not take receipt* they both have the same behavioural scope: the process *Auditing individual vote* and they have different original locations one *Scanner attendant* and the latter *Voter.* They both have the same structural scope *After scanning measures.* In our point of view, they both suggest an obvious relationship, as if one is the cause of the other. It gives the opportunity to the safety analyst to improve the analysis.

5. **Clarifying case or case dismissed** Two hazards that have also been shown to have a close relationship (*possiblyEntails()*) are *More wrong voters marked as issued* and *Voter takes more than one ballot form.* The first hazards refers to the *Registration Desk* marking


Property assertions: Voter_does_not_present_RHS_for_scanning

Object property assertions

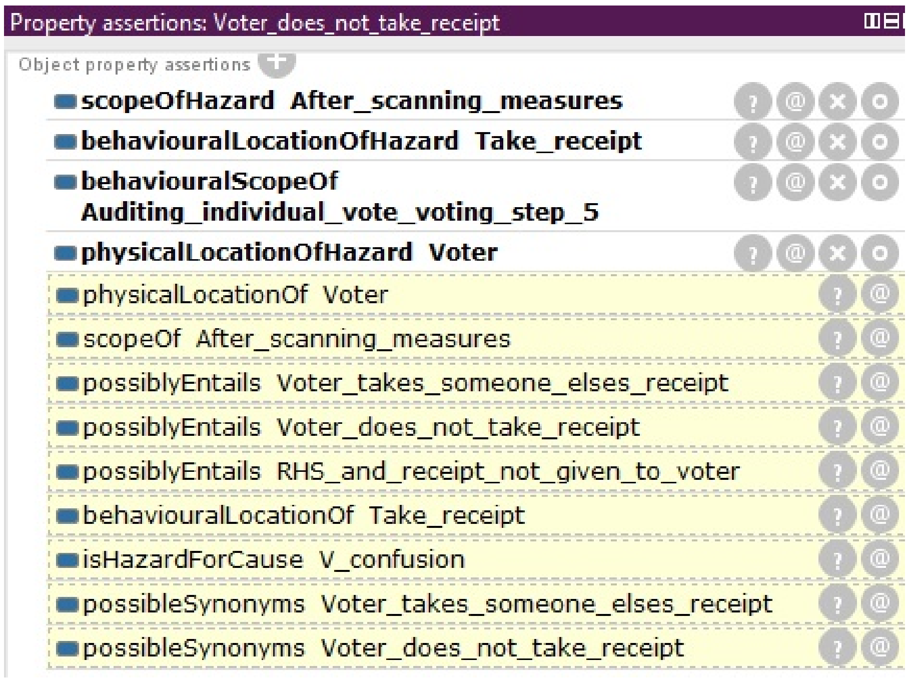
physicalLocationOfHazard Voter
behaviouralScopeOf Scan_the_right_part
scopeOfHazard Prep_for_scanning_measures
behaviouralLocationOfHazard Present_RHS_for_scanning
physicalLocationOf Voter
scopeOf Prep_for_scanning_measures
possiblyEntails Voter_presents_RHS_for_scanning_more_than_once
possiblyEntails Voter_does_not_present_RHS_for_scanning
possiblyEntails RHS_not_received_by_authority_staff
behaviouralLocationOf Present_RHS_for_scanning
isHazardForCause V_Confusion_put_it_in_a_box
possibleSynonyms Voter_presents_RHS_for_scanning_more_than_once
possibleSynonyms Voter_does_not_present_RHS_for_scanning

Figure 8.17: Hazard: Voter does not present RHS for scanning

more wrong voters as having being issued a ballot form, and the second refers to the *Voter* taking more than one ballot form. It is not clear what the analysis exactly might refer to with *More wrong voters marked as issued*, the process in this case is *Issuing a ballot form* and the structural location is *Registration Desk*. It could refer to the action of voters who did not receive a ballot form are marked as if they did. In this case there could be a relationship between these two hazards, in the sense that because a *Voter takes more than one ballot form* then *Registration Desk* decides to mark some other voter as issued a ballot form and as a consequence *More wrong voters marked as issued* a ballot form. There are many assumptions to make in this case, nevertheless, the analyst would have the opportunity to rename the hazards, to understand if they do hold a close relationship or if it is better to dismiss this result.

False Positives

The results have shown 21 cases of false positives results and all those case with a strong relation like synonymity. This can be explained because in all those cases the deviations (hazards) belong to the same process but with different guide word, this mean the meaning from one description to the other sometimes varies just a little and therefore the locations as well as the what would be affected would be the same processes or parts of the system. For example, consider the


Property assertions: Voter_does_not_take_receipt

Object property assertions

- scopeOfHazard After_scanning_measures
- behaviouralLocationOfHazard Take_receipt
- behaviouralScopeOf Auditing_individual_vote_voting_step_5
- physicalLocationOfHazard Voter
- physicalLocationOf Voter
- scopeOf After_scanning_measures
- possiblyEntails Voter_takes_someone_elses_receipt
- possiblyEntails Voter_does_not_take_receipt
- possiblyEntails RHS_and_receipt_not_given_to_voter
- behaviouralLocationOf Take_receipt
- isHazardForCause V_confusion
- possibleSynonyms Voter_takes_someone_elses_receipt
- possibleSynonyms Voter_does_not_take_receipt

Figure 8.18: Hazard: Voter does not take receipt

following synonymity case for component *Teller* with process *T4 Receive batch of encrypted votes* the following hazards have been found to have a possible synonymity relationship:

- *Teller does not receive batch* with guide word **No**
- *Teller deoes not receive batch on time* with guide word **Late**

This could be avoided writing a plug-in to improve the results of the reasoner specifically for this kind of cases that are expected to happen in a HAZOP analysis.

8.4 Discussion and Conclusions

The authors of the Prêt á Voter HAZOP have highlighted the rigorousness of the analysis. They have also argued that the level of completeness of the analysis was limited by the level of detail of design available to the analysis team. They considered that the completeness of the analysis would have been substantially increased if they were provided with information such as software and user interface design, details of procedure to be followed by staff in the polling station and details of the specific pieces of hardware and other equipment to be used during the election then. Nevertheless, the Prêt á Voter HAZOP is sufficiently extensive. Given the rigorousness of the analysis, there were not an overwhelming number of ambiguous hazards or repetitive hazards. This is not an unexpected result, even more, it gives significance to our results and to the uselfuness of our method because it can show that we can find redundances in refined analysis and while doing so add rigorousness to it.

Our method proved useful in refining such thorough analysis in the following ways:

1. It does not add more burden to the finalised analysis, because it requires only the information from the original hazard analysis;

2. It adds more rigor to the analysis, thanks to some level of systematic computer-guided iteration;

3. It helps to find redundant information in the hazards analysis worksheets;

4. It helps to find hazards that have been overlooked or seen as redundant.

The main limitation faced during the application of the ontology-based analysis was linked to missing details (e.g., in the original description of the system available in the public domain did not include the specification of all tasks). In addition, the causes described in the HAZOP analysis presented in [Lea11] were given using non-technical terms, such as *malicious*, *confusion*, *technical fault*. With these non-technical terms, the developed ontology is unable to capture significant relationship among hazards and causes.

Table 8.19: Deviations found to have similarities

Deviation	Process code	Process	Super Process	Phase
Teller does not generate keys	BG1	Generate Key	Producing keys	prevoting
Teller does not generate keys	T1	Generate 2 public key pairs	Producing keys	prevoting
Teller does not publish keys	T2	Publish public keys	Producing key Teller	prevoting
Teller does not publish keys	BG2	Publish the public key	Producing key BG	prevoting
Voter does not present RHS for scanning	V7	Present RHS for scanning	Handing in RHS	voting
Voter presents RHS for scanning more than once	V7	Present RHS for scanning	Handing in RHS	voting
RHS not received by authority staff	LS1	Receive RHS	Handing in RHS	voting
RHS and receipt not given to voter	LS5	Give right part and receipt	Issuing receipt	voting
Voter does not take receipt	V8	Take receipt	Issuing receipt	voting
Voter takes someone else's receipt	V8	Take receipt	Issuing receipt	voting

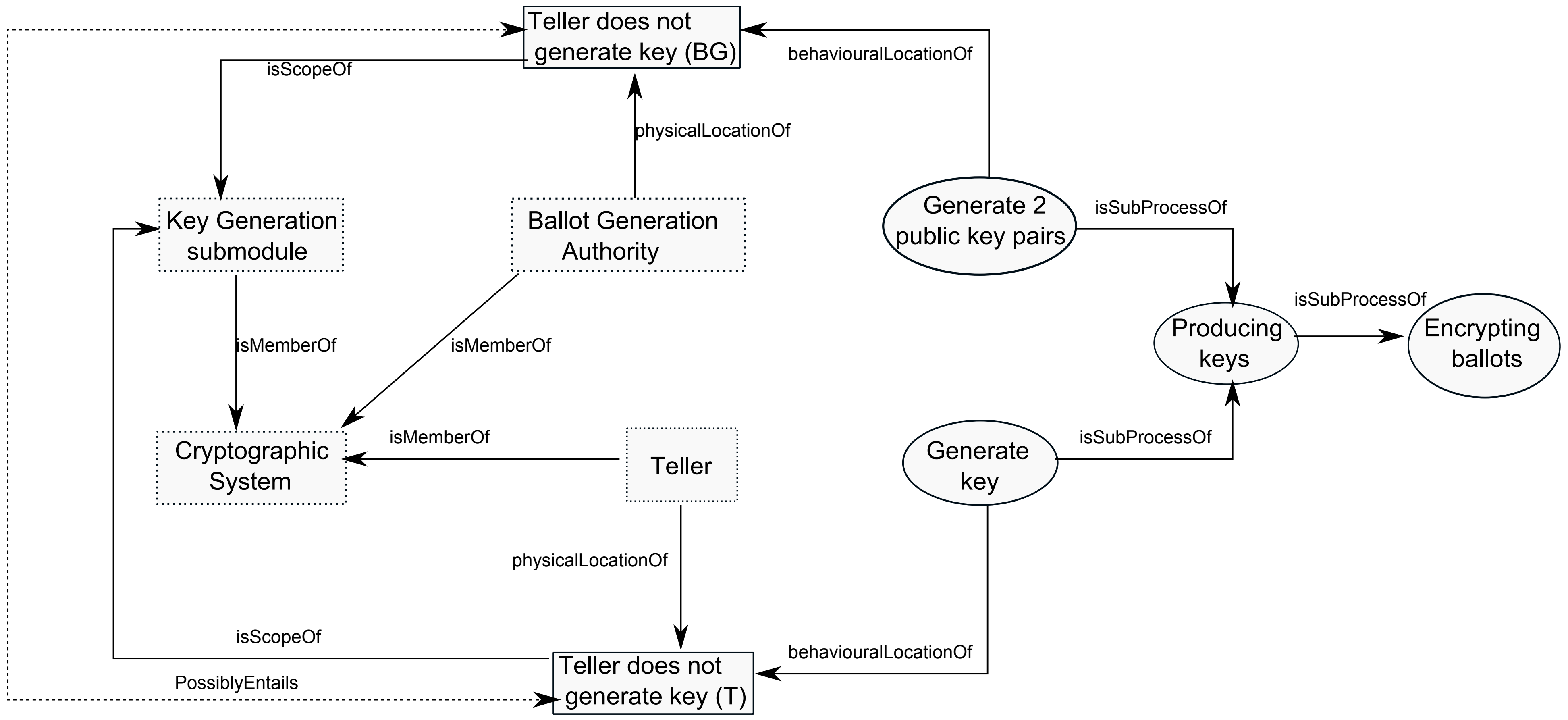


Figure 8.19: Example of the relationships of deviations *Teller does not generate keys*

9 The West Corridor Light Rail Transit (LRT) System

This case study is an industrial example based on the West Corridor LRT System. The West Rail Line is a 12.1-mile (19.47km) extension of the existing 35-mile (56.3km) Light Rail Transit (LRT) System in the city of Denver, Colorado, USA. RTD (Regional Transportation District) is the owner and operator of the project [rt11]. The West Rail Line is now officially operating as the West Line and it runs between Denver Union Station and Jefferson County Government Golden Station, adding 11 new stations, 6 Park-n-Rides, and 3 new Call-n-Rides [FRTDoDRD14].

9.1 System Overview

The West Rail Line includes the following subsystems [EA06]: Alignment, Track, Stations, Train control, and Overhead Catenary System (OCS).

Alignment. Alignment is the horizontal location of a railroad as described by curves and tangents. In our example, alignment includes all the components that are part of the rail line path such as tunnels and crossrails. The Alignment provides a path and physical infrastructure that will allow the circulation of the LRV (Light Rail Vehicle) over the track.

Track. Track is the structure consisting of the rails, fasteners, railroad ties (sleepers, British English) and ballast (or slab track), plus the underlying subgrade. A Track System provides the physical structure over which the LRV (Light Rail Vehicle) travels.

Stations. A station is a railway facility where trains regularly stop to load or unload passengers and/or freight. It generally consists of at least one track-side platform and a station building (depot) providing ancillary services such as ticket sales and waiting rooms. If a station is on a single-track line, it often has a passing loop to facilitate traffic movements. Stations may be at ground level, underground, or elevated. Connections may be available to intersecting rail lines or other transport modes such as buses, trams or other rapid transit systems.

Train Control. The train control system provides help in controlling the speed of the train,

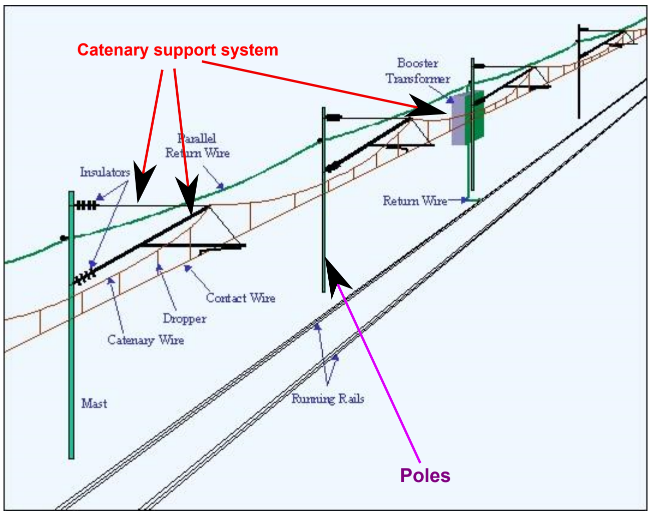


Figure 9.1: Schematic diagram of the catenary support system (modified from [www14c])

to prevent accidents and improve circulation. Specifically, the train control system is used for: train separation or collision avoidance; line speed enforcement; temporary speed restrictions; rail worker wayside safety.

Overhead Catenary System (OCS) The Overhead Catenary System (OCS) comprises different components, including wires, transformers and other parts that hold up the catenary support system that transmits energy to the LRV (see Figures 9.1). Wires are typically suspended between poles, and bridges are used to support overhead contact between the train and wires energized with electricity.

9.2 Preliminary Hazard Analysis of the LRT System

A preliminary hazard analysis was performed in 2006 as part of the West Corridor LRT Project Final Engineering Design Phase [EA06]. The analysis covers the main components of the system. A total of 44 hazards were identified among the different subsystems of the LRT System. These hazards were distributed in the following way, 12 hazards in Alignment, 6 hazards in Track, 12 hazards in Stations, 8 hazards in Train Control, 6 hazards in the Overhead Contact System.

Table 9.1: Preliminary Hazard Analysis

Item#	*Hazard Description*	*Potential Cause*	*Effect on Subsystem/System*
A-1	LRV Fire	A-2(LRV stopped adjacent to a wayside fire) Ca-1 Electrical short circuit igniting flammable materials Ca-2 Human action igniting flammable materials Ca-3 Ignition of flammable liquids on-board, beneath, or on top of vehicle	Injury, death, equipment damage, service disruption
A-2	Fire/smoke on alignment	STA-1 Fire at station Ca-4 Fire at wayside building or brush Ca-5 Motor vehicle fire at crossing Ca-6 Motor vehicle fire at crossing or adjacent to ROW Ca-7 Wood tie fire at special work Ca-8 Ignition of flammable materials being used for maintenance or stored near alignment	Injury to passengers, employees, or the public, equipment damage, service disruption
A-3	Fire/smoke in tunnel	A-1 A-2 Ca-9 Ignition of flamable materials in tunnel system or structure Ca-10 Ignition of flamable materials stored in tunnel Ca-11 Ignition of debris in tunnel	Injury, death, equipment or system damage, service disruption
STA-1	Fire/smoke on station platform	C-4 Electrical wiring fault Ca-12 Ignition of flammable gas/liquid Ca-13 Fire on adjacent property Ca-14 Vandalism Ca-15 Human error	Injury, death, equipment damage/loss, or service disruption
STA-4	Exposed electrical wiring or equipment	Ca-16 Improper design, installation, or maintenance Ca-17 Vandalism	Injury to patrons and employees due to electric shock

9.3 Application of the Ontology-Based Method to the West Corridor LRT Case Study

This section demonstrates how the ontology-based method presented in this thesis can be used to enhance the HAZOP analysis of the Prêt á Voter system. As discussed in Section 4.1, the ontology-based analysis method includes two steps and seven stages. It is not a hazard analysis but it is a method to provide a structure (or path) from which different initial hazard analysis can be evolved to a more refined versions. Also, it aims to provide scenarios about how the hazards evolve. This information can help the analysts gather a deeper understanding of how certain hazards could happen or originate. These results are presented further below, in Section 8.3.2.

9.3.1 Step 1: Carry out a systematic exploration of dependencies

This step has 4 stages. In this particular case study, the steps are instantiated as follows:

- Stage 1: Collect information about the West Corridor LRT system.
- Stage 2: Revise the hazard analysis with respect to all collected information.
- Stage 3: Develop two diagrams. One diagram describes the processes of the railway system. The second diagram describes the structure of the railway system. These diagrams are also represented in the ontology.
- Stage 4: Search implicit and overlooked relationships.

The following sub-sections describe these stages in more detail.

9.3.2 Stage 1: Hazard Analysis and System Description

The West Corridor LRT system is to transport passengers in a safe way through running the LRV West to East and backwards. The description of the system used here is based on that provided in [EA06]. The definitions of terms are from the reference glossaries [www14a, Tra14, www14c, www14d, www14b] in rail transport systems.

9.3.3 Stage 2: Systematic revision

The authors of the PHA for this case study [EA06] explicitly relate hazards A-1 *LRV fire* and A-2 *Fire/smoke on alignment* as a cause for the hazard A-3 *Fire/smoke in tunnel.* In addition they also explicitly identify hazard STA-1 as cause of hazard A-2, and hazard STA-12 as cause of A-4 (see Table 9.1).

Hazard *TC-5 Side and tip lights not illuminating or visible* was not represented in the ontology because it has exactly the same causes than hazard *TC-6 Grade warning system not visible or audible.* In addition, hazard TC-5 could be a cause of hazard TC-6 and therefore in this case it was chosen to represent hazard TC-6 only.

Some hazards (STA-5, STA-6, STA-7, STA-10) are not further considered in the analysis, as they are too general and the reasoning engine would not be able to provide further insights on them.

9.3.4 Stage 3: Ontology guided revision of inputs

Structure. Figure 9.2 shows the diagram representing the structure of the West Corridor LRT system. Table 9.2 show the components and parts of subsystem Track. The first column enumerates the components (*Permanent way*, *Rail fasteners*, *Track bed* and *Maintenace workers*). Note that component *Maintenace workers* is a *common driver* because it belongs to two subsystems at the same time. Table 9.3 shows the subsystem OCS. This subsystem includes the *catenary*, the *catenary supporting system*, the *pantograph* and the *surroundings of the catenary.* The subsystem *stations* includes: the *electrical wiring system*, the *platforms*, the *waiting rooms*, the *exiting paths* and the *surroundings of the stations.* The subsystem *Alignment*, whose structure is shown in Table 9.6,includes the following components: the *restricted alignment*, the *generic alignment*, the *tunnels*, the *maintenance alignment procedures*, the *maintenance workers* and the *grade crossing.* These last two components are *common drivers*, the component *Maintenance workers* also belongs to subsystem *Tracks* and the component *Grade crossing* also belongs to subsystem *Train control.*

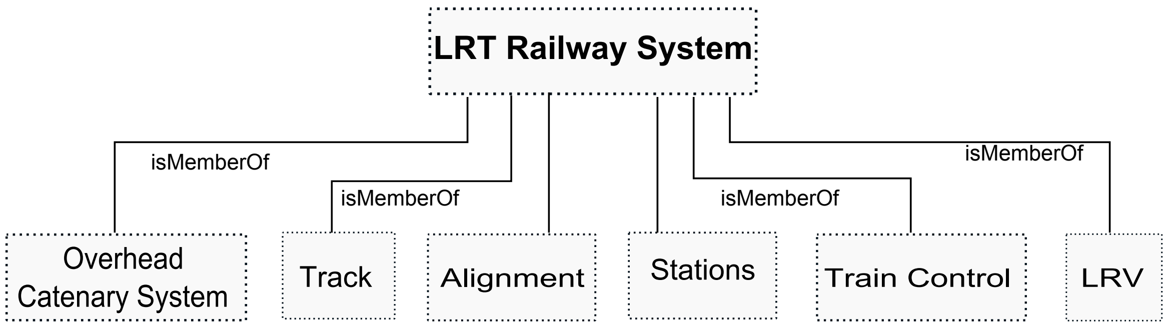


Figure 9.2: The West Corridor LRT System top structure

Processes. Figure 9.3 shows the main processes of the West Corridor LRT System. The main process is *Transporting.* The subprocesses of *Transporting* are: *Grade crossing management*, *Managing alignment*, *Managing stations*, *Circulating*, process *Managing stations*, *Transmission of power*, *Guide the LRV* and *Load and unload passengers.*

Table 9.7 shows the processes related to managing the alignment. The first column shows

Table 9.2: Subsystem Track

Components	Parts	Parts of parts
Permanent way	Rail	Expansion joints
		Switch
		Wheel-rail interface
	Sleepers	
Rail fasteners	Spikes	
	Fixing equipment	
	Rail anchor	
Track bed	Ballast	
	Subgrade	
Maintenance workers		

Table 9.3: Subsystem Overhead Contact System

Components	Parts
Catenary supporting system	Poles
Catenary	
Pantograph	
Surroundings catenary	

these main subprocesses, which are *Maintaining alignment*,*Protecting restricted alignment* and *Planning alignment*. The following columns are the respective subprocesses of these processes.

Table 9.11 shows the dependent subprocesses of process *Managing tracks*. The first column shows the immediate subprocesses of *Managing tracks* and the second column shows the respective subprocesses of these processes.

Table 9.10 shows all the subprocesses of process *Managing stations*. The first column shows the subprocesses *Maintaining stations*, *Providing facilities*, *Planning stations*. The second column shows the subprocesses of the previously mentioned processes.

Table 9.9 shows all the subprocesses of process *Grade crossing management*. The first column shows the immediate subprocesses of *Grade crossing management*, these processes are *Blocking crossing*, *Warning at crossing*, *Grade crossing maintenance*, *Giving way*, *Car stopping*, *LRV crossing*. The following columns are subprocesses of these processes.

Table 9.8 shows the subprocesses of process *Transmission of Power*. The first column shows processes *Protecting access to power areas*, *Allowing current to flow*, *Maintaining power equipment*, *Planning catenary*, *Supporting catenary*, *Feeding LRV*. Second column shows the respective subprocesses of the processes in the first column.

Table 9.4: Subsystem Stations

Components	Parts
Electrical wiring system	Cabinet Wires
Platforms	Platform gap Platform Waiting area
Waiting rooms	
Exiting paths	
Surroundings stations	

Table 9.5: Subsystem Train Control

Components	Parts	Parts of Part	Parts of Part
Grade Crossing	Grade crossing warning system	Light system	Circuitry
			Light bulbs
		Sign system	
	Grade crossing controller	Train controller (crossing)	
	Surroundings crossing		
	(Alignment) Crossing		
	Gate arm		

Table 9.6: Subsystem Alignment

Components	Parts
Restricted alignment	Surroundings at restricted alignment
	Restricted alignment at important venues
Maintenace workers	Maintenance workers procedures
Generic alignment	
Tunnel	
Maintenance alignment Procedures	
Grade Crossing	

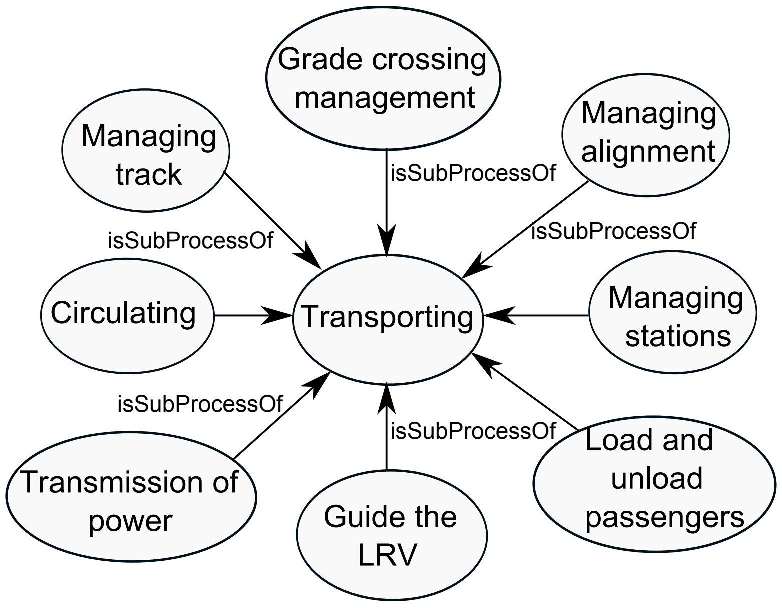


Figure 9.3: The West Corridor LRT System main processes

Table 9.7: Process: Managing alignment

Process	Subprocess	Subprocess
Maintaining alignment	Maintaining tunnel	Cleaning tunnel
		Unblocking tunnel
	Preventing flooding	Draining
		Providing defences
	Cleaning alignment	cleaning surroundings
	Unblocking alignment	
	Guiding work of personnel	
Protecting restricted alignment	Protecting surroundings	
	Using warnings and signs	
	Fencing alignment	
Urban planning alignment	Designing alignment	
	Constructing alignment	

Table 9.8: Process Transmission of Power

Process	Subprocess
Protecting access to power areas	Protecting from tresspasses
	Protecting from crashes
	Protecting surroundings
Allowing current to flow	Powering catenary
	Preventing disruption of power
Maintaining power equipment	Guiding electric personnel
Planning catenary	Designing catenary
	Constructing catenary
Supporting catenary	
Feeding LRV	

Table 9.9: Process: Grade Crossing Management

Process	Subprocess	Subprocess	Subprocess	Subprocess
Blocking crossing	Gate barrier operation	Gate control	Gate preview	Controlling train approach
		Mechanical electrical gate operation		
Warning at crossing	Warning with lights			
	Warning with sounds and signs			
	Coordinating signs			
	Coordinating traffic lights			
Grade crossing maintenance	Warnings maintenance			
	Gate maintenance			
	Unblocking crossing			
Giving way	Car approaching			
	Car restarting			
Car stopping				
LRV crossing				

Table 9.10: Process: Managing Stations

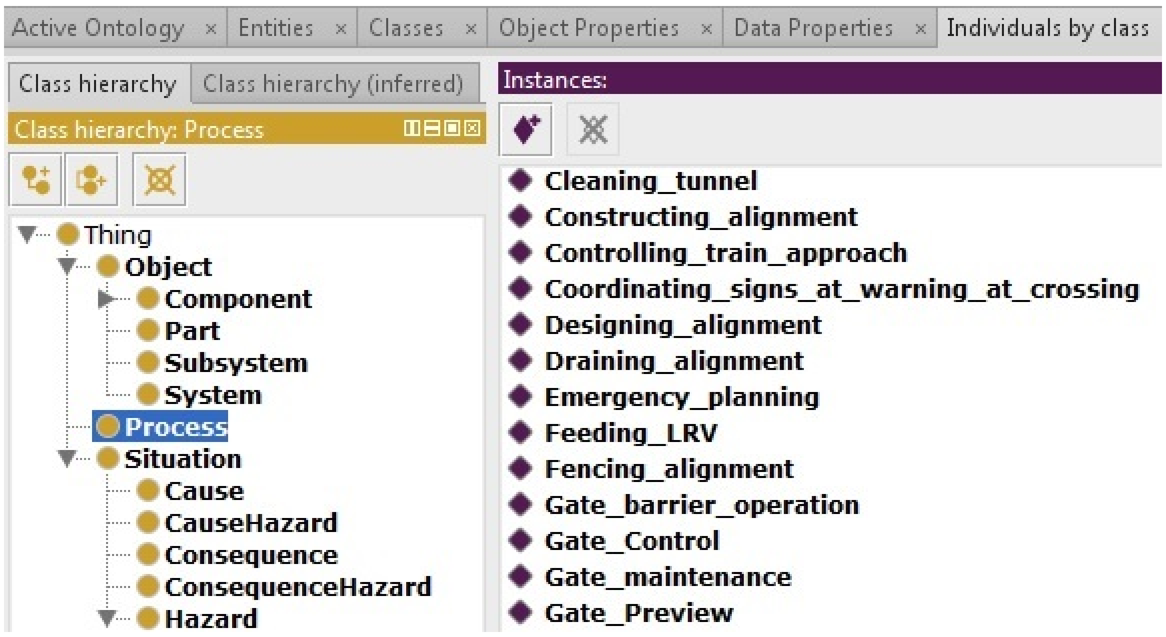
Process	Subprocess	Subprocess	Subprocess
Maintaining stations	Keeping surroundings		
	Emergency planning		
	Maintaining facilities	Unblocking platforms	
		Unblocking pathways	
Providing facilities	Providing spaces to wait for train		
	Providing electric power		
	Providing exiting paths		
Planning stations	Designing stations	Designing platform	Designing platform-LRV level
			Designing passenger space
		Designing waiting rooms	
		Designing exiting paths	
	Constructing stations	Constructing platform	
		Constructing waiting rooms	
		Constructing exiting paths	
Protecting stations	Protecting station from surroundings		
	Protecting premises from vandalism		
Controlling crowd	Controlling crowd in platform		
	Controlling crowd in exiting paths		
	Controlling crowd in waiting rooms		

Table 9.11: Process: Managing Tracks

Process	Subprocess
Guiding train	Proving surface
	Cushioning weight
Maintaning tracks	Anti-corrosion maintenance
	Replacing rails
	Replacing switches
	Cleaning tracks
Planning tracks	Designing tracks
	Constructing tracks

Figure 9.4 shows the subsystems of the LRT System represented in the ontology. Figure 9.5 shows a subset of the processes represented in the ontology, Figure 9.6 shows a subset of the hazard represented in the ontology.

Figure 9.4: Subsystems of LRT System represented in the ontology

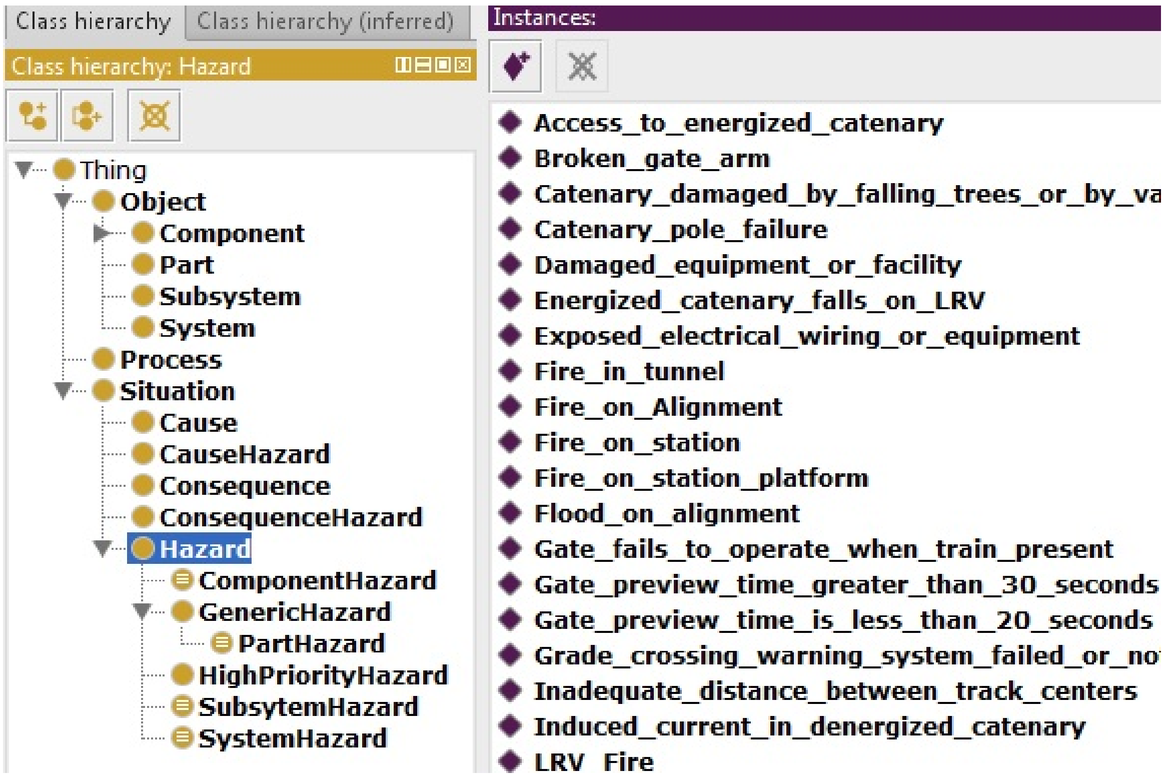


- Thing
  - Object
    - Component
    - Part
    - Subsystem
    - System
  - Process
  - Situation
    - Cause
    - CauseHazard
    - Consequence
    - ConsequenceHazard
    - Hazard

Instances:

- Cleaning_tunnel
- Constructing_alignment
- Controlling_train_approach
- Coordinating_signs_at_warning_at_crossing
- Designing_alignment
- Draining_alignment
- Emergency_planning
- Feeding_LRV
- Fencing_alignment
- Gate_barrier_operation
- Gate_Control
- Gate_maintenance
- Gate_Preview

Figure 9.5: Subset of Processes represented in the ontology



```
Thing
  Object
    Component
    Part
    Subsystem
    System
  Process
  Situation
    Cause
    CauseHazard
    Consequence
    ConsequenceHazard
    Hazard
      ComponentHazard
      GenericHazard
        PartHazard
      HighPriorityHazard
      SubsytemHazard
      SystemHazard
```

```
Access_to_energized_catenary
Broken_gate_arm
Catenary_damaged_by_falling_trees_or_by_va
Catenary_pole_failure
Damaged_equipment_or_facility
Energized_catenary_falls_on_LRV
Exposed_electrical_wiring_or_equipment
Fire_in_tunnel
Fire_on_Alignment
Fire_on_station
Fire_on_station_platform
Flood_on_alignment
Gate_fails_to_operate_when_train_present
Gate_preview_time_greater_than_30_seconds
Gate_preview_time_is_less_than_20_seconds
Grade_crossing_warning_system_failed_or_no
Inadequate_distance_between_track_centers
Induced_current_in_denergized_catenary
LRV Fire
```

Figure 9.6: Subset of hazards represented in the ontology

9.3.5 Stage 4: Search implicit and overlooked relationships

The Protégé tool is now used to find indirect relationships between hazards and causes. Figure 9.7 shows a screenshot of the analysis in Protégé. The main findings are discusses in the rest of this subsection.

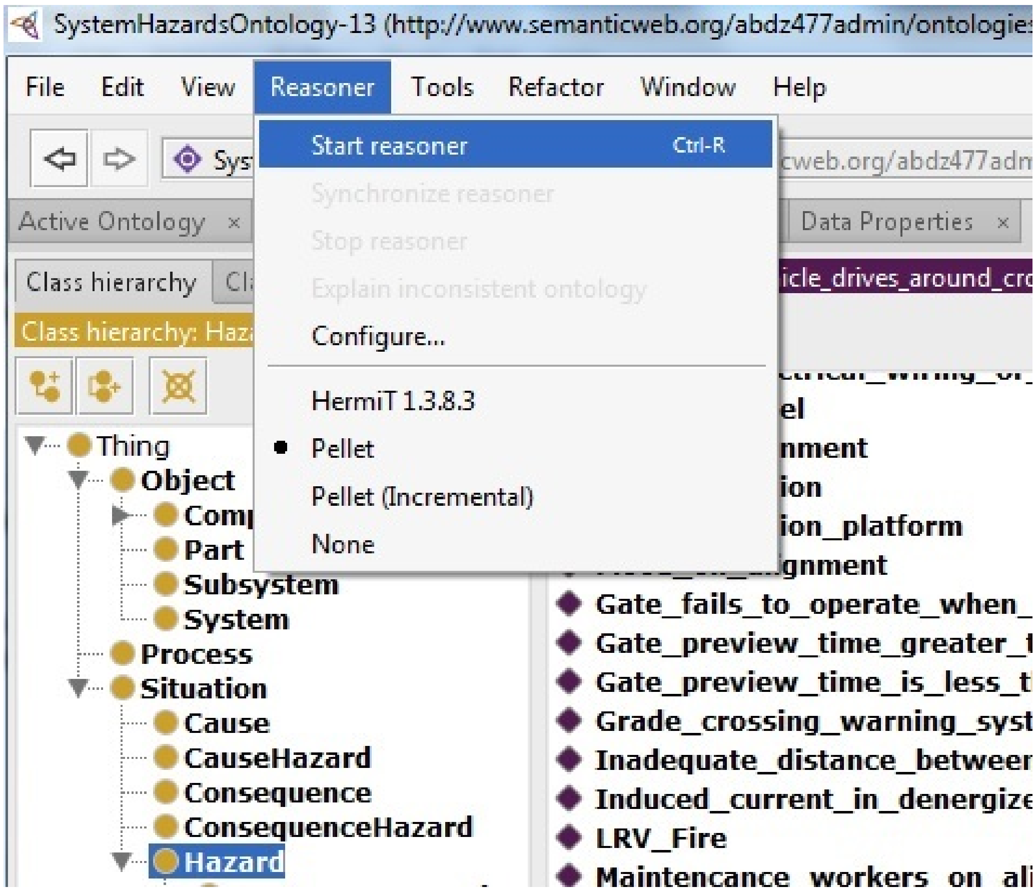


Figure 9.7: Starting the inference engine

Driving and crossing. The following hazard and cause are suggested by the reasoning engine (see Figure 9.8) to have a synonym relationship:

1. TC-7 *Road vehicle drives around crossing gate*
2. C-1 *Vehicle drives onto alignment at crossing*

where the *crossing* is the crossroad between the train path and the motor road. Statement C-1 *Vehicle drives onto alignment at crossing* is cause of Hazard A-6 *Motor vehicle on alignment.* Hazard A-6 and its cause and Hazard TC-7 *Road vehicle drives around crossing gate* refer to a motor vehicle inside the train path. Hazard *Road vehicle drives around crossing gate* appears to be referring to the vehicle going on the adjacent part of the crossing while *Vehicle drives onto alignment at crossing* seems to have two kind of meanings, either it is referring to the

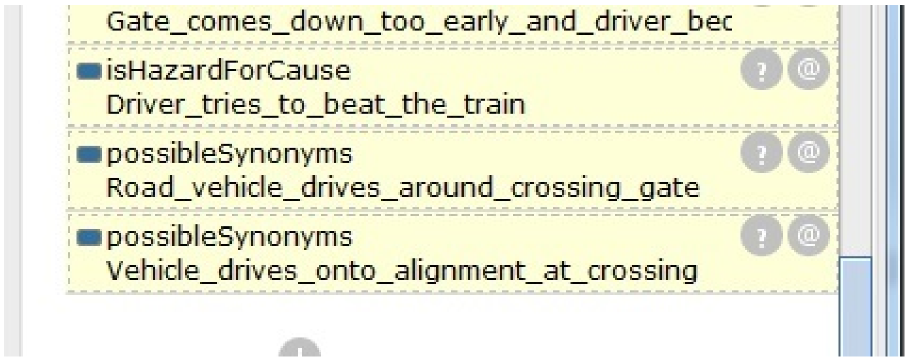

Gate_comes_down_too_early_and_driver_bec
isHazardForCause
Driver_tries_to_beat_the_train
possibleSynonyms
Road_vehicle_drives_around_crossing_gate
possibleSynonyms
Vehicle_drives_onto_alignment_at_crossing

Figure 9.8: Example results produced by Protégé for driving and crossing.

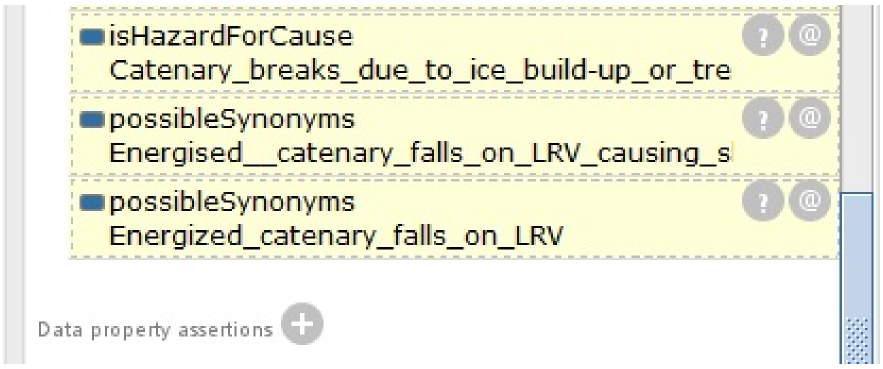

isHazardForCause
Catenary_breaks_due_to_ice_build-up_or_tre
possibleSynonyms
Energised__catenary_falls_on_LRV_causing_s
possibleSynonyms
Energized_catenary_falls_on_LRV

Figure 9.9: Example result produced by Protégé for the catenary system.

vehicle entering the actual crossing, or it could also have a general reference to the vehicle entering onto the alignment around the crossing. The second interpretation would make TC-7 and C-1 synonyms and because C-1 is cause of hazard Motor vehicle on alignment rather than *Motor vehicle on crossing* we could conclude that TC-7 and C-1 are synonyms and apply the appropriate relationship in this case.

Catenary. The following hazard and cause are suggested by the reasoning engine (see Figure 9.9) to have a synonym relationship:

1. Cause C-2 *Energised catenary falls on LRV causing shock for passengers*
2. OCS-2 *Energised catenary falls on LRV*

Stament C-2 *Energised catenary falls on LRV causing shock for passengers* is cause of hazard OCS-1 *Access to energised catenary.* In this case there is no doubt C-2 is synonym with OCS-2.

Gate arm. The results produced by the reasoning engine indicate that the following hazards might have an entailment relationship:

1. TC-2 *Broken gate arm*

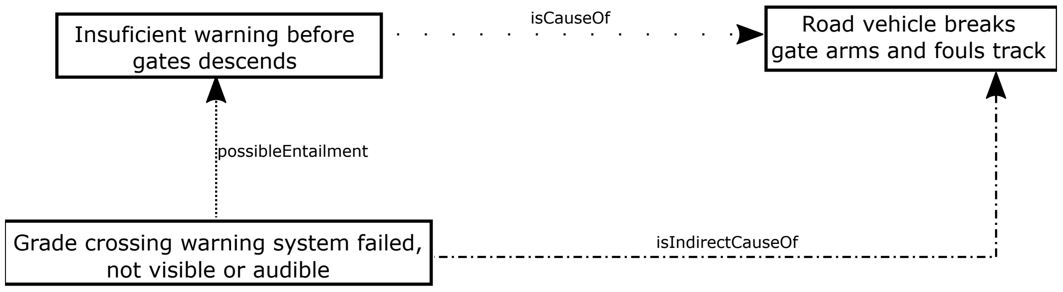

Insuficient warning before gates descends
isCauseOf
Road vehicle breaks gate arms and fouls track
possibleEntailment
Grade crossing warning system failed, not visible or audible
isIndirectCauseOf

Figure 9.10: Adding a non direct causality relationship

2. TC-8 *Road vehicle breaks gate arm and fouls track*

Hazard TC-8 entails hazard TC-2 because both cases refer to the gate arm being broken but hazard TC-8 gives an specific case for this to happen and in the case of hazard TC-2 the gate could have been broken by different other causes. Hazard TC-2 does not entail TC-8. Then, we could actually say that there is a possibility of hazard TC-8 being an indirect cause of hazard TC-2.

Grade crossing warning system. The following hazard and cause are suggested by the reasoning engine to have a synonym relationship:

1. C-3 *Insufficient warning before gate descends*
2. TC-6 *Grade crossing warning system not visible or audible*

C-3 is cause of hazard TC-8 *Road vehicle breaks gate arm and fouls track.* Cause C-3 *Insufficient warning before gate descends* seem to refer to the lack of warning, because of incorrect design or failure of the grade crossing warning system. This way, TC-6 would actually entail hazard C-3. We can also conclude that there is a causality relationship between hazard *Grade crossing warning system failed, or not visible or audible* and hazard *Road vehicle breaks gate arm and fouls track* and it is exemplified in Figure 9.10.

Electrical wiring system in stations The results produced by Protégé indicate that hazard STA-4 *Exposed electrical wiring or equipment* has a possible entailment relationship with cause C-4 *Electrical wiring fault* and this is a cause of hazard STA-1 *Fire on station platform.* It rather seems that hazard STA-4 could be a cause of *Electrical wiring fault*, if so, we would need to update *Electrical wiring fault* into a *CauseHazard.*

Generic Hazard: Fire Another answer from analysis with Protégé is that Cause *LRV stopped adjacent to a wayside fire* might relate to (*mightRelateTo*) hazards *Fire on Alignment* and *LRV Fire. LRV stopped adjacent to a wayside fire* is definitely related to *LRV Fire* because the former is a direct cause of the latter. Also, cause *LRV stopped adjacent to a wayside fire* could be related with hazard *Fire on Alignment*. Following this path of reasoning, *Fire on Alignment* then would be indirect causes of *LRV Fire* and we could add this relationship to our ontology. In addition, another answer from the reasoning engine is that hazard *Fire on Alignment* might relate to hazard *Fire in tunnel*.

False positives There are 4 groups of relations that suggest each one of them is a entailment, the reason for these is because each groups is composed of same hazard causes. In addition, Cause *Inadequate removal of heavy snow*, which is cause of hazard A-10 *Snow or ice*, is suggested to have a entailment relationship with hazard A-11 *Poor visibility on restricted alignment*. This result has been taken as a false positive. Finally, hazards A-9 *Maintenance workers on alignment in unknown location* and A-5 *Pedestrian on restricted alignment* are suggested to have an entailment relationship but this has been dismissed.

9.3.6 Step 2: Perform an automated search of hazard paths

Stage 5: Generation of State-Based Hazard Models from Ontology Rules

This is the stage where a state-based model is constructed from the hazards and causes in the ontology. In this case a model has been created from the subset of the PHA and it is based on Table 9.1. The search for hazard paths is done with the help of a model checker called MCMAS. The whole MCMAS model can be found in Appendix A.

Stage 6: Exploration of the state-based hazard model

In this stage we want to explore the state hazard model generated in order to generate hazard paths. This way, possible scenarios are identified about how hazards can develop. The focus, in this particular case, is to identify paths to hazard A-1 *LRV Fire*

The following property, expressed in CTL Logic, has been used to find these kind of paths: $\mathsf{EF}A1$ (or $\exists\Diamond A1$). In MCMAS the atomic proposition $h1$ would be represented in MCMAS like `hazard1 if Environment.A1 = true` and then we verify the following property `EF hazard1`.

Stage 7: Visualizing Hazard Paths Graphs are generated to show to the analysts possible paths from low impact hazards to higher importance hazards. For simplicity, the transition system is represented here only with the name of hazards that can be read from the results of the search. Figure 9.11 shows a path to hazard *LRV Fire*.

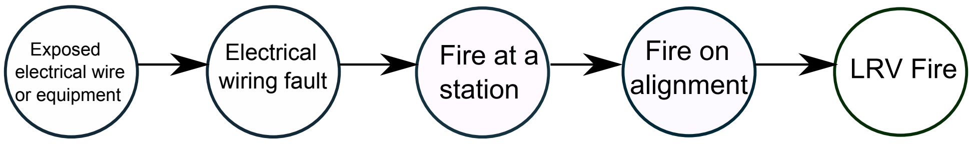


Figure 9.11: Hazard path for "LRV Fire"

9.4 Discussion

This case study was based on an industrial example, the West Corridor LRT System. A preliminary hazard analysis (PHA) was performed as part of the West Corridor LRT Project Final Engineering Design Phase [EA06], the PHA was extensive and the system was complex. The development of the ontology helped towards the modeling of the system because, despite the little information provided about the system in the project, adding more specifity to the hazards helped to understand the complexity of the system under study and model it. This is the first case study where the whole ontology based method was used. The results from the first step of the method are described in Section 9.3.5. The amount of relationships encountered showed that the method did helped to refine the hazard analysis, making it more precise and adding causal relationships. The second step of the method was also important and the generation of hazard scenarios was possible and an example was shown.

10 In-Trail Procedure Case Study

The In-Trail Procedure (ITP) is a procedure used in controlled airspace to allow one aircraft to pass over another one while flying over the Atlantic Ocean. The pilot first checks that the criteria for performing an ITP (passing) manoeuvre are satisfied, and then asks air traffic controllers (ATC) permission to execute the manoeuvre.

10.1 System Overview

A description of the procedure is given in the RTCA [SC-08] Safety Performance and Interoperability Requirements Document for the In-trail Procedure in Oceanic Airspace. Relevant excerpts are reported here:

"For a standard Flight Level change, the controller uses standard, procedure-based separation minima and procedures to ensure that separation will exist between an aircraft requesting a Flight Level change and all other aircraft at the initial, intermediate and requested Flight Levels. The ATSA-ITP was developed to enable either leading or following Same Track aircraft to perform a climb or descent to a requested Flight Level through Intervening Flight Levels that might otherwise be disallowed when using current standard separation minima. The ITP Equipment would allow the flight crew to determine if the criteria for an ITP request are met with respect to one or two Reference Aircraft at Intervening Flight Levels [...] Once these criteria are met, the flight crew may request an ITP, identifying the Reference Aircraft in the request. ATC would verify that the ITP and Reference Aircraft were Same Track and that the maximum Closing Match Differential was not exceeded [...] If the controller then determines that separation minima will be met with all Other Aircraft, the climb or descent request may be granted. The controller does not determine or verify the separation distance from the Reference Aircraft."

10.2 Preliminary Hazard Analysis for ITP

PHA is a Hazard Analysis technique generally used at the early stages of system development. It is usually combined with other hazard analysis techniques when the system under study is more developed. The PHA was informed by resources gathered from the SKYbrary [www19] knowledgebase, an online source of safety knowledge regarding flight operations, air traffic management (ATM) and aviation safety in general. It is also a portal where users can access the safety data from regulatory authorities, service providers and industry. SKYbrary has taken several years to develop and aims to contribute in promoting best practice and knowledge in aviation safety.

SKYbrary provides articles about different operational issues, human performance, air-ground communications among many other topics involving flight safety. Among the many operational issues presented in SKYbrary, substantive information can be found about Lost of Separation, Level Bust, call sign confusion. These topics are vastly explained, they have a description, the effects of these operational issues, typical scenarios and contributory factors, as well as examples of some accidents and incidents involving them.

Based on this knowledge, I have then formulated a list of hazards, causes and consequences presented. The main system hazard is: "Aircraft violates minimum separation". The initial version of the PHA can be seen in Table 10.1.

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
H	Aircraft violates minimun separation	Aircraft fails to fly at the level to which it has been cleared (Level Bust).	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Flight crew is not flying instructed or expected speed or rates of climb (or any other ITP setting).	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Flight crew is unable to act fast enough to new clearance and passes through new cleared level.	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Flight crew is unable to proceed fast enough to an ITP clearance	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Flight crew executes a cleared ITP which does not meet criteria	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Pilot executes ITP procedure after request but without clearance	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Pilot aborts an ITP clearance when criteria is met	Death of people due to a mid-air collision
H	Aircraft violates minimun separation	Pilot does not execute necessary abortion from previous clearance	Death of people due to a mid-air collision
h_1	Aircraft fails to fly at the level to which it has been cleared (Level Bust)	The ATC reassigns a FL after a clearance has been passed already	Aircraft violates minimun separation
h_1	Aircraft fails to fly at the level to which it has been cleared (Level Bust)	Flight crew does not follow an ATC clearance	Aircraft violates minimun separation

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
h_2	Flight crew is not flying instructed or expected speed or rates of climb (or any other ITP setting)	Flight crew accepts a clearance correctly but unadvertedly set it up incorrectly	Aircraft violates minimun separation
h_2	Flight crew is not flying instructed or expected speed or rates of climb (or any other ITP setting)	Flight crew accepts and records a clearance correctly but does not follow it	Aircraft violates minimun separation
h_3	Flight crew is unable to act fast enough to new clearance and passes through new cleared level	The ATC passed a reclearance at last moment due to former incorrect clearance.	Aircraft violates minimun separation
h_4	Flight crew is unable to proceed fast enough to an ITP clearance	Pilot is not focus on task of getting clearance done	Aircraft violates minimun separation
h_4	Flight crew is unable to proceed fast enough to an ITP clearance	The ITP clearance is passed late	Aircraft violates minimun separation
h_5	Flight crew executes ITP procedure after request but without clearance	Flight crew misses or incorrectly interprets a message from ATC (e.g. clerance denied)	Aircraft violates minimun separation
h_5	Flight crew executes ITP procedure after request but without clearance	Pilot is too confident in getting clearance from ATC	Aircraft violates minimun separation
h_6	Flight crew executes a cleared ITP which does not meet criteria	Controller unadvertedly provides a clearance where separation is inadequate	Aircraft violates minimun separation
h_6	Flight crew executes a cleared ITP which does not meet criteria	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	Aircraft violates minimun separation
h_6	Flight crew executes a cleared ITP which does not meet criteria	ATC confirms a clerance without verification	Aircraft violates minimun separation

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
h_7	Pilot aborts an ITP clearance when criteria is met	ATC sends abort request to wrong aircraft	Aircraft violates minimun separation
h_7	Pilot aborts an ITP clearance when criteria is met	ATC sends abort request to aircraft when it is not needed	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	Pilot fails to follow abort instructions	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	ATC fails to send an abort request within reasonable time	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	Pilot lacks knowledge about how to proceed when abnormal termination is judged necessary	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	Equipment does not react to abort settings	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	ATC does not send necessary abort request	Aircraft violates minimun separation
h_8	Pilot does not execute necessary abortion from previous clerance	ATC sends abort request to wrong aircraft	Aircraft violates minimun separation
$h_{1.1}$	The ATC reassigns a FL after a clearance has been passed already	ATC approves ITP with low awareness of current conflicting traffic	Aircraft fails to fly at the lvel it has been cleared (level bust)
$h_{1.2}$	Flight crew does not follow an ATC clearance	Lack of competence of the air crew	Aircraft fails to fly at the lvel it has been cleared (level bust)
$h_{1.2}$	Flight crew does not follow an ATC clearance	Negligence	Aircraft fails to fly at the lvel it has been cleared (level bust)

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
$h_{1.2}$	Flight crew does not follow an ATC clearance	Equipment malfunction	Aircraft fails to fly at the lvel it has been cleared (level bust)
$h_{2.1}$	Flight crew accepts a clearance correctly but unadvertedly set it up incorrectly	The pilot is distracted from his primary tasks and performs an innapropiate action on the equipment	Flight crew is not fying instructed or expected speeds or rates of climb (and descend)
$h_{2.1}$	Flight crew accepts a clearance correctly but unadvertedly set it up incorrectly	The pilot is tired and overworked	Flight crew is not fying instructed or expected speeds or rates of climb (and descend)
$h_{2.1}$	Flight crew accepts a clearance correctly but unadvertedly set it up incorrectly	The pilot left in charge is not trained properly	Flight crew is not fying instructed or expected speeds or rates of climb (and descend)
$h_{2.2}$	Flight crew accepts and records a clearance correctly but does not follow it	A discussion of non relevant matter distract pilots from duties (flight management error)	Flight crew is not fying instructed or expected speeds or rates of climb (and descend)
$h_{2.2}$	Flight crew accepts and records a clearance correctly but does not follow it	Aircraft technical equipment (e.g. altitude alert) does not operate as designed	Flight crew is not fying instructed or expected speeds or rates of climb (and descend)
$h_{3.1}$	The ATC passed a re-clearance at last moment due to former incorrect clearance	Lack of situational awareness of the ATC	Flight crew is unable to act fast enough to new clearance and passes through new cleared level
$h_{4.1}$	The ITP clearance is passed late	ATC unable to cope with a sudden increase in workload	Flight crew is unable to proceed fast enough to an ITP clearance
$h_{4.1}$	The ITP clearance is passed late	Delay in communication due to blocked transmision or interfearance	Flight crew is unable to proceed fast enough to an ITP clearance

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
$h_{4.1}$	The ITP clearance is passed late	ATC is distracted from primary tasks.ATC is distracted from primary tasks	Flight crew is unable to proceed fast enough to an ITP clearance
$h_{5.1}$	Flight crew misses or incorrectly interprets a message from ATC (e.g. clearance denied)	The ATC issues a complex transmission containing more than two instructions (e.g., speed, altitude and heading)	Flight crew executes ITP procedure after request but without clearance
$h_{5.1}$	Flight crew misses or incorrectly interprets a message from ATC (e.g. clearance denied)	Inadequate English proficiency, or use of standard phraseology	Flight crew executes ITP procedure after request but without clearance
$h_{5.1}$	Flight crew misses or incorrectly interprets a message from ATC (e.g. clearance denied)	Inadequate speed of transmission	Flight crew executes ITP procedure after request but without clearance
$h_{5.1}$	Flight crew misses or incorrectly interprets a message from ATC (e.g. clearance denied)	Communication not clear because of radio interfearance or blocked transmission	Flight crew executes ITP procedure after request but without clearance
$h_{5.2}$	Pilot is too confident in getting clearance from ATC	Lack of discipline and competence from Flight crew	Flight crew executes ITP procedure after request but without clearance
$h_{6.1}$	ATC confirms a clearance without verification	The pilot mishears the level clearance and he does not read back the clearance	Flight crew executes a cleared ITP which does not meet criteria
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	ATC does not correct a misreading of level or altitude from the pilot and approves it	Flight crew executes a cleared ITP which does not meet criteria
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	ATC has not heard an erroneous readback and has not corrected it	Flight crew executes a cleared ITP which does not meet criteria

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	Controller makes a misjudgement about an inappropiate separation	Flight crew executes a cleared ITP which does not meet criteria
$h_{6.3}$	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	Call signs coincidentally contain the same alphanumeric characters in a different order (e.g. AB1234 and BA 2314)	Flight crew executes a cleared ITP which does not meet criteria
$h_{6.3}$	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	Airlines schedule flights with similar call signs to be in the same airspace at the same time	Flight crew executes a cleared ITP which does not meet criteria
$h_{6.3}$	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	Airlines allocate commercial flight numbers as call-signs; these are normally consecutive and therefore similar (e.g. RUSHAIR 1431, RUSHAIR 1432, etc)	Flight crew executes a cleared ITP which does not meet criteria
$h_{7.1}$	ATC sends abort request to wrong aircraft	ATC operator is interrupted or distracted	Pilot aborts an ITP clearance when criteria is met
$h_{7.1}$	ATC sends abort request to wrong aircraft	Call signs coincidentally contain the same alphanumeric character in a different order	Pilot aborts an ITP clearance when criteria is met
$h_{7.2}$	ATC sends abort request to aircraft when it is not needed	ATC is dissoriented due to conflicting sources of information	Pilot aborts an ITP clearance when criteria is met
$h_{7.2}$	ATC sends abort request to aircraft when it is not needed	ATC is not properly aware of the surrounding traffic	Pilot aborts an ITP clearance when criteria is met
$h_{8.1}$	ATC fails to send an abort request within reasonable time	The ATC reacts late to a wrong clearance	Pilot does not execute necessary abortion from previous clearance

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
$h_{8.2}$	Pilot fails to follow abort instructions	The ATC instructions are misunderstood	Pilot does not execute necessary abortion from previous clearance
$h_{8.3}$	ATC does not send abort request	Unavailability due to ATC being understaffed	Pilot does not execute necessary abortion from previous clearance
$h_{1.1.1}$	ATC approves ITP with low awareness of current conflicting traffic	Updates from aircraft data are corrupted or delayed	The ATC reassigns a FL after a clearance has been passed already
$h_{1.1.1}$	ATC approves ITP with low awareness of current conflicting traffic	Disorientation about aircraft track or level	The ATC reassigns a FL after a clearance has been passed already
$h_{3.1.1}$	Lack of situational awareness of the ATC	Interruptions and distractions at the ATC office	The ATC passed a reclearance at last moment due to former incorrect clearance
$h_{3.1.1}$	Lack of situational awareness of the ATC	Lack of training of ATC personnel	The ATC passed a reclearance at last moment due to former incorrect clearance
$h_{5.1.1}$	The ATC issues a complex transmission containing more than two instructions (e.g. speed, altitude and heading)	ATC lacks discipline in communications	Flight crew misses or incorrectly interprets a message fron ATC (e.g. clearance denied)
$h_{6.2.1}$	Controller makes a misjudgement about an inappropiate separation	Pilot confuses criteria for requesting ITP	Controller unadvertedly provides a clearance where separation is inadecuate
$h_{6.2.1}$	Controller makes a misjudgement about an inappropiate separation	Pilot request ITP not knowing that it does not meet criteria	Controller unadvertedly provides a clearance where separation is inadecuate

Table 10.1: Preliminary Hazard Analysis for ITP

N0	*Hazard*	*Cause*	*Consequence*
$h_{6.2.2}$	ATC has not heard an erroneous read-back and has not corrected it	ATC uses read back time to carry out other tasks	Controller unadvertedly provides a clearance where separation is inadecuate
$h_{6.2.3}$	ATC does not correct a misreading of level or altitude from the pilot and approves it	Pilot reads back incorrect clearance within a complex transmission	Controller unadvertedly provides a clearance where separation is inadecuate
$h_{6.2.1.1}$	Pilot request ITP not knowing that it does not meet criteria	Aircraft is giving wrong data about aircraft position with regards ITP	Controller makes a misjudgement about an inappropiate separation
$h_{6.2.1.2}$	Pilot confuses criteria for requesting ITP	Lack of training	Controller makes a misjudgement about an inappropiate separation
$h_{6.2.1.2}$	Pilot confuses criteria for requesting ITP	Tiredness	Controller makes a misjudgement about an inappropiate separation
$h_{6.2.1.1.1}$	Aircraft is giving wrong data about aircraft position with regards ITP	Missetting of aircraft equipment	Pilot request ITP not knowing that it does not meet criteria
$h_{6.2.1.1}$	Aircraft is giving wrong data about aircraft position with regards ITP	Inattention to equipment malfunction	Pilot request ITP not knowing that it does not meet criteria
$h_{6.2.1.1}$	Aircraft is giving wrong data about aircraft position with regards ITP	Malfunctions in communication device	Pilot request ITP not knowing that it does not meet criteria

10.2.1 Application of the Ontology-Based Method to the ITP Case Study

This section presents the application of the ontology-based presented in this thesis to the ITP case study. It will be shown how the method provides a structure (or path) for evolving the results of the hazard analysis. Also, it will be shown how the method helps the analyst to identify scenarios that can generate deeper understanding of the causes of certain hazards.

Step 1: Carry out a systematic exploration of dependencies

This step has 4 stages. In this particular case study, the stages are instantiated as follows:

- Stage 1, would be the collected information about, aviation and safety, and also the ITP Procedure.
- Stage 2, would be the revision of the PHA for ITP with respect to all the information that has been collected.
- Stage 3, would be the development of two diagrams. First, a diagram of the processes that involve our ITP System. Second, a diagram of structure of the ITP System. These diagrams are also represented in our ontology.
- Stage 4, would be Search implicit and overlooked relationships.

Following, Stage 3 and Stage 4 will describe in more detail.

Stage 1: Hazard Analysis and System Description The main sources of information were the description of the ITP procedure in the RTCA requirements document [SC-08] and the SKYbrary [www19] knowledgebase.

Stage 2: Systematic Revision A systematic revision of the PHA results is carried out by comparing hazards and related causes with those identified by Leveson et al in [FC12], where the same ITP system was analysed using the System Theoretic Process Analysis (STPA).

In order to facilitate the reading of this section, a primer on STPA is now provided. STPA represents the system under analysis as a control model. In its simplest form, the control model has one *controller* and one *controlled process*. Hazards and related causal factors are explored by analysing the control relations in the model. Specifically, the analysis is carried out in three main steps:

- *Step 1: Identifying Unsafe Control Actions.* In this step, the analyst needs to identify the actions that could potentially lead the system in a hazardous system state. In addition, the analyst also needs to identify the absence of certain actions that could lead the system in a hazardous system state.

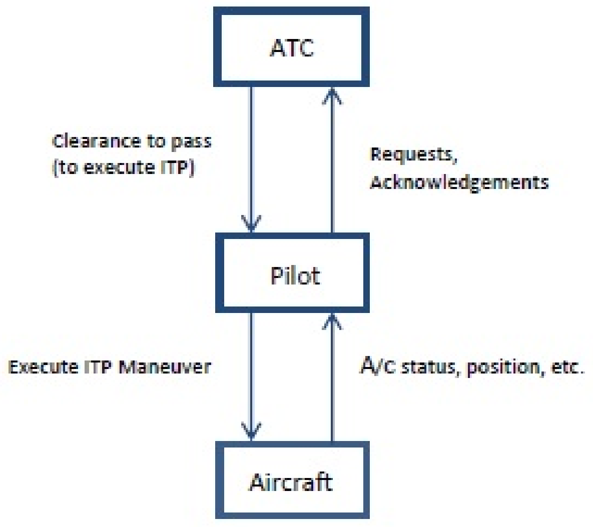

ATC
Clearance to pass (to execute ITP)
Requests, Acknowledgements
Pilot
Execute ITP Maneuver
A/C status, position, etc.
Aircraft

Figure 10.1: Basic Control Structure for ITP (Taken from [Lev13])

- *Step 2: Identifying the causes of the Unsafe Control Actions.* Once that the unsafe control actions are identified, the next step is to identify potential causal factors for each unsafe control action.
- *Step 3: Identify safety requirements.* When the second step of the STPA analysis has been finished, the found causes should be eliminated or controlled in the design at the system level or they must be translated into requirements at the system components.

The control structure of the ITP system, as defined in [FC12], is shown in Figure 10.1. In the model, the controllers are the Pilot (Flight Crew) and ATC. Unsafe control actions are thus identified using a set of guide phrases.

Unsafe control actions for the Flight Crew controller are:

1. ITP executed when not approved.
2. ITP executed when ITP criteria are not satisfied.
3. ITP executed with incorrect climb rate, final altitude, etc.
4. ITP executed too soon before approval.
5. ITP executed too late after.
6. Flight Crew continues with maneuveur in dangerous situation.
7. Flight Crew aborts unnecessarily.
8. Flight Crew does not follow regional procedure while aborting.

Similarly, unsafe control action for the ATC are:

1. Approval given when criteria are not met.
2. Approval given to incorrect aircraft.
3. Approval given too early.
4. Approval given too late.
5. Aircraft should abort but instructions not given.
6. Abort instructions given when abort is not necessary.
7. Abort instructions given too late.

The second step of the STPA analysis identifies what element in system design could trigger the unsafe control action (see section 3.2.1). At the end of this step, each unsafe control action will have a description of its causes with respect to the process model where this cause is supposed to occur. In order to help with this task, two different graphs called control loops are created (See Figures 10.2 and 10.2), these control loops are related to two controllers, the ATC and the ITP flight crew. More safety constraints are identified after the second part of the STPA analysis is performed.

The causal factors for the ITP case study, as identified in [FC12], are in Table 10.2. To facilitate a systematic review of the PHA results, relevant PHA results are also shown in Table 10.2. As it can be seen in the table, each causal factor identified by the STPA analysis can be related to one or more hazards of the PHA analysis. This increases our confidence that the initial PHA analysis does not have any clear deficiency.

Stage 3: Ontology guided revision of inputs The behavioural and structural representations of the ITP system are now developed to perform the ontology-guided revision of the Preliminary Hazard Analysis of ITP.

The behavioural representation is made of processes. Figure 10.4 shows the processes of the ITP system. The main process is called *ITP Process*, all processes would be sub-process of this main process. Namely, the main *ITP Process* has three direct sub-processes (see Figure 10.4): *ITP Approval Process*, *Performing ITP*, and *Terminating ITP*. These sub-processes represent the three phases of the ITP Procedure related to: the evaluation of the ITP criteria; the request sent to ATC; and the final step of the procedure, where the Flight Crew either reaches the altitude level where the aircraft was cleared, or aborts the execution and remains at the same level. Figure 10.5 shows some of these processes represented in the Protégé ontology.

The structural representation for the ITP System Figure 10.6. It has two subsystems, the *ITP Communication Subsystem* and the *ITP Flight Subsystem*. The *ITP Communication Subsystem* includes all components involved in the ITP clearance request. The *ITP Flight*

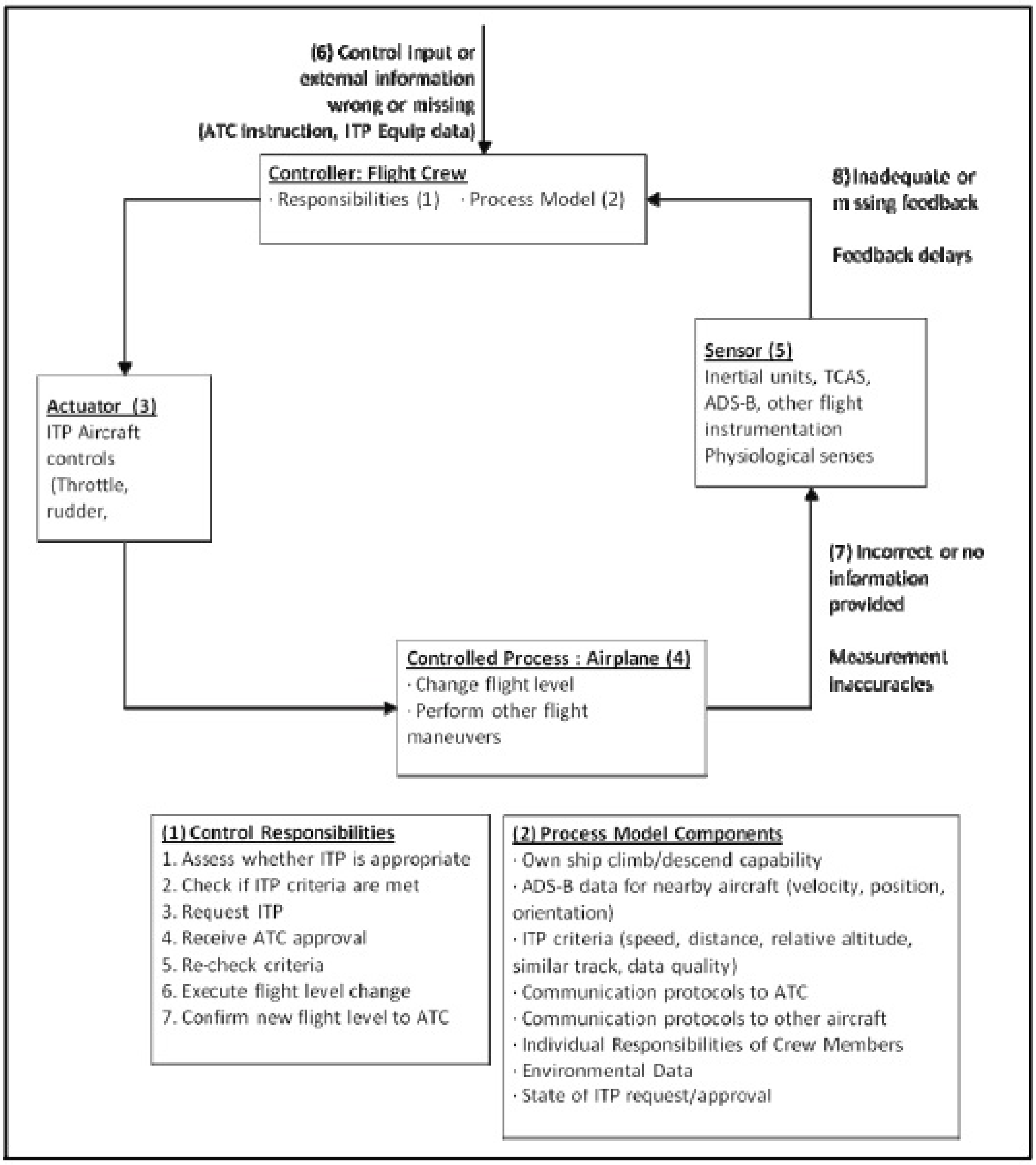

(6) Control Input or external information wrong or missing (ATC instruction, ITP Equip data)
Controller: Flight Crew
· Responsibilities (1) · Process Model (2)
8) Inadequate or missing feedback
Feedback delays
Actuator (3)
ITP Aircraft controls (Throttle, rudder,
Sensor (5)
Inertial units, TCAS, ADS-B, other flight instrumentation
Physiological senses
(7) Incorrect or no information provided
Measurement inaccuracies
Controlled Process : Airplane (4)
· Change flight level
· Perform other flight maneuvers
(1) Control Responsibilities
1. Assess whether ITP is appropriate
2. Check if ITP criteria are met
3. Request ITP
4. Receive ATC approval
5. Re-check criteria
6. Execute flight level change
7. Confirm new flight level to ATC
(2) Process Model Components
· Own ship climb/descend capability
· ADS-B data for nearby aircraft (velocity, position, orientation)
· ITP criteria (speed, distance, relative altitude, similar track, data quality)
· Communication protocols to ATC
· Communication protocols to other aircraft
· Individual Responsibilities of Crew Members
· Environmental Data
· State of ITP request/approval

Figure 10.2: Control Loop (Taken from [FC12])

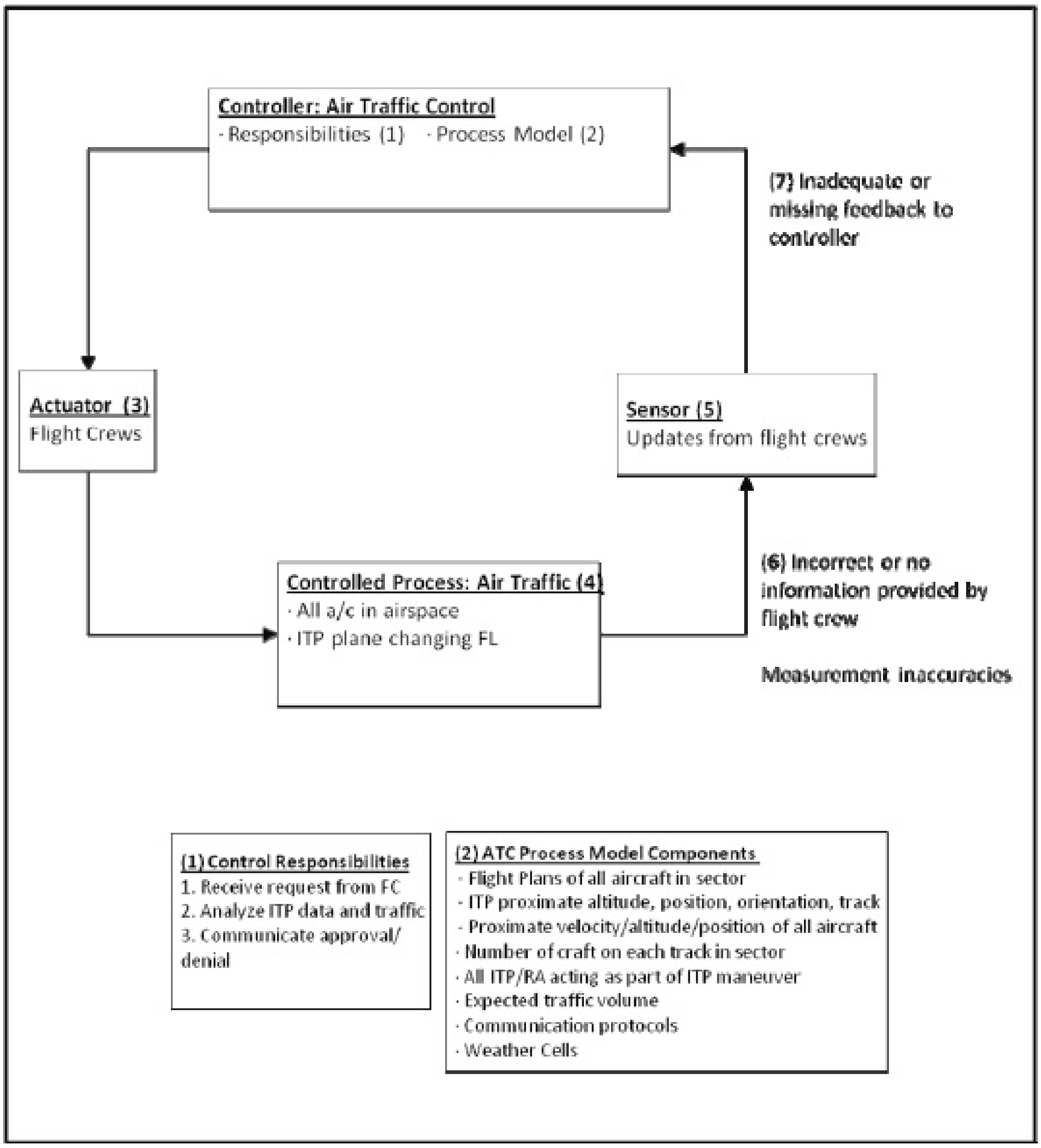

Controller: Air Traffic Control
- Responsibilities (1) - Process Model (2)
(7) Inadequate or missing feedback to controller
Actuator (3)
Flight Crews
Sensor (5)
Updates from flight crews
Controlled Process: Air Traffic (4)
· All a/c in airspace
· ITP plane changing FL
(6) Incorrect or no information provided by flight crew
Measurement inaccuracies
(1) Control Responsibilities
1. Receive request from FC
2. Analyze ITP data and traffic
3. Communicate approval/ denial
(2) ATC Process Model Components
- Flight Plans of all aircraft in sector
- ITP proximate altitude, position, orientation, track
- Proximate velocity/altitude/position of all aircraft
· Number of craft on each track in sector
· All ITP/RA acting as part of ITP maneuver
· Expected traffic volume
· Communication protocols
· Weather Cells

Figure 10.3: Control Loop(Taken from [FC12])

Table 10.2: Hazards of PHA for ITP vs Unsafe control actions of STPA applied to ITP

N^0	UCA or Hazard	Definition
1	UCA h_5	ITP executed when not approved. Flight crew executes ITP procedure after request but without clearance.
2	UCA h_6	ITP executed when ITP criteria are not satisfied. Flight crew executes a cleared ITP which does not meet criteria
3	UCA h_2	ITP executed with incorrect climb rate, final altitude, etc. Flight crew is not flying instructed or expected speeds or rates of climb (or any other ITP setting).
4	UCA h_5	ITP executed too soon before approval. Flight crew executes ITP procedure after request but without clearance.
5	UCA h_4	ITP executed too late. Flight crew unable to proceed fast enough to an ITP clearance.
6	UCA $h_{8.2}$	Flight Crew continues with maneuveur in dangerous situation. Pilot fails to follow abort instructions.
7	UCA h_7	Flight Crew aborts unnecessarily. Pilot aborts an ITP clearance when criteria is met.
8	UCA $h_{6.2}$	Approval given when criteria are not met. Controller unadvertedly provides a clearance where separation is ineadequate.
9	UCA $h_{6.3}$	Approval given to incorrect aircraft. The pilot accepts a level clearance intended for another aircraft.
10	UCA $h_{4.1}$	Approval given too late. The ITP clearance is passed late.
11	UCA $h_{8.3}$ $h_{7.1}$	Aircraft should abort but instructions not given. ATC does not send abort requests. ATC sends abort request to wrong aircraft.
12	UCA $h_{7.2}$	Abort instructions given when abort is not necessary. ATC sends abort request to aircraft when it is not needed.
13	UCA $h_{8.1}$	Abort instructions given too late. ATC fails to send abort request within reasonable time.

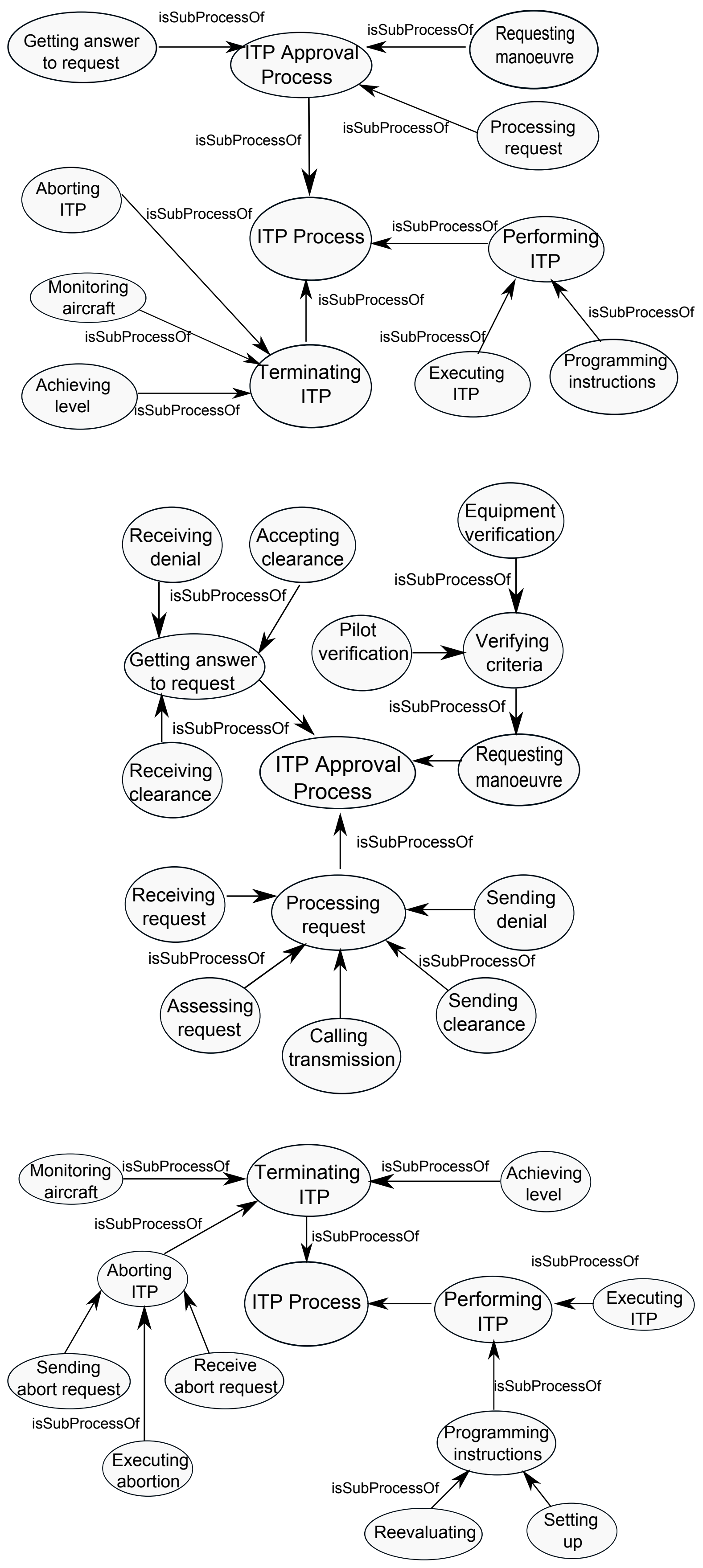
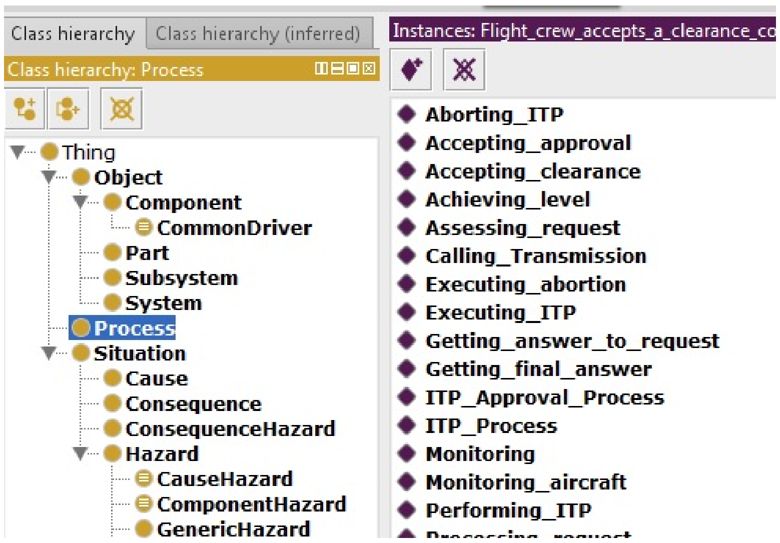


Figure 10.4: Processes of the ITP System


Class hierarchy
Class hierarchy (inferred)
Class hierarchy: Process
Instances: Flight_crew_accepts_a_clearance_co

```
Thing
    Object
        Component
            CommonDriver
        Part
        Subsystem
        System
    Process
    Situation
        Cause
        Consequence
        ConsequenceHazard
        Hazard
            CauseHazard
            ComponentHazard
            GenericHazard
```

- Aborting_ITP
- Accepting_approval
- Accepting_clearance
- Achieving_level
- Assessing_request
- Calling_Transmission
- Executing_abortion
- Executing_ITP
- Getting_answer_to_request
- Getting_final_answer
- ITP_Approval_Process
- ITP_Process
- Monitoring
- Monitoring_aircraft
- Performing_ITP

Figure 10.5: Subset of Processes of the ITP System in Protégé

Subsystem includes all components involved in the execution of flight level once a clearance has been performed.
The components that are part of the *ITP Communication Subsystem* are the following:

- ATC
- Transmission
- Aircraft
- Flight Crew

The components that are part of the *ITP Flight Subsystem* are the following:

- Aircraft
- Flight Crew

As we can notice, components *Aircraft* and *Flight Crew* belong to both subsystems. These kind of components, in our representation, are called *common drivers*. Figure 10.7 shows these represented in the ontology.

Finally, not only we have represented in our ontology these behavioural and physical structures of the ITP System, but also the hazards and cause found in the PHA for ITP, shown in Table 10.1. Figure 10.8 shows a subset of all the set of hazards and causes represented in the ontology.

Stage 4: Search implicit and overlooked relationships The inference engine of Protégé is now used to find entailment or synonym relationships. The inference rules I have created to perform the analysis in Protégé are explained in Section 4.3.4 of Chapter 4.

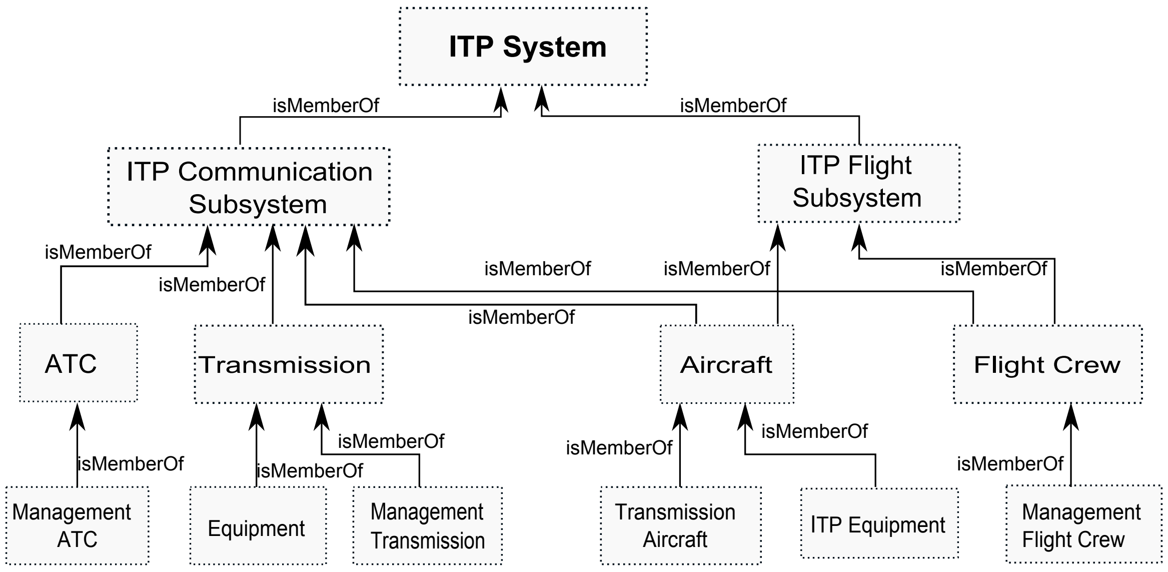


Figure 10.6: Parts of ITP System

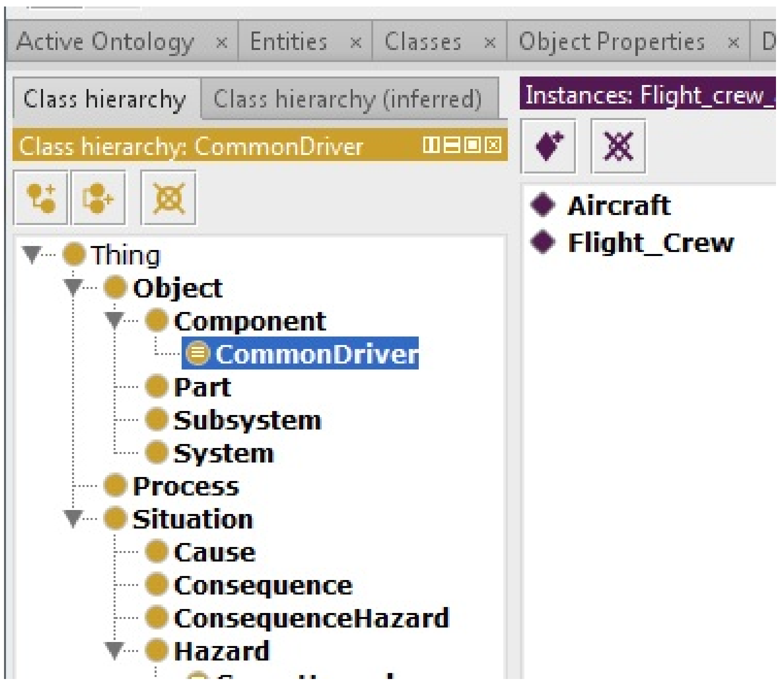
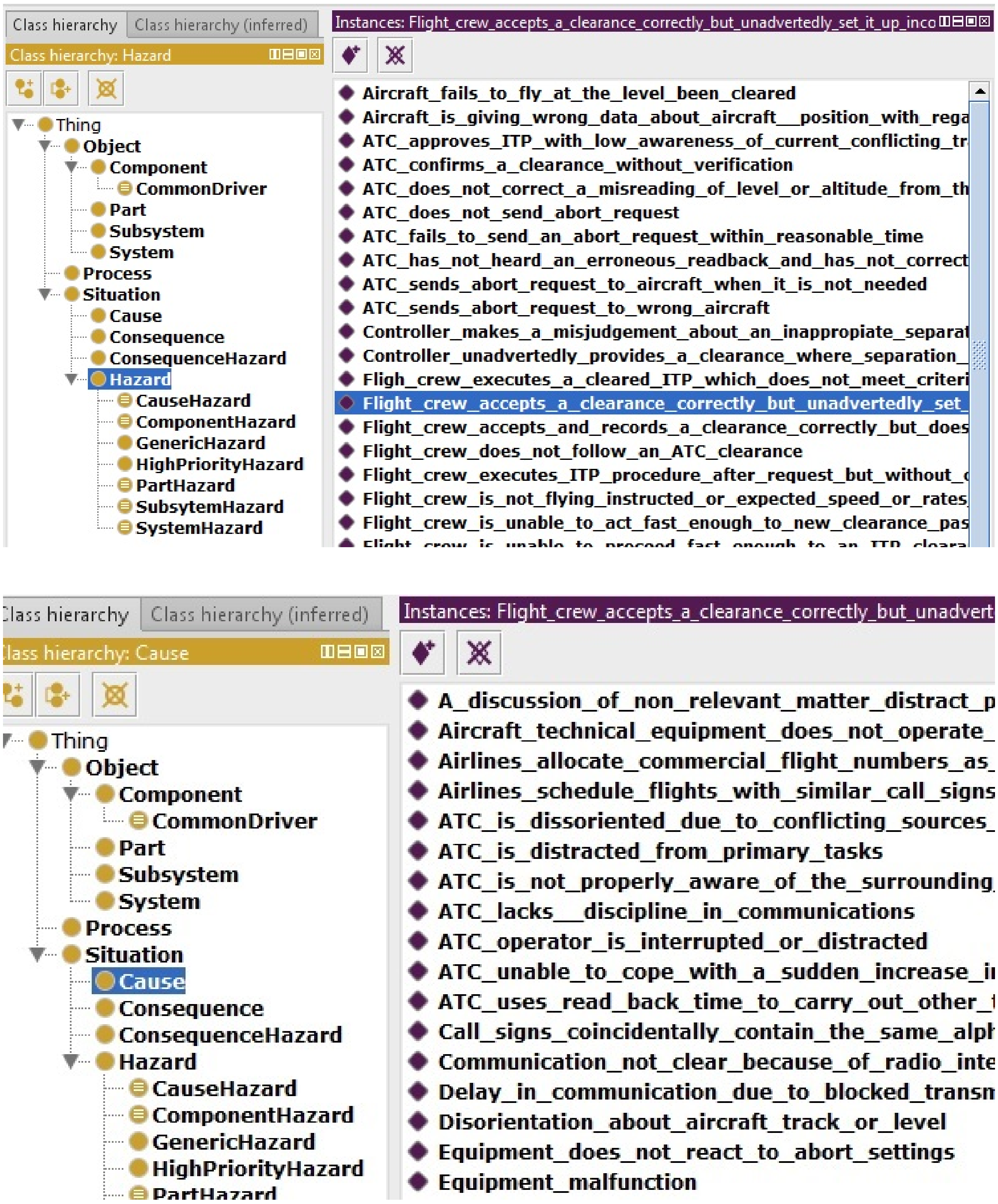


Figure 10.7: Subset of Processes of the ITP System in Protégé


Class hierarchy
Class hierarchy (inferred)
Class hierarchy: Hazard
Instances: Flight_crew_accepts_a_clearance_correctly_but_unadvertedly_set_it_up_inco

Thing
- Object
  - Component
    - CommonDriver
  - Part
  - Subsystem
  - System
- Process
- Situation
  - Cause
  - Consequence
  - ConsequenceHazard
  - Hazard
    - CauseHazard
    - ComponentHazard
    - GenericHazard
    - HighPriorityHazard
    - PartHazard
    - SubsytemHazard
    - SystemHazard

Aircraft_fails_to_fly_at_the_level_been_cleared
Aircraft_is_giving_wrong_data_about_aircraft__position_with_rega
ATC_approves_ITP_with_low_awareness_of_current_conflicting_tr
ATC_confirms_a_clearance_without_verification
ATC_does_not_correct_a_misreading_of_level_or_altitude_from_th
ATC_does_not_send_abort_request
ATC_fails_to_send_an_abort_request_within_reasonable_time
ATC_has_not_heard_an_erroneous_readback_and_has_not_correct
ATC_sends_abort_request_to_aircraft_when_it_is_not_needed
ATC_sends_abort_request_to_wrong_aircraft
Controller_makes_a_misjudgement_about_an_inappropiate_separat
Controller_unadvertedly_provides_a_clearance_where_separation_
Fligh_crew_executes_a_cleared_ITP_which_does_not_meet_criteri
Flight_crew_accepts_a_clearance_correctly_but_unadvertedly_set_
Flight_crew_accepts_and_records_a_clearance_correctly_but_does
Flight_crew_does_not_follow_an_ATC_clearance
Flight_crew_executes_ITP_procedure_after_request_but_without_c
Flight_crew_is_not_flying_instructed_or_expected_speed_or_rates
Flight_crew_is_unable_to_act_fast_enough_to_new_clearance_pas

Class hierarchy
Class hierarchy (inferred)
Class hierarchy: Cause
Instances: Flight_crew_accepts_a_clearance_correctly_but_unadvert

Thing
- Object
  - Component
    - CommonDriver
  - Part
  - Subsystem
  - System
- Process
- Situation
  - Cause
  - Consequence
  - ConsequenceHazard
  - Hazard
    - CauseHazard
    - ComponentHazard
    - GenericHazard
    - HighPriorityHazard
    - PartHazard

A_discussion_of_non_relevant_matter_distract_p
Aircraft_technical_equipment_does_not_operate_
Airlines_allocate_commercial_flight_numbers_as_
Airlines_schedule_flights_with_similar_call_signs
ATC_is_dissoriented_due_to_conflicting_sources_
ATC_is_distracted_from_primary_tasks
ATC_is_not_properly_aware_of_the_surrounding
ATC_lacks__discipline_in_communications
ATC_operator_is_interrupted_or_distracted
ATC_unable_to_cope_with_a_sudden_increase_i
ATC_uses_read_back_time_to_carry_out_other_
Call_signs_coincidentally_contain_the_same_alpł
Communication_not_clear_because_of_radio_inte
Delay_in_communication_due_to_blocked_transm
Disorientation_about_aircraft_track_or_level
Equipment_does_not_react_to_abort_settings
Equipment_malfunction

Figure 10.8: Subset of Hazards and Causes of the ITP System in Protégé

Process: Approval of ITP The following hazards, from the PHA, have been found by the reasoning engine to possibly have an entailment relationship. After these hazards have been highlighted, one needs to analyse these hazards in order to update our ontology, if it is concluded that some replacement should be done.

1. $h_{1.1}$ "The ATC reassigns a Flight Level after a clearance has been passed already". In this context, where *reassigning a flight level* means try to correct a given flight level because the flight level given with the former clearance was not the correct one.

2. $h_{3.1}$ "The ATC passed a re clearance at last moment due to incorrect clearance", where *re clearance* means that a new clearance has given to the flight crew because the previous one was found incorrect, and *last moment* refers to the short time the ATC has before the flight crew starts to execute the previous clearance.

The hazard "The ATC reassigns a Flight Level after a clearance has been passed already" could be taken as a bit more generic than the hazard "The ATC passed a re clearance at last moment due to incorrect clearance". A discussion with domain experts is needed here to understand whether the two hazards have the same meaning or they are different.

Based on these results, three actions can be made to improve the analysis:

1. I can confirm that both hazards have an entailment relationship and make this relationship explicit in the ontology and add the relationship $isIndirectCauseOf()$ respectively to the causes of the hazards.

2. I can conclude that both hazards have same meaning but I want them to be in the ontology so I will make the synonym relationship explicit in the ontology add the relationship $isIndirectCauseOf()$ respectively to the causes of the hazards.

3. I can conclude that the two hazards have same meaning and erase one of them.

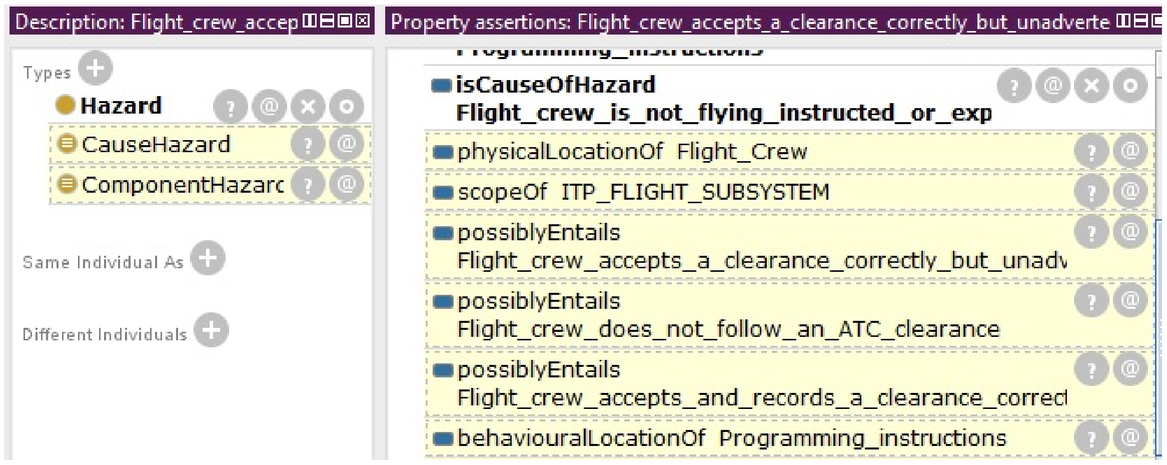
Process: Processing clearance request The following hazards are also found by the reasoning engine to possibly hold an entailment relationship.

1. $h_{1.1.1}$ "ATC approves ITP with low awareness of current conflicting traffic".

2. $h_{3.1.1}$ "Lack of situational awareness of the ATC".

The hazard "Lack of situational awareness of the ATC" seems more general than "ATC approves ITP with low awareness of current conflicting traffic". This is because in both cases it is highlighted the lack of situational awareness. But, in the case of the second hazard, there is an action taken by the ATC as well, which is the approval of the ITP request.

In order to better understand the context of these two hazards, we can also take a further look into the PHA. Hazard "Lack of situational awareness of the ATC" is a causal factor of


Description: Flight_crew_accep

Types

Hazard

CauseHazard

ComponentHazarc

Same Individual As

Different Individuals

Property assertions: Flight_crew_accepts_a_clearance_correctly_but_unadverte

Programming_instructions

isCauseOfHazard Flight_crew_is_not_flying_instructed_or_exp

physicalLocationOf Flight_Crew

scopeOf ITP_FLIGHT_SUBSYSTEM

possiblyEntails Flight_crew_accepts_a_clearance_correctly_but_unadv

possiblyEntails Flight_crew_does_not_follow_an_ATC_clearance

possiblyEntails Flight_crew_accepts_and_records_a_clearance_correct

behaviouralLocationOf Programming_instructions

Figure 10.9: Results shown in Protégé

hazard “The ATC passed a re clearance at last moment due to incorrect clearance”. Hazard “ATC approves ITP with low awareness of current conflicting traffic” is causal factor of hazard “The ATC reassigns a Flight Level after a clearance has been passed already”. Based on this one could reasonably argue that hazards $h_{1.1}$ “The ATC reassigns a Flight Level after a clearance has been passed already” and $h_{3.1}$ “The ATC passed a re clearance at last moment due to incorrect clearance” have same meaning. Therefore, the hazards we are comparing now are also intended to mean the same.

As in the previous case, discussion with domain experts is necessary to understand if the two hazards are equivalent.

Process: Performing ITP The following hazards have been found, by the reasoning engine, to possibly have an entailment relationship:

1. $h_{2.1}$ “Flight crew accepts a clearance correctly but unadvertedly set it up incorrectly”.

2. $h_{2.2}$ “Flight crew accepts and records a clearance correctly but does not follow it”.

3. $h_{1.2}$ “Flight crew does not follow an ATC clearance”.

The reasoning engine shows this set of hazards as holding an entailment relationship.Figure 10.9. Now, it is up to the analyst to work with the results. To start with, the reason why hazards $h_{2.1}$ “Flight crew accepts a clearance correctly but inadvertently set it up incorrectly” and $h_{2.2}$ “Flight crew accepts and records a clearance correctly but does not follow it” have such a strong relationship is because they are causes of the same hazard. Therefore, any relationship can be dismissed for the analyst.

However, there is also an entailment relationship between hazards “Flight crew does not follow an ATC clearance” and “Flight crew accepts and records a clearance correctly but does not follow it”. In addition, there is also an entailment relationship between hazards “Flight crew does not follow an ATC clearance" and "Flight crew accepts a clearance correctly but inadvertently set it up incorrectly". In fact, hazard "Flight crew does not follow an ATC clearance" seems to be equivalent to both hazards combined together.

Based on these results, three actions can be made to improve the analysis:

1. I could confirm that both hazard $h_{2.1}$ and hazard $h_{1.2}$ have an entailment relationship and make this relationship explicit in the ontology and add the relationship $isIndirectCauseOf()$ respectively to the causes of the hazards.

2. I could also confirm that both hazard $h_{2.2}$ and hazard $h_{1.2}$ have an entailment relationship so make this relationship explicit in the ontology add the relationship $isIndirectCauseOf()$ respectively to the causes of the hazards.

3. Make step 1 and 2.

4. Replace "Flight crew does not follow an ATC clearance" by these two hazards.

False Positives The results have shown 11 cases of false positives results where results like synonymity or entailment are present in group of causes of same hazards. For example a possible synonymity relationship has been found in the following group of causes:

- *Pilot confuses criteria for requesting ITP*

- *Pilot request ITP not knowing that it does not meet criteria*

They are both causes of same hazard *Controller makes a misjudgement about innapropiate separation* and because they might share locations as well as the what could be affected by those causes would be the same processes or parts of the system. This could be avoided writing a plug-in to improve the results of the reasoner specifically for this kind of cases that are expected to happen in a PHA analysis. In addition, there are 3 more cases of false positives among the set of causes of the ontology.

Step 2: Perform an automated search of hazard paths

This step involves the identification of hazards paths.

Stage 5: Generation of State-Based Hazard Models from Ontology Rules This is the stage where a state-based model is constructed from the hazards and causes in the ontology. The search for hazard paths is done with the help of a model checker called MCMAS. The whole MCMAS model can be found in Appendix A.

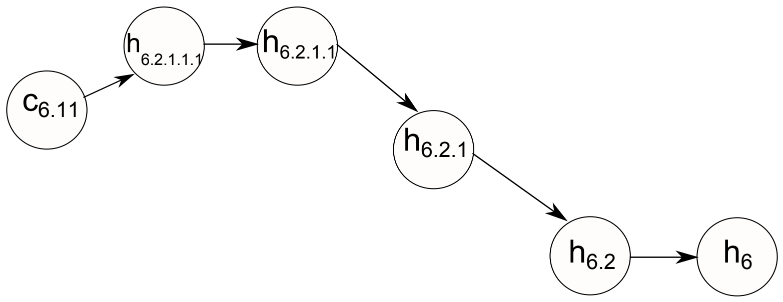


Figure 10.10: Hazard path that leads to hazard "Flight crew executes a cleared ITP which does not meet criteria"

Table 10.3: Hazard path that leads to hazard "Flight crew executes a cleared ITP which does not meet criteria"

ID	Hazard	is Cause of Hazard
H	Aircraft violates minimun separation	
h_6	Flight crew executes a cleared ITP which does not meet criteria	H
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	h_6
$h_{6.2.1}$	Controller makes a misjudgement about an inappropiate separation	$h_{6.2}$
$h_{6.2.1.1}$	Pilot request ITP not knowing that it does not meet criteria	$h_{6.2.1}$
$h_{6.2.1.1.1}$	Aicraft is giving wrong data about aircraft position with regards ITP	$h_{6.2.1.1}$
$c_{6.11}$	Malfunctions in communication device	$h_{6.2.1.1.1}$

Stage 6: Exploration of the state-based hazard model The PHA has been developed in a way that the hazards can be traced. Nevertheless, it is not immediately seen. It needs a search within the whole hazard model. In this stage we want to explore the state hazard model generated in order to generate hazard paths. This way, possible scenarios are identified about how hazards can develop.

Currently, we use the model checker MCMAS to generate these paths. For example, if we want to find a path to hazard "Flight crew executes a cleared ITP which does not meet criteria", the model checked finds the path shown in Figure 10.10. Table 10.3 describes the content of Figure 10.10: the first column in the table is the ID of the hazard or cause, the second column is the description of the hazard or cause, the third column tells if the hazard, or cause, described is cause of other hazard.

Table 10.4 shows hazard $h6$ with its structural and behavioural location described and the causes of these hazards with their structural and behavioural locations represented as well. All this information can be taken from the ontology representation.

Table 10.4 could be an initial step for the analyst who would want to continue a further analysis, the kind done in STPA with their scenarios, on certain hazards with regards their causes and where these causes are originated.

Table 10.4: Scenario where hazard and causes are shown

Hazard: Flight crew executes a cleared ITP which does not meet criteria			
Structural Location: Flight Subsystem Behavioural Location: Performing ITP			
ID	Causes	Process	Structure
$h_{6.1}$	ATC confirms a clerance without verification	ITP Approval Process	Communication Subsystem
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	Sending clearance	Communication Subsystem
$h_{6.3}$	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	Accepting clearance	Flight crew

Table 10.5: Scenario where a path of hazards and their locations are shown

Hazard: Flight crew executes a cleared ITP which does not meet criteria			
Structural Location: Flight Subsystem Behavioural Location: Performing ITP			
ID	Hazards in path	Process	Structure
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	Sending clearance	Communication subsystem
$h_{6.2.1}$	Controller makes a misjudgement about an inappropiate separation	Assessing request	ATC
$h_{6.2.1.1}$	Pilot request ITP not knowing that it does not meet criteria	Requesting maneuver	Flight Crew
$h_{6.2.1.1.1}$	Aicraft is giving wrong data about aircraft position with regards ITP	Verifying criteria	Aircraft
$c_{6.11}$	Malfunctions in communication device	Verifying criteria Aicraft	ITP equipment

Table 10.6: Scenario for hazard "Flight crew executes ITP procedure after request but without clearance"

ID H	Hazard Aircraft violates minimun separation	is Cause of Hazard
h_5	Flight crew executes ITP procedure after request but without clearance	H
$h_{5.1}$	Flight crew misses or incorrectly interprets a message from ATC (e.g. clearance denied)	h_5
$h_{5.2}$	Pilot is too confident in getting clerance from ATC	h_5
$h_{5.1.1}$	The ATC issues a complex transmission containing more than two instructions (e.g. speed, altitude and heading)	$h_{5.1}$
$c_{5.1}$	Communication not clear because of radio interfearance or blocked transmission	$h_{5.1}$
$c_{5.2}$	Inadequate English proficiency, or use of standard phraseology	$h_{5.1}$
$c_{5.3}$	Inadequate speed of transmission	$h_{5.1}$
$c_{5.4}$	ATC lacks discipline in communications	$h_{5.1.1}$
$c_{5.5}$	Lack of discipline and competence from Flight crew	$h_{5.2}$

Table 10.5 shows another kind of scenario, one where a path of hazards is shown (see Figure 10.11) and where one can also locate where these individual hazards are originated behaviourally and structurally.

The full description of the hazards can be found in Table 10.7, Table 10.6, Table 10.8

10.2.2 Discussion

The automated analysis was able to suggest dependencies that where not considered in the PHA. Scenarios were identified that may warrant further investigation. For example, Tables 10.7, 10.6 and 10.8 shows scenarios where the ATC performance and ability have shown to be poor. This could be for a number of reasons, e.g., increased working hours causing more fatigue, or lack of training due to a bad managerial team. Therefore, those multiple events that could lead to an accident might have same common roots on the way the ATC is managed. These initial scenarios can therefore be the basis for a deeper analysis of different root cause that could affect the way the system operates and lead to an accident.

Table 10.7: Scenario for hazard "Flight crew executes a cleared ITP which does not meet criteria"

ID	Hazard	is Cause of Hazard
H	Aircraft violates minimun separation	
h_6	Flight crew executes a cleared ITP which does not meet criteria	H
$h_{6.1}$	ATC confirms a clerance without verification	h_6
$h_{6.2}$	Controller unadvertedly provides a clearance where separation is inadequate	h_6
$h_{6.3}$	The pilot accepts a level clearance intended for another aircraft (call-sign confusion)	h_6
$h_{6.2.1}$	Controller makes a misjudgement about an inappropiate separation	$h_{6.2}$
$h_{6.2.2}$	ATC has not heard an erroneous feedback and has not corrected it	$h_{6.2}$
$h_{6.2.3}$	ATC does not correct a misreading of level or altitude from the pilot and approves it	$h_{6.2}$
$h_{6.2.1.1}$	Pilot request ITP not knowing that it does not meet criteria	$h_{6.2.1}$
$h_{6.2.1.2}$	Pilot confuses criteria for requesting ITP	$h_{6.2.1}$
$h_{6.2.1.1.1}$	Aicraft is giving wrong data about aircraft position with regards ITP	$h_{6.2.1.1}$
$c_{6.1}$	The pilot mishears the level clearance and he doesn't read back the clearance	$h_{6.1}$
$c_{6.2}$	Call signs coincidentally contain the same alphanumeric characters in a different order (e.g. AB1234 and BA 2314)	$h_{6.3}$
$c_{6.3}$	Airlines schedule flights with similar call signs to be in the same airspace at the same time.	$h_{6.3}$
$c_{6.4}$	Airlines allocate commercial flight numbers as call-signs; these are normally consecutive and therefore similar (e.g. RUSHAIR 1431, RUSHAIR 1432, etc.)	$h_{6.3}$
$c_{6.5}$	ATC uses read back time to carry out other tasks	$h_{6.2.2}$
$c_{6.6}$	Pilot reads back incorrect clearance within complez trasnmission	$h_{6.2.3}$
$c_{6.7}$	Tiredness	$h_{6.2.1.2}$
$c_{6.8}$	Lack of training	$h_{6.2.1.2}$
$c_{6.9}$	Misseting of aircraft equipment	$h_{6.2.1.1.1}$
$c_{6.10}$	Inattention to equipment malfunction	$h_{6.2.1.1.1}$
$c_{6.11}$	Malfunctions in communication device	$h_{6.2.1.1.1}$

Table 10.8: Scenario for hazard “Pilot aborts an ITP clearance when criteria is met”

ID	Hazard	is Cause of Hazard
H	Aircraft violates minimun separation	
h_7	Pilot aborts an ITP clearance when criteria is met	H
$h_{7.1}$	ATC sends abort request to wrong aircraft	h_7
$h_{7.2}$	ATC sends abort request to aircraft when it is not needed	h_7
$c_{7.1}$	ATC is interrupted or distracted	$h_{7.1}$
$c_{7.2}$	Call signs coincidentally contain the same alphanumeric character in a different order	$h_{7.1}$
$c_{7.3}$	ATC is disoriented due to conflicting information sources	$h_{7.2}$
$c_{7.4}$	ATC is not properly aware of the surrounding traffic	$h_{7.2}$

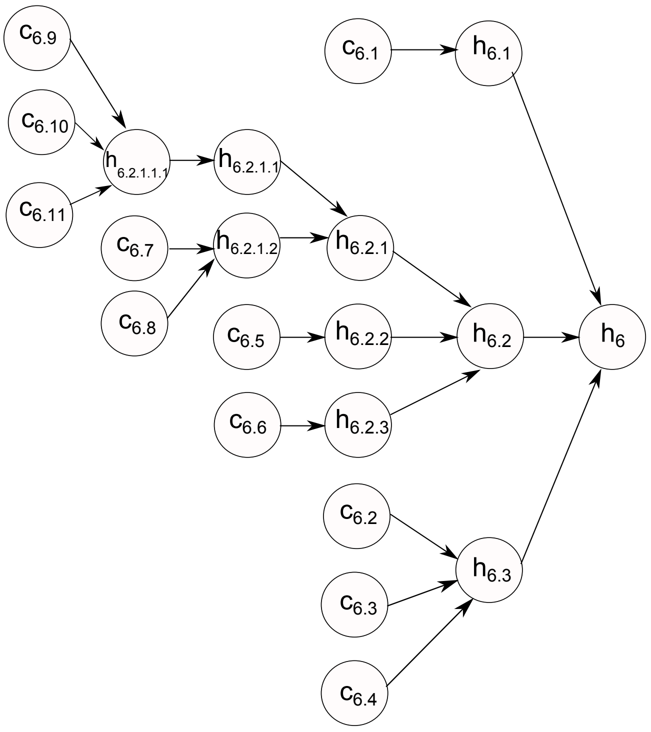


Figure 10.11: Scenario for hazard “Flight crew executes a cleared ITP which does not meet criteria”

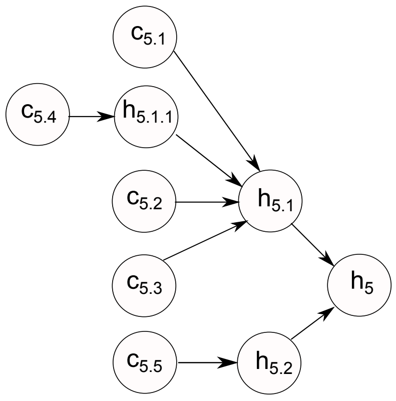


Figure 10.12: Scenario for hazard “Flight crew executes ITP procedure after request but without clearance”

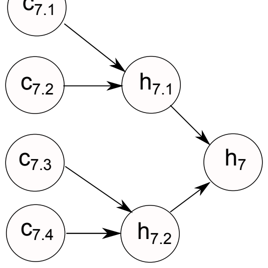


Figure 10.13: Scenario for hazard “Pilot aborts an ITP clearance when criteria is met”

11 Conclusion

Through the analysis of the problems with Hazard Analysis (HA) and through iterative application to case studies an approach has been develop to improve the results of HA. This thesis has presented a systematic method for assessing and improving the quality of hazard analysis results. The method is based on the use of an ontology that captures knowledge about the system, the hazards and causes and consequences of hazards. The method has been developed based on recent technologies and tools (Protégé, ontologies, reasoning engines, model checking, and model checking techniques such as MCMAS). The method is meant to be used by safety analysts, to help them perform a rigorous exploration of dependencies between hazards, and make sure all important relations are taken into account in the analysis. The method has been applied in a variety of situations, the way how it works has been shown and a number of issues have been raised. Another important result of this research is that a thorough validation of the proposed ontology has been done. The ontology has been instantiated in several real world situations, and it has been checked that the ontology is truly capturing the domain conceptualisation, this is a well grounded evaluation process [dAF14]. The method proposed has also shown, through the case studies presented, that a systematic and thorough documentation of the hazard analysis and its respective system documentation is produced. A measure to judge a hazard analysis is also documentation [RvdB02]. More specific findinds will be discussed in further sections of this chapter.

Future work involves, first of all, to develop a fully integrated tool that includes a module that takes as input the whole ontology and generates the state based hazard model with all the necessary transformations. In addition, this tool would need further developments towards the involvement and interworking of different stakeholders.

The reminder of this chapter presents final remarks on the developed analysis method and the obtained results.

11.1 Results

Below, different conclusions are drawn, some directly related to a stage of the method and finally, more general conclusions. The proposed method consist of the following steps and stages:

- **Step 1: Carry out a systematic exploration of dependencies**. In this step, information provided in hazards worksheets (description in natural of the system under analysis, identified hazards, their causes in system design, and potential consequences) are collected and systematically analysed. The step is decomposed in four sequential stages:
  - **Stage 1**: Collect information about the system under analysis.
  - **Stage 2**: Revise the hazard analysis with regards to the collected information.
  - **Stage 3**: Develop structural and behavioural representations of the system under analysis.
  - **Stage 4**: Search implicit and overlooked relationships.
- **Step 2: Perform an automated search of hazard paths**. This step disambiguates information in hazard analysis worksheets by constructing and analysing a state based model of hazard paths. The exploration of different relationships (paths) between hazards is carried out with the help of an automatic search engine. Overlooked hazard paths can be captured in the form of hazard scenarios highlighting common causes for different hazards, or hazards whose severity or frequency needs to be reclassified. The step can be decomposed in three sequential stages:
  - **Stage 5**: Generate a state-based hazard models from the ontology rules
  - **Stage 6**: Explore the state-based hazard model.
  - **Stage 7**: Visualize hazard paths.

The proposed method has been appplied to a variety of case studies from railway system, aviation, medical systems and voting systems which are very different in scope and maturity. The method has been shown to be feasible and it has found a number of issues in different stages. Below these findings will be expanded, either per stage or more general results.

Stage 1: Hazard Analysis and System Description This stage is just common practice in hazard analysis but by having it more explicitly, we will just contribute to a more organized way to document a hazard analysis so that when it needs to be consulted in the future, the hazard analysis can be traced back and understood in its completeness.

All case studies give direct reference to the documentation used to start the hazard analysis and the system description (see chapters 6, 7, 9, 8, 10). It has been shown to be useful when going back to the case study after it has been applied back in time.

Stage 2: Systematic revision of inputs System description involves different representations in a same study. A systematic revision of inputs helps with the comparison of different models so it is possible to check if they refer to same behaviour in same manner. This is to answer the question whether the way the hazard analysis has been developed can be totally traced back to the models in the system design and exactly which models. This stage aims to raise awareness of the possible various ways we are refering to part of the behaviour or structure in our system design and hazard analysis. For example, while analysing case study Prêt á Voter (see Chapter 8) on this stage, it was possible to discover some gaps in the description of the Voting and Post-Voting phases, a number of processes (tasks) that were given in the hazard analysis were not mentioned in the description of the behaviour of the system, all these documents belonged to the documentation of the HAZOP. Another example of the utility of this stage is found in the LRT System case study (see Chapter 9), where it was possible to highlight some hazards that were not given enough description and context and therefore were not considered as input in the ontology, this also happened in Prêt á Voter but with a reduced number of hazards, if the original authors of the hazards analysis were performing this stage of the method, they could have also improved those descriptions.

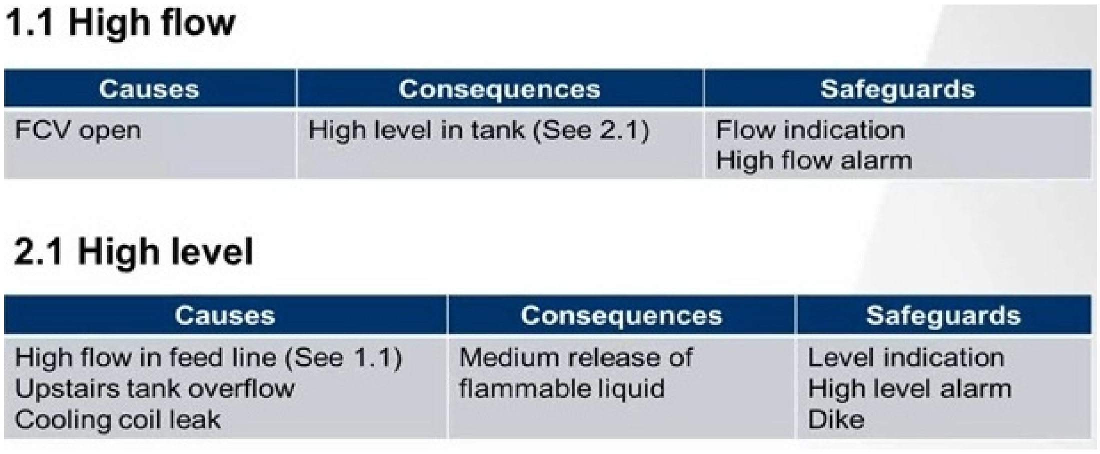
The findings in this stage are importatnt because this stage is previous to populating the ontology which is more significative because it shows how the method works even at the early stages.

Stage 3: Ontology guided revision of inputs This Stage aims to help the analyst with reviewing, but also improving, the current system representation and also the hazard analysis representation. In order to do so, the proposed method provides an additional representation, which can be part of the documentation of the system description. This representation is then used in the ontology in order to study hazards and its relationships. In doing so, some case studies experienced changes in the representation. Also, when no change has been applied to the system or hazard representation, performing this stage helps to understand better those representations.

In the case studies, previosly documented, representing the hazard analysis in the ontology helped to understand better what the hazards were refering to and reduce ambiguity. Additionally, hazards need to be specific, and this stage adds specificity to the hazard descriptions by means of revision of all hazards and cause and their locations and scopes. This stage also highlighted that the description of hazard causes were not rigorous enough and the descriptions we quite vague for example, "technical fault", "malicious".

One significant result of this stage has been the various instatiations of the foundational ontology. It has been instatiated in all case studies (chapters 6, 7, 9, 8, 10). This is an important result because it demonstrates that the ontology is truly capturing the domain conceptualisation, this is a well grounded evaluation process [dAF14] for ontologies.

This stage is very intense because it is the stage where the ontology is populated. In this stage, it is possible to return to Stage 1 which means there is a feedback that can help to improve the documentation of the hazard analysis, the representation of the system and the hazard analysis worksheets.



1.1 High flow

Causes	Consequences	Safeguards
FCV open	High level in tank (See 2.1)	Flow indication High flow alarm

2.1 High level

Causes	Consequences	Safeguards
High flow in feed line (See 1.1) Upstairs tank overflow Cooling coil leak	Medium release of flammable liquid	Level indication High level alarm Dike

Figure 11.1: HAZOP strategy: How to fill in HAZOP worksheets

Stage 4: Search for implicit and overlooked relationships We have stressed out that when using the ontology, it helps us with explicitly highlighting relationships that are known by the experts, analysts or any other stakeholders, even though it is not clearly stated in the written hazard. For example, the LRT Railway case study was developed over a Preliminary Hazard Analysis (PHA) of a less detailed system. Pret a Voter case study was done over the HAZOP study for a computer based system, where the hazard analysis has been done over the roles of the actors and systems. In both cases, we found different hazards had non-direct relationships (in some cases synonyms). This has helped us to highlight the contribution of a computer-aided review of hazard analysis. In addition, we showed how ontologies can be used to reason about these dependencies.

Stage 5: Generation of the State Based Hazard Model The various hazard analysis techniques currently in use have their own recommended strategies in order to find a way to enter the information and follow it. One must have a strategy and one must maintain the strategy [Lor13]. Figure 11.1 was taken from the HAZOP webinar [Lor13]. It shows how to link causes and consequences between different deviations (hazards) in a HAZOP study. This recommended strategy is to explicitly link a hazard as a cause or consequence to another hazard if this is the case; this strategy is also followed in another Hazard Identification techniques like Preliminar Hazard Analysis. For example, in the Railway case study, the PHA authors explicitly relate the hazard *A-1* `LRV fire` as a causes for the hazard *A-3* `Fire/smoke in tunnel`.

Nevertheless, sometimes this strategy is not followed thoroughly. Our methods aims, on the one hand, to help the analyst to make these links between hazards and causes (or consequences) more explicit. On the other hand, it focuses on discovering more of these relationships that might have been overlooked. This contributes to the creation of a hazard model: hazards and the paths they take. This Hazard model is called a state based hazard model.

Stage 6: Exploration of the State Based Hazard Model When this stage has been applied on the presented case studies, different hazard paths (scenarios) have been found. This improves the Hazard Analysis process because creates scenarios otherwise overseen by the safety analysts. Hazard paths have been found for the Medication Prescription System case study, Generic Infusion Pump User Interface (UI) Case Study, the LRT Railway Case Study and the ITP Case Study.

Stage 7: Visualizing Hazard Paths When hazard paths have been found then a visualization of the hazard paths contributes to expose more clearly scenarios where hazard develop and this way improve the results of the hazard analysis. In all the case studies mentioned in Stage 6 the visualition of the hazards paths has also been done.

11.2 Limitation of the research

The proposed method is not a new hazard analysis technique but a systematic approach that helps to refine the results of hazard analysis techniques. In addition, improves the documentation of the hazard analysis and the system under study.

The proposed method does not claim to immediately and automatically increase the amount of hazards that have been found in the applied hazard analysis, which needs to be done since it is an input for the method. However, its does provide a systematic approach in which the hazard analysis is revised and refined, and where a graphical and an ontological representation of the system, for the only purpose of the hazard analysis, are obtained. During this systematic process, a better understanding, and documentation, of the hazard analysis and the system under study might lead to the discovery of more hazards and dependencies. Nevertheless, it has also been discussed that no current hazard analysis technique has claimed to always find more hazards than any other particular hazard analysis technique. STPA, for example, claims to provide a different approach to how hazards are found that is more akin to current highly software intense systems. The application of the their case studies then shows how better the hazard analysis is in practice, sometimes these case studies are compared to already applied hazard analysis. Another claim of the STPA is that the way the analysis is presented to stakeholders provides a better picture of the hazards that are being studied therefore improves the final results of the hazards analisys.

In addition, this method helps to add specificity to the hazards descriptions and also add more detail to the system being designed. The better the system is designed and the hazards described the better the results. For example, if the system is extremely trivial in design the more false positives would be found. Moreover, it is not a natural language processing system so, for example, if the statement "Snow on track" is described twice in the hazard analysis as two different hazards, but the analyst would give the wrong associations in the ontology, for example the wrong processes or structural locations then these two identical written hazards will not be found as synonyms by the reasoning engine.

11.2.1 Human Factors

The consideration of human factors in hazard analysis and the way hazards are sometimes presented in simplified ways as *human error* without analysing underlying causes, as has been discussed in different studies, for example [VVSvdSThE97, RDS+06, LAWV04]; has not been directly tackled in this research.

Many hazards analysis ignore *human error* or consider it in overly simplistic terms. Which is insuficient as humans operate in complex ways, and resulting accident reflect that complexity. For example, when analysis a healthcare system, long hours, fatigue, the stress level of healthcare workers are not analysed more deeply. Another example would be the way people interact with technology and errors happens, the causes should be studies well beyond *human error*. Many times accidents do not occur because of one single human error but rather because of a series of small, seemingly insignificant errors that, when added together, result in catastrophic consequences.

In [KS08], the authors argue that a systematic approach to analyse the situation that may lead to human error is required. The approach should be able to predict the conditions that support the occurrence of error. This means the attention is shifting from the error itself and focusing at the factors that support the occurrence of the error. A systematic approach to look for the causes that lead to a particular error is needed. Much work needs to be focused in this area. It could be argued that tackling how the hazards are described and how the causes of those hazards are presented could be a starting point. In this regard, Chapter 7 refers to the Generic Infusion Pump User Interface (UI) Case Study and shows a hazard analysis that aims to deal with underlying causes to human error because it is tackling design error in HCI (Human Computer Interaction), this way design issues that might induce human errors are found and prevented. Subsequently, The implementation of an ontology based method that could find out, by the use of a reasoning engine, rules of inference and relationships, the possible causes to an error could contribute to this need. In addition, another contribution could be the generation of models where the development of hazard paths could show scenarios that lead to a particular error.

Moreover, when the hazard analysis do tackle *human error* in its complexity, the proposed ontology could, for example, by the use of the reasoning engine and additional rules of inference, find common cause to various hazards which might contribute to finding underlying causes to *human error*.

Finally, future work should also focus directly in tackling this area which is quite relevant and also current for hazard analysis.

11.3 Limitations of the chosen evaluation framework

The evaluation framework has been useful in validation the ontology and the various steps of the proposed method. However, now that the current method has been initially evaluated and findings have been shown, a new approach to evaluate the method has to be applied. The evaluation framework needs to enter a new stage where another party, and not only the author of the thesis, applies the method and provides feedback with regards the different stages of the method, and ultimately methods to measure effectiveness, such as the ones discusses in Section 2.7, should be applied as well.

11.4 Practical implications and challenges that might arise when adopting this approach by new parties

Even though, the hazard analyses presented in Chapter 9 and in Chapter 8 have not been performed by the author of this thesis, and those hazard analysis were not performed to contribute to the research carried out by the author of this thesis; the method was not applied by outside parties. Various challenges might arise when the method might be applied by an expert safety analyst:

- First of all, the need to use a graphical notation and representation, structural and behavioural, for the system might be seen as adding more labor to their already demanding job. Nevertheless, hazard analyses like STPA do require that as well.

- Stage 3 of the proposed method is intense and laborious. The analyst might need a convincing reason to be persuaded to perform it. However, from the experience of applying this stage in all the case studies, it is possible to say that this stage does produce a lot of information that contributes to improve the representation of the system under study as well as the hazards analysis. Moreover, it does contribute to better understand the system that is being analysed.

- Currently there is not an appropriate user interface to help to populate the ontology since Protégé is being used for this purpose. Understanding and using Protégé will add more work to the safety analyst. A Graphical User Interface(GUI) is, therefore, an important aspect that needs to be covered. A GUI will also be important in order to help the analyst to use the reasoning engine for the purpose of extracting particular knowledge from the ontology.

11.5 Future Directions

The method proposed is well with trend to more model based developments as well as supports communication an understanding between multi stakeholder. This is why, more case studies where, for example, two different systems have been analysed separately and two hazard analysis has been integrated into one should be done with this proposed method in order to understand the capabilities as well as future adjustments in this direction. There is also potential to expand and enhance the tool so various manual stages of hazard analysis are supported.. Finally, Using Protégé for a safety analyst might add more work to their tasks. The idea is that there should be a front-end tool that interacts with the analyst and with the ontology. This way the input of data will be more user-friendly to the safety analyst and the results could also be presented in a more understandable way to the user.

Additionally, human factors and studying how the proposed method is applied in a real case study from starting from scratch are quite decisive research topics that are important for the feedback and assessment of the method.

11.5.1 Discussion

Various different systems have been portrayed in the proposed notations for the system and the proposed graphical and their respective ontological representations have shown to be versatile. The graphical representations of the system were created together with the proposed method and they are a contribution to the documentation of the hazard analysis.

The method has shown to provide a systematic and comprehensive revision of the hazard analysis. This way, adding precision to the descriptions because each hazard, cause and consequence needs to explicitly localise where it occurs in the system with regards the structural and behavioural representation. Various studies asserted that the most significant flaws in hazard analysis techniques applied at the early stages of system design are often related to the omission of hazards and hazard causes [Har10]. The proposed method contributes to reduce such omissions by:

- It contributes to make hazards more specific.
- It contributes to get more understanding of the hazards.
- It helps the analyst not only with reviewing but also improving the current system representation and also the hazard analysis representation.

The generation of hazard paths, that ultimately are scenarios, contributes to further analysis of hazards. Because of the nature of the tasks, which are time consuming, it is not always possible to go into in-depth analysis of each particular hazards with regards tolocal parts of the system. This method contributes to it. In addition, possible scenarios can be generated and this contributes to improve the result of the hazard analysis.

The ontology representation where the knowledge about the system and hazard analysis is saved allows for the knowledge to be extracted and represented in various tabular forms. This is, various different extractions of the ontology can be rehearsed with the use of rules and the ontology reasoner, for example, a common cause to various hazard if it does exist. Contributing to the refinement of the hazard analysis.

The various tables (Table 10.3, Table 10.4, Table 10.5, Table 10.7, Table 10.8) shown in chapter 10 show that the information regarding the hazard analysis stored in the ontology can be presented in various tabular form. Some of these tables follow a similar way the that ones presented in STPA. This is to show how ontologies can help to hazard analysis techniques to be more versatile with the information they are presenting.

Finally, it is difficult to analyse, interpret or, even, organise the large amount of collected information during analysis sessions and other sources. It is sometimes difficult to follow a defined strategy in order to write the hazard analysis and hazard analysis worksheets. This method contributes to tackle this problem by means of acting as an organised and linked repository, in addition to the knowledge base. This is particularly important because studies such as [RvdB02] state that a measure to judge hazard analysis is also documentation.

Bibliography

[ALRL04] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dependable Secur. Comput.*, 1(1):11–33, January 2004.

[Arn] Arney, David and Jetley, Raoul and Jones, Paul and Lee, Insup and Ray, Arnab and Sokolsky, Oleg and Zhang, Yi. The Generic Infusion Pump Project, The Generic Patient Controlled Analgesia Pump Hazard Analysis. `http://rtg.cis.upenn.edu/gip.php3`.

[AS10] R. Akerkar and P. Sajja. *Knowledge-Based Systems*. Jones & Bartlett Learning, 2010.

[Aut14] Civil Aviation Authority. Introduction to bow tie. `http://www.caa.co.uk/default.aspx?catid=2786&pagetype=90`, October 2014.

[Bar02] Khaled; Zubrow Dave Barnard, Julie; El Emam. Using capture-recapture models for the reinspection decision. *International Conference on Software Quality*, 12(0):1–13, October 2002.

[BCM$^+$90] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Inform. and Comput.*, 98/2, 1990.

[bs607] *BS EN 61025:2007 Fault tree analysis (FTA)*. BSI, 2007.

[bs610] *BS EN 61508-4:2010 Functional safety of electrical/electronic/ programmable electronic safety related systems. Definitions and abbreviations*. BSI, 2010.

[bs616] *BS EN 61882:2016 Hazard and operability studies (HAZOP studies). Application guide*. BSI, 2016.

[bs618] *BS EN IEC 60812:2018 Failure modes and effects analysis (FMEA and FMECA)*. BSI, 2018.

[CC09] Susan Cantrell and Pat Clemens. Finding all the hazards how do we know we are done? *Professional Safety, American Society of Safety Engineers*, 54(11), November 2009.

[Cen10] Center for Devices and Radiological Health, US FDA. Total produc life cycle: infusion pump – premarket notification [510(k)] submissions, April 2010.

[CGL94] E. M. Clarke, O. Grumberg, and D. E. Long. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems*, 16(5):1512–1542, 1994.

[CGP99] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, Cambridge, Massachusetts, 1999.

[CGp00] Oscar Corcho and Asuncin Gmez-prez. A roadmap to ontology specification languages. pages 80–96. Springer-Verlag, 2000.

[CJB99] Balakrishnan Chandrasekaran, John R Josephson, and V Richard Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems and their applications*, 14(1):20–26, 1999.

[COP] Stanford University CO-ODE Project. Protege-owl.

[Cor] Josep Corbella. 'no debemos permitir que las maquinas decidan por nosotros'.

[dAF14] Ricardo de Almeida Falbo. Sabio: Systematic approach for building ontologies. In *Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering co-located with 8th International Conference on Formal Ontology in Information Systems, ONTO.COM/ODISE@FOIS 2014, Rio de Janeiro, Brazil, September 21, 2014.*, 2014.

[Dev02] Vladan Devedzić. Understanding ontological engineering. *Commun. ACM*, 45(4):136–144, April 2002.

[Dev14] Across Safety Development. Bow tie resources. `http://www.acrosssafety.com/`, October 2014.

[DG17] A. De Grey. *The Next Step: Exponential Life*. Turner, Ediciones S.A., 2017.

[dM] Ramon Lopez de Mantaras. Marvin minsky: 'tenemos un monton de expertos estupidos'.

[DSSO11] O. Daramola, T. Stralhane, G. Sindre, and I. Omoronyia. Enabling hazard identification from requirements and reuse-oriented hazop analysis. In *Managing Requirements Knowledge (MARK), 2011 Fourth International Workshop on*, 2011.

[EA06] David Evans and Inc. Associates. West corridor lrt project final engineering design phase preliminary hazard analysis draft revision 0. `http://www.rtd-fastracks.com/media/uploads/wc/WC_Risk_Assessment_Draft_06-06.pdf`, June 2006.

[EKM$^+$07] V. Ermagan, I. Krueger, M. Menarini, J.-i. Mizutani, K. Oguchi, and D. Weir. Towards model-based failure-management for automotive software. In *Software Engineering for Automotive Systems, 2007. ICSE Workshops SEAS '07. Fourth International Workshop on*, pages 8–8, 2007.

[Eng62] D. C. Engelbart. Augmenting Human Intellect: A Conceptual Framework. Air Force Office of Scientific Research, AFOSR-3233, `www.bootstrap.org/augdocs/friedewald030402/augmentinghumanintellect/ahi62index.html`, 1962.

[Eri05] Clifton A. Ericson. *Hazard analysis Techniques for System Safety.* Wiley, Hoboken, NJ :, 2005.

[FA] Food and Drug Administration. Fda, year =.

[FC12] C.H. Fleming and Langley Research Center. *Safety assurance in NextGen.* NASA contractor report. National Aeronautics and Space Administration, Langley Research Center, 2012.

[Fel90] Christiane Fellbaum. English verbs as a semantic net. *International Journal of Lexicography*, 3(4):278–301, 1990.

[FRTDoDRD14] Colorado FasTracks Regional Transportation District of Denver (RTD) Denver. West rail line home. `http://www.rtd-fastracks.com/wc_1`, 2014.

[fSMPTC07a] Institute for Safe Medication Practices Toronto Canada. Fluorouracil incident root cause analysis. `http://www.ismp-canada.org/download/reports/FluorouracilIncidentMay2007.pdf`, May 2007.

[fSMPTC07b] Institute for Safe Medication Practices Toronto Canada. Fluorouracil incident root cause analysis: Follow-up. `http://www.ismp-canada.org/download/reports/FluorouracilIncidentMay2007.pdf`, May 2007.

[GHM+08] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. Owl 2: The next step for owl. *Journal of Web Semantics*, 6(4):309 – 322, 2008. Semantic Web Challenge 2006/2007.

[GMF+03] John H Gennari, Mark A Musen, Ray W Fergerson, William E Grosso, Monica Crubzy, Henrik Eriksson, Natalya F Noy, and Samson W Tu. The evolution of protg: an environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, 58(1):89 – 123, 2003.

[Gru95] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, 43(56), 1995.

[Har10] T.L. Hardy. Using accident reports to improve the hazard identification process. *28th International System Safety Conference*, September 2010.

[Her] HermiT reasoner home page. `http://hermit-reasoner.com/`. (accessed: 02/10/2019).

[HM03] V. Haarslev and R. Mueller. Racer: An OWL reasoning agent for the Semantic Web. In *Proceedings of the International Workshop on Applications, Products and Services of Web-based Support Systems at the IEEE International Conference on Web Intelligence*, pages 91–95, Halifax, Canada, 2003.

[HR04] M. Huth and M. Ryan. *Logic in Computer Science.* Cambridge University Press, 2004.

[Int00] International Organization for Standardization. *ISO 14971: medical devices - application of risk management to medical devices.* ISO, 2000.

[ISO] ISO Technical Management Board.

[JGOD02] Tixier J., Dusserre G., Salvi O., and Gaston D. Review of 62 risk analysis methodologies of industrial plants. *Journal of Loss Prevention in the Process Industries*, 15(4):291–303, 2002.

[JJITJW02] Paul L Jones, Joseph Jorgens III, Alford R Taylor Jr, and Markus Weber. Risk management in the design of medical device software systems. *Biomedical instrumentation & technology*, 36(4):237–266, 2002.

[Kle01] T.A. Kletz. *Hazop and Hazan: Identifying and Assessing Process Industry Hazards.* The Institution of Chemical Engineers, 2001.

[KRW12] Jens Kolb, Manfred Reichert, and Barbara Weber. Using concurrent task trees for stakeholder-centered modeling and visualization of business processes. In *S-BPM ONE 2012*, number 284 in CCIS, pages 237–251. Springer, April 2012.

[KS08] PMW Körvers and PJM Sonnemans. Accidents: a discrepancy between indicators and facts! *Safety Science*, 46(7):1067–1077, 2008.

[Lad05] Peter B Ladkin. Ontological analysis. 2005.

[Lad10] Peter Bernard Ladkin. Ontological hazard analysis of a communication bus system. July 2010.

[LAWV04] Melinda Lyons, Sally Adams, Maria Woloshynowych, and Charles Vincent. Human reliability analysis in healthcare : A review of techniques. *International Journal*, 16(4):223–237, 2004.

[LdM15] Ramon Lopez de Mantaras. Algunas reflexiones sobre el presente y futuro de la inteligencia artificial. 2015.

[Lea10] Kieran Leach. *A Demonstration of the Applicability of HAZOP to Voting Systems*. New Castle University, 2010.

[Lea11] Kieran Andrew Leach. *A Demonstration of the Applicability of HAZOP to Voting Systems*. University of Newcastle upon Tyne, 2011.

[LEE01] BURTON H. LEE. Using fmea models and ontologies to build diagnostic models. *AI EDAM*, 15:281–293, 8 2001.

[Lev11] Nancy G Leveson. *Engineering a safer world: Systems thinking applied to safety*. MIT Press, 2011.

[Lev13] Thomas John Leveson, Nancy G. An stpa primer, 2013.

[LG10] I.A. Letia and A. Groza. Developing hazard ontology for supporting haccp systems in food supply chains. In *Intelligent Systems and Informatics (SISY), 2010 8th International Symposium on*, pages 57–62, 2010.

[Lor13] Don Lorenzo. Hazop: Tactics to streamline the review process webinar. `https://www.youtube.com/watch?v=o2x0i_fLby0`, September 2013.

[LQR09] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: a model checker for the verification of multi-agent systems. In *Proceedings of CAV '09*, 2009.

[LWW10] Gang Liu, Yanni Wang, and Chonglong Wu. Research and application of geological hazard domain ontology. In *Geoinformatics, 2010 18th International Conference on*, pages 1–6, 2010.

[Min92] Marvin Minsky. Future of ai technology. *Toshiba Review*, 47(7), 1992.

[MPB$^+$08] R Mayer, C. Plank, A. Bohner, S. Kollarits, A. Corsini, F Ronchetti, H. Siegel, L. Noessing, V. Mair, U. Sulzenbacher, D. Tosoni, S. Cimarosto, A. Zanco, S. Todorov, L. Krastev, N. Wergles, W. Gasperl, M. Mayerl, T. Toli, H. Haradalia, N. Koutsias, S. Kreuzer, C. Liehr, C. Rachoy, J. Papez, and P. Jindra. Monitor: Hazard monitoring for risk assessment and risk communication. *Georisk: Assessment and Management of Risk for Engineered Systems and Geohazards*, 2(4):195–222, 2008.

[oD07] Ministry of Defence. *Defence Standard 00-56, Safety Management Requirements for Defence Systems.* 2007.

[oDotUSoA12] Department of Defense of the United States of America. *Deparment of Defense Standard Practice, System Safety.* 2012.

[oL20] United States Department of Labor. Occupational safety and health administration. `https://www.osha.gov/`, 2020.

[PL03] W. Penczek and A. Lomuscio. Verifying epistemic properties of multi-agent systems via bounded model checking. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, pages 209–216, 2003.

[Pol62] M. Polanyi. *Personal Knowledge: Towards a Post-critical Philosophy.* Routledge and Kegan Paul, 1962.

[PRCF06] Sofia Guerra P R Caseley and Peter Froome. Measuring hazard identification. *1st IET International Conference on System Safety*, pages 23–28, June 2006.

[PWB$^+$] Philippe Palanque, Marco Winckler, Regina Bernhaupt, Eugenio Alberdi, Lorenzo Strigini, and Peter Ryan. Arove-v: Assessing the resilience of open verifiable e-voting systems.

[Rai01a] Railtrack. *Engineering Safety Management*, volume 3. Railtrack, 2001.

[Rai01b] Railtrack. *Engineering Safety Management*, volume 1 and 2. Railtrack, 2001.

[RBH$^+$09] Peter YA Ryan, David Bismark, James A Heather, Steve A Schneider, and Zhe Xia. The prêt à voter verifiable election system. *IEEE transactions on information forensics and security*, 4(4):662–673, 2009.

[RDS$^+$06] Jim Ramsay, Frank Denny, Kara Szirotnyak, Jonathan Thomas, Elizabeth Corneliuson, and Kim L Paxton. Identifying nursing hazards in the emergency department: A new approach to nursing job hazard analysis. *Journal of safety research*, 37(1):63–74, 2006.

[rt11] railway technology.com. The west corridor light rail transit, united states of america. `http://www.railway-technology.com/projects/the-west-corridor-light-rail-transit/`, September 2011.

[RvdB02] J.L Rouvroye and E.G van den Bliek. Comparing safety analysis techniques. *Reliability Engineering and System Safety*, 75(3):289 – 294, 2002.

[Saf10] C.C.P. Safety. *Guidelines for Preventing Human Error in Process Safety*. Wiley, 2010.

[SC-08] RTCA (Firm). SC-186. *Safety Performance and Interoperability Requirements Document for the In-trail Procedure in Oceanic Airspace (ATSA-ITP) Application*. RTCA, Incorporated, 2008.

[SCfBIR15] Stanford University School of Medicine Stanford Center for Biomedical Informatics Research. Protege 5.0 beta version. `http://protege.stanford.edu/`, 2015.

[SDM$^+$09] Joseph Sussman, Rebecca S. Dodder, Joshua B. McConnel, Ali Mostashari, and Sgouris Sgouridis. The clios process a user's guide. `https://esd.mit.edu/Faculty_Pages/sussman/CLIOS-PROCESS.pdf`, Feb 2009.

[SK89] J. Suokas and R. Kakko. On the problems and future of safety and risk analysis. *Journal of Hazardous Materials*, 21(2):105 – 124, 1989.

[Som12] F. Somenzi. *CUDD: CU Decision Diagram Package Release 2.5.0*, 2012. `http://vlsi.colorado.edu/~fabio/CUDD/`.

[SOR10] T. Stralhane, I. Omoronyia, and F. Reicenbach. Ontology-guided requirements and safety analysis. In *6th International Conference on Safety of Industrial Automated Systems*, 2010.

[SPG$^+$07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *Web Semant.*, 5(2):5153, June 2007.

[ST99] William Swartout and Austin Tate. Guest editors' introduction: Ontologies. *IEEE Intelligent Systems*, 14(1):18–19, January 1999.

[Ste12] R.A. Stephans. *System Safety for the 21st Century: The Updated and Revised Edition of System Safety 2000*. Wiley, 2012.

[Ter17] Teresa Guerrero. Lopez de Mantaras: 'Una maquina puede ser creativa e ingeniosa pero no un genio', June 2017.

[Tra14] Transport Canada. Rail transportation. http://www.tc.gc.ca/eng/rail-menu.htm, September 2014.

[Uni] Stardog Union. Pellet: An open source owl dl reasoner for java.

[Uni13] Stanford University. Wordnet, a lexical database for english. http://wordnet.princeton.edu/, April 2013.

[Vin14] J.W. Vincoli. *Basic Guide to System Safety.* Wiley, 2014.

[VRMS99] Andre Valente, Thomas Russ, Robert MacGregor, and William Swartout. Building and (re)using an ontology of air campaign planning. *IEEE Intelligent Systems*, 14(1):27–36, January 1999.

[VVSvdSThE97] W Van Vuuren, CE Shea, Tjerk W van der Schaaf, and Pays-Bas). Technische hogeschool (Eindhoven. *The development of an incident analysis tool for the medical field.* Eindhoven University of Technology, Faculty of Technology Management, 1997.

[WB] Han-Hsiang Wang and Frank Boukamp. *Ontology-Based Job Hazard Analysis Support*, chapter 66, pages 676–685.

[WB11] H. Wang and F. Boukamp. Ontology-based representation and reasoning framework for supporting job hazard analysis. *Journal of Computing in Civil Engineering*, 25(6):442–456, 2011.

[Win99] Patrick H Winston. Why i am optimistic. *Invited Speaker, AAAI-99*, 1999.

[WJG01] Rune Winther, Ole-Arnt Johnsen, and BjornAxel Gran. Security assessments of safety critical systems using hazops. In *Computer Safety, Reliability and Security*, volume 2187 of *Lecture Notes in Computer Science*, pages 14–24. 2001.

[WLAHR$^+$09] PeterC. Wierenga, Loraine Lie-A-Huen, SophiaE. Rooij, NiekS. Klazinga, Henk-Jan Guchelaar, and SusanneM. Smorenburg. Application of the bow-tie model in medication safety risk analysis. *Drug Safety*, 32(8):663–673, 2009.

[WSM$^+$15] Louise V Wain, Nick Shrine, Suzanne Miller, Victoria E Jackson, Ioanna Ntalla, María Soler Artigas, Charlotte K Billington, Abdul Kader Kheirallah, Richard Allen, James P Cook, et al. Novel insights into the genetics of smoking behaviour, lung function, and chronic obstructive pulmonary disease (uk bileve): a genetic association study in uk biobank. *The Lancet Respiratory Medicine*, 3(10):769–781, 2015.

[WSWS01] B. J. Wielinga, A. Th. Schreiber, J. Wielemaker, and J. A. C. Sandberg. From thesaurus to ontology. In *Proceedings of the 1st international conference on Knowledge capture*, K-CAP '01, pages 194–201, New York, NY, USA, 2001. ACM.

[www14a] www.allenrailroad.com. Railroad glossary and definitions. `http://www.allenrailroad.com/consulting/Railroad_Glossary.htm`, September 2014.

[www14b] www.railsigns.uk. Rail signs and signals of great britain. `http://www.railsigns.uk`, September 2014.

[www14c] www.railway-technical.com. Railway technical web pages. `http://www.railway-technical.com`, September 2014.

[www14d] www.trafficsigns.us. Manual of traffic signs. `http://www.trafficsign.us/railsign.html`, September 2014.

[www19] www.skybrary.aero. Skybrary. `https://www.skybrary.aero/index.php/Main_Page#operational-issues`, July 2019.

[YGHS12] Xiquan Yang, Rui Gao, Zhengfu Han, and Xin Sui. Ontology-based hazard information extraction from chinese food complaint documents. In Ying Tan, Yuhui Shi, and Zhen Ji, editors, *Advances in Swarm Intelligence*, volume 7332 of *Lecture Notes in Computer Science*, pages 155–163. Springer Berlin Heidelberg, 2012.

[ZJJ10] Yi Zhang, Paul L Jones, and Raoul Jetley. A hazard analysis for a generic insulin infusion pump. *Journal of diabetes science and technology*, 4(2):263, 2010.

[ZMJT19] Yi Zhang, Paolo Masci, Paul Jones, and Harold Thimbleby. User interface software errors in medical devices: Study of us recall data. *Biomedical instrumentation & technology*, 53(3):182–194, 2019.

A Appendix A

Listing A.1: PescriptionHazardPaths.ispl

```
Semantics=MultiAssignment;

Agent Environment
  Vars:   -- Declaration of the states

    h1 : boolean; --h1 is a hazard
    h2 : boolean; --h2 is a hazard
    h3 : boolean; --h3 is a hazard
    h4 : boolean; --h4 is a hazard
    h5 : boolean; --h5 is a hazard
    h6 : boolean; --h6 is a hazard
    h7 : boolean; --h7 is a hazard
    h8 : boolean; --h8 is a hazard
  end Vars


  Actions = { none }; -- Declaration of actions
                       --none means no action or nothing happens.
  Protocol:
    Other : { none};
  end Protocol
  Evolution:
    h1 = true if h2 = true ;                  -- RULE 1
    h2 = true if h3 = true ;                  -- RULE 2
    h3 = true if h4 = true ;                  -- RULE 3
    h4 = true if h5 = true ;                  -- RULE 4
    h6 = true if h7 = true ;                  -- RULE 5
    h6 = true if h8 = true ;                  -- RULE 6
  end Evolution
end Agent

Agent Initiator   -- This agent effects the Initiating mechanism
  Vars:
    -- we need variables for every agent, ignore it
    name : boolean;
  end Vars
  -- Declaring the actions
  Actions = {I1, I2, I3, I4, I5, I6,I7, none};
  Protocol:
    -- They all can happen in every state but nothing happens
    -- if there is no transition
    Other : {I1, I2, I3, I4, I5, I6,I7, none};
  end Protocol
  Evolution:
    name = true if name = true;       -- syntax
  end Evolution
end Agent

Evaluation
    hazard1 if Environment.h1 = true;  -- atomic proposition
    hazard2 if Environment.h6 = true;  -- atomic proposition

end Evaluation

InitStates
  -- Defining initial state, where causes should be true
  -- and hazards false
  Environment.h1 = false and Environment.h2 = false and
  Environment.h3 = false and Environment.h4 = false and
  Environment.h5 = true  and Environment.h6 = false and
  Environment.h7 = true  and Environment.h8 = true  and
  Initiator.name = true;

end InitStates



Formulae
  EF hazard1; -- CTL formula
  EF hazard2; -- CTL formula
end Formulae
```

Listing A.2: DataEntryHazardPaths.ispl

```
Semantics=MultiAssignment;

Agent Environment
  Vars:   -- Declaration of the states

    h1 : boolean; --h1 is a hazard
    h2 : boolean; --h2 is a hazard
    h3 : boolean; --h3 is a hazard
    h4 : boolean; --h4 is a hazard
    h5 : boolean; --h5 is a hazard
    h6 : boolean; --h6 is a hazard
    h7 : boolean; --h7 is a hazard
    h8 : boolean; --h8 is a hazard
    h9 : boolean; --h9 is a hazard
    h10 : boolean; --h10 is a hazard
    h11 : boolean; --h11 is a hazard
    h12 : boolean; --h12 is a hazard
    h13 : boolean; --h13 is a hazard
    h14 : boolean; --h14 is a hazard
    h15 : boolean; --h15 is a hazard
    h16 : boolean; --h16 is a hazard
    h17 : boolean; --h17 is a hazard
    h18 : boolean; --h18 is a hazard
    h19 : boolean; --h19 is a hazard
    h20 : boolean; --h20 is a hazard
    h21 : boolean; --h21 is a hazard
    h22 : boolean; --h22 is a hazard
    h23 : boolean; --h23 is a hazards
  end Vars


  Actions = { none }; -- Declaration of actions
                        --none means no action or nothing happens.
  Protocol:
    Other : { none};
  end Protocol
  Evolution:
    h1 = true if h4  = true ;
    h1 = true if h5  = true ;
    h2 = true if h4  = true ;
    h2 = true if h6  = true ;
    h3 = true if h16 = true ;
    h3 = true if h17 = true ;
    h4  = true if h10  = true ;
    h5  = true if h7   = true ;
    h5  = true if h8   = true ;
    h6  = true if h9   = true ;
    h6  = true if h17  = true ;
    h6  = true if h18  = true ;
    h7  = true if h12  = true ;
    h8  = true if h11  = true ;
    h10 = true if h15  = true ;
    h11 = true if h14  = true ;
    h12 = true if h13  = true ;
    h13  = true if h21   = true ;
    h14  = true if h21   = true ;
    h15  = true if h21   = true ;
    h16  = true if h21   = true ;
    h17  = true if h21   = true ;
    h18  = true if h19   = true ;
    h18  = true if h20   = true ;
    h22  = true if h3    = true ;
    h23  = true if h22   = true ;
  end Evolution
end Agent

Agent Initiator   -- This agent effects the Initial
  Vars:
    -- we need variables for every agent, ignore it
    name : boolean;
  end Vars
  -- Declaring the actions
  Actions = {I1, I2, I3, I4, I5, I6,I7, none};
  Protocol:
    -- They all can happen in every state but nothing happens
    -- if there is no transition
    Other : {I1, I2, I3, I4, I5, I6,I7, none};
  end Protocol
  Evolution:
    name = true if name = true;        -- syntax
  end Evolution
end Agent

Evaluation
    hazard1 if Environment.h1  = true;  -- atomic proposition
    hazard2 if Environment.h2  = true;  -- atomic proposition
    hazard3 if Environment.h23 = true;

end Evaluation

InitStates  -- Defining initial state:

  Environment.h1 = false and Environment.h2 = false and
  Environment.h3 = false and Environment.h4 = false and
  Environment.h5 = false and Environment.h6 = false and
  Environment.h7 = false  and Environment.h8 = false and
  Environment.h9  = true   and Environment.h10 = false and
  Environment.h11 = false and Environment.h12 = false and
  Environment.h13 = false and Environment.h14 = false and
  Environment.h15 = false and Environment.h16 = false and
  Environment.h17 = false and Environment.h18 = false and
  Environment.h19 = true and Environment.h20 = true and
  Environment.h21 = true and Environment.h22 = false and
  Environment.h23 = false and Initiator.name = true;

end InitStates



Formulae
  EF hazard1; -- CTL formula
  EF hazard2; -- CTL formula
  EF hazard3; -- CTL formula
end Formulae
```

Listing A.3: railwayModel.ispl

```
Agent Environment
  Vars:   -- Declaration of the states
   a1  : boolean; --a1 is a hazard
   a2  : boolean; --a2 is a hazard
   a3  : boolean; --a3 is a hazard
 sta1  : boolean; --sta1 is hazard
   c4  : boolean; --c4 causehazard
 sta4  : boolean; --sta4 is a hazard
  end Vars


  Actions = { none }; -- Declaration of actions
                      --none means no action or nothing happens.
  Protocol:
    Other : { none};
  end Protocol
  Evolution:
    a1 = true if   a2 = true; --RULES
    a2 = true if sta1 = true;
    a3 = true if   a1 = true;
    a3 = true if   a2 = true;
  sta1 = true if   c4 = true;
    c4 = true if sta4 = true;
  end Evolution
end Agent

Agent Initiator   -- This agent effects the Initiating mechanism
  Vars:
    -- we need variables for every agent, ignore it
    name : boolean;
  end Vars
  -- Declaring the actions
  Actions = {I1, I2, I3, I4, I5, I6,I7, none};
  Protocol:
    -- They all can happen in every state but nothing happens
    -- if there is no transition
    Other : {I1, I2, I3, I4, I5, I6,I7, none};
  end Protocol
  Evolution:
    name = true if name = true;
  end Evolution
end Agent

Evaluation
    hazard1 if Environment.a1 = true;  -- atomic proposition
    hazard2 if Environment.a3 = true;  -- atomic proposition
end Evaluation

InitStates
 -- Defining initial state, where contributing factors should
 -- be true and hazards false
 Environment.a1 = false and Environment.a2 = false and
 Environment.a3 = false and Environment.c4 = false and
 Environment.sta1 = false and
 Environment.sta4 = true and
 Initiator.name = true;

end InitStates



Formulae
  EF hazard1; -- CTL formula
  EF hazard2; -- CTL formula
end Formulae
```

Listing A.4: ITPModel.ispl

```
Semantics=MultiAssignment;

Agent Environment
 Vars:   -- Declaration of the states
  h5:    boolean; --hazard
  h51:   boolean; --hazard
  h52:   boolean; --hazard
  h511:  boolean; --hazard
  h6:    boolean; --hazard
  h61:   boolean; --hazard
  h62:   boolean; --hazard
  h63:   boolean; --hazard
  h621: boolean; --hazard
  h622: boolean; --hazard
  h623: boolean; --hazard
  h6211: boolean; --hazard
  h6212: boolean; --hazard
  h62111: boolean;
  h7:     boolean;
  h71:    boolean;
  h72:    boolean;
  end Vars


  Actions = { none }; -- Declaration of actions
                      --none means no action or nothing happens.
  Protocol:
    Other : { none};
  end Protocol
  Evolution:
   h5 = true  if h51 = true;
   h5 = true  if h52 = true;
   h51 = true if h511 = true;
   h6 = true  if h61 = true ;    -- RULE
   h6 = true  if h62 = true ;    -- RULE
   h6 = true  if h63 = true ;    -- RULE
   h62 = true if h621 = true ;    -- RULE
   h62 = true if h622 = true ;    -- RULE
   h62 = true if h623 = true ;
   h621 = true  if h6211 = true ;
   h621 = true  if h6212 = true ;
   h6211 = true if h62111 = true;
   h7 = true  if h71 = true;
   h7 = true  if  h72 = true;
  end Evolution
end Agent

Agent Initiator   -- This agent effects the Initiating mechanism
  Vars:
    name : boolean;                  --
  end Vars
  Actions = {I1, I2, I3, I4, I5, I6,I7, none}; -- Declaring the actions
  Protocol:
    Other : {I1, I2, I3, I4, I5, I6,I7, none}; --
  end Protocol
  Evolution:
    name = true if name = true;
  end Evolution
end Agent

Evaluation
    hazard1 if Environment.h5 = true;  -- atomic proposition
    hazard2 if Environment.h6 = true;  -- atomic proposition
    hazard3 if Environment.h7 = true;
end Evaluation

InitStates  -- Defining initial state

  Environment.h6 = false  and Environment.h61 = true and
  Environment.h62 = false and Environment.h63 = true and
  Environment.h621 = false and Environment.h622 = true and
  Environment.h623 = true and Environment.h6211 = false and
  Environment.h6212 = true and Environment.h62111 = true and
  Environment.h5 = false and Environment.h51 = false and
  Environment.h52 = true and Environment.h511 = true and
  Environment.h7 = false and Environment.h71 = true and
  Environment.h72 = true and
  Initiator.name = true;

end InitStates



Formulae
  EF hazard1; -- CTL formula
  EF hazard2; -- CTL formula
end Formulae
```