



Title	Development of Logic Programming Technique (LPT) for Marine Accident Analysis
Author(s)	Awal, Zobair Ibn
Citation	大阪大学, 2016, 博士論文
Version Type	VoR
URL	<a href="https://doi.org/10.18910/59594">https://doi.org/10.18910/59594</a>
rights	
Note	

*The University of Osaka Institutional Knowledge Archive : OUKA*

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

Doctoral Dissertation

Development of  
Logic Programming Technique (LPT) for  
Marine Accident Analysis

(ロジックプログラミングを用いた  
海難事故解析手法の開発)

Zobair Ibn Awal

July 2016

Graduate School of Engineering,  
Osaka University

Development of  
Logic Programming Technique (LPT) for  
Marine Accident Analysis

(ロジックプログラミングを用いた  
海難事故解析手法の開発)

Zobair Ibn Awal

A Dissertation Submitted to the  
Department of Naval Architecture and Ocean Engineering,  
Graduate School of Engineering, Osaka University,  
in Partial Fulfillment of the Requirements for the  
Degree of Doctor of Engineering

July 2016

Supervised by  
Professor Kazuhiko Hasegawa

# Acknowledgement

At first the author would like to express his heartfelt thankfulness and deepest gratitude to almighty Allah for giving him this opportunity to complete the doctoral research successfully in time.

Author's journey of higher studies began in 2011 when he was awarded the Monbukagakusho scholarship for Masters and Doctoral program at Osaka University, Japan. After successful completion of the Masters degree the author devoted himself in maritime accident research. This thesis thus represents a portion of author's learnings and contribution on safety science. The thesis also represents the author's hardship, agony, surprises, happiness and many other emotions that can only be felt!

The author gratefully remembers each and every moment with Professor Kazuhiko Hasegawa who has not only been a research supervisor, but also a friend, a guide, a problem solver and many more adjectives can be said beside his name. The thrilling experiences while attending the conferences (home and abroad) with him will always be remembered by the author. The author expresses an earnest "Thank You" to Professor Kazuhiko Hasegawa.

The Author conveys his regards and thankfulness to his parents for bringing him to this beautiful world. The author misses the incessant love of his younger brother, Ziad, as he is far away along with his parents. Nevertheless, their relentless prayers and blessings are always felt by the author. The Author prays for their long healthy and happy life.

The most amazing thing happened to the author is perhaps the birth of his baby girl, Nashwa Nawal. An awesome experience which is beyond any words of expression. The author expresses deepest gratitude to his wife Naomi, who had been patiently standing by the author during this journey in Osaka, Japan.

The author would like to thank his lab mates Akiko Sakurada, Satoshi Usada, Uchihiro Tadashi, Masahiro Sakai, Yaseen Adnan Ahmed and many more for their endless support not only in the laboratory but also supporting outside of the laboratory.

The author also expresses thankfulness to his colleagues at Bangladesh University of Engineering & Technology (BUET) for their encouragement and support towards continuing higher studies in Japan.

It is indeed extremely difficult to acknowledge all within this limited space. However, the author expresses his thankfulness to all whose names not mentioned here but contributed towards completion of his study and living in Japan.

Zobair Ibn Awal  
3<sup>rd</sup> July 2016  
Osaka, Japan

# Contents

<b>Acknowledgement.....</b>	<b>i</b>
<b>Contents.....</b>	<b>ii</b>
<b>List of Tables.....</b>	<b>iv</b>
<b>List of Figures .....</b>	<b>v</b>
<b>Abstract .....</b>	<b>1</b>
<b>Chapter 1: Understanding the Accident Problem.....</b>	<b>5</b>
1.1 Introduction.....	5
1.2 The Changing Human Civilization .....	5
1.3 Maritime Accidents: The Problem.....	8
1.4 The Problem Space .....	14
1.5 Chapter Summary .....	18
<b>Chapter 2: Study on Notable Maritime Disasters .....</b>	<b>19</b>
2.1 Introduction.....	19
2.2 Accident of MV Costa Concordia.....	19
2.3 Accident of MV Bright Field .....	23
2.4 Accident of MV Planet V.....	26
2.5 Chapter Summary .....	29
<b>Chapter 3: Review of Accident Theories and Models.....</b>	<b>30</b>
3.1 Introduction.....	30
3.2 Classification of Accident Theories and Models .....	30
3.3 Accident Proneness and Statistical Models .....	36
3.4 Domino Theory .....	40
3.5 Epidemiologic Theory.....	42
3.6 Control Theoretic Hypothesis .....	45
3.7 System Theoretic Hypothesis.....	46
3.8 Tools for Accident Analysis and Investigation.....	49
3.9 Chapter Summary .....	51
<b>Chapter 4: Fundamentals of Logic Programming Technique (LPT).....</b>	<b>52</b>
4.1 Introduction.....	52
4.2 The Monkey-Banana Problem .....	52
4.3 Definition and Types of Logic .....	56
4.4 Propositional Logic vs Predicate Logic .....	60
4.4.1 Propositional Logic.....	60
4.4.2 Predicate Logic .....	61
4.4.3 A Comparison .....	62
4.5 Logic Programming .....	62

4.5.1 <i>Programming with Relations</i> .....	65
4.5.2 <i>Use of Prolog: Clause, Facts and Rules</i> .....	65
4.6 Fundamentals of Arguments .....	66
4.6.1 <i>Deductive Argument</i> .....	66
4.6.2 <i>Inductive Argument</i> .....	67
4.6.3 <i>Consistency, Validity, Soundness and Completeness</i> .....	67
4.7 Technique 1: Propositional Logic Based Technique .....	68
4.8 Technique 2: Agent Based Perception-Action Technique .....	73
4.8.1 <i>Design of Simple Reflex Agent</i> .....	74
4.8.2 <i>Performance, Environment, Actuator and Sensors (PEAS)</i> .....	76
4.8.3 <i>Ship Agent</i> .....	77
4.8.4 <i>Captain Agent</i> .....	78
4.8.5 <i>SOOW Agent</i> .....	79
4.8.6 <i>JOOW Agent</i> .....	79
4.8.7 <i>Helmsman Agent</i> .....	80
4.9 Chapter Summary .....	82
<b>Chapter 5: Development of Logic Worlds and Analysis</b> .....	<b>83</b>
5.1 Introduction .....	83
5.2 Example 1 – Propositional Logic Based Technique .....	83
5.3 Example 2 – Propositional Logic based Technique .....	90
5.4 Example 3 – Agent Based Perception-Action Technique .....	96
5.4.1 <i>Assumptions</i> .....	96
5.4.2 <i>Ship Agent</i> .....	97
5.4.3 <i>Captain's Knowledge</i> .....	98
5.4.4 <i>SOOW's Knowledge</i> .....	99
5.4.5 <i>JOOW's Knowledge</i> .....	100
5.4.6 <i>Model Run and Discussion</i> .....	101
5.5 Chapter Summary .....	105
<b>Chapter 6: Conclusion and Recommendation</b> .....	<b>106</b>
6.1 Some Important Features of Logic Programming Technique (LPT) ..	106
6.2 Concluding Remarks .....	108
6.3 Recommendations .....	110
<b>References</b> .....	<b>112</b>
<b>Appendix A: Prolog Code of Monkey Banana Problem</b> .....	<b>125</b>
<b>Appendix B: Solution of Monkey Banana Problem</b> .....	<b>129</b>
<b>Appendix C: Example 1 - Propositional Logic Based Technique</b> .....	<b>131</b>
<b>Appendix D: Example 2 - Propositional Logic Based Technique</b> .....	<b>136</b>
<b>Appendix E: Example 3 - Agent Based Perception-Action Technique</b> .....	<b>140</b>
<b>Appendix F: List of Publications</b> .....	<b>152</b>

## List of Tables

Table 2.1: List of errors took place before the accident of MV Costa Concordia.....	22
Table 2.2: Events occurred prior to the accident of MV Bright Field.....	25
Table 2.3: Final events before the MV Planet V accident.....	27
Table 4.1: Summary of definition and characteristics of logic.....	57
Table 4.2: Difference between propositional logic and predicate logic. ....	63
Table 4.3: Example of PEAS definition of different agents.....	77
Table 4.4: Example of perception-action arguments.....	82
Table 5.1: Ten propositional logics for Example 1. ....	84
Table 5.2: Fourteen propositional logics for Example 2. ....	91
Table 5.3: Assumptions for ship maneuvering model.....	98
Table 5.4: Captain's perceptions. ....	99
Table 5.5: Captain's actions. ....	99
Table 5.6: SOOW's perceptions.....	100
Table 5.7: SOOW's actions. ....	100
Table 5.8: JOOW's perceptions.....	100
Table 5.9: JOOW's actions.....	101
Table 5.10: Results of logical deductions of crew perception-actions (small-scale chart chosen for navigation).....	103
Table 5.11: Results of logical deductions of crew perception-actions (large-scale chart chosen for navigation).....	104

## List of Figures

Figure 1.1: World population growth through history (McFalls, 2003).....	6
Figure 1.2: Evolution of human beings in terms of travel speed in history. ....	8
Figure 1.3: Some major maritime accidents - (a) Fire onboard SS Morrow Castle in 1934 (SS Morro Castle (1930), 2016), (b) SS Torrey Canyon oil spill in 1967 (Oil spills, 2013), (c) Amoco Cadiz oil spill in 1978 (Amoco Cadiz oil spill, 2016), (d) Herald of Free Enterprise accident in 1987 (Early, 2016), (f) Piper Alpha disaster in 1988 (Piper Alpha, 2016), (g) Tanker ship Prestige after splitting in two in 2002 (Hamilos, 2013) and (h) Accident of Costa Concordia in 2012 (Derbyshire, 2012).....	9
Figure 1.4: A timeline of notable marine accidents and actions taken by authorities. ....	11
Figure 1.5: The socio-technical system involved in risk management (Rasmussen, 1997). ....	17
Figure 2.1: The final path of MV Costa Concordia before the accident in timeline (Costa Concordia Disaster, 2016). ....	20
Figure 2.2: Allision of MV Bright Field (National Transportation Safety Board, 1998). ....	24
Figure 2.3: A Snapshot of MV Planet V very close to the pontoon of MTS Vantage (Dutch Safety Board, 2013). ....	26
Figure 2.4: The final path of MV Planet V (green line) and MTS Vantage (red line) (Dutch Safety Board, 2013). ....	28
Figure 3.1: Review of accident theories and models by different researchers in timeline.....	31
Figure 3.2: Major accident theories and their chronological appearance (Awal and Hasegawa, 2015a). ....	35
Figure 3.3: Classification of accident theories/models/etc in timeline.....	36
Figure 3.4: Research works on accident proneness and statistical models. ....	37
Figure 3.5: Research works related to domino theory in timeline.....	41
Figure 3.6: Accident studies in timeline based on epidemiologic concept. ....	43
Figure 3.7: Control theoretic accident models in timeline. ....	46
Figure 3.8: Systems theoretic research works in timeline. ....	48
Figure 3.9: Various tools for accident analysis in timeline. ....	50



Figure 4.1: Classic monkey-banana problem: ‘how’ to get the banana?.....	53
Figure 4.2: Solution of the monkey-banana problem. ....	54
Figure 4.3: Classification of different types of logic.....	58
Figure 4.4: Simple world with two ships and underwater rocks. ....	69
Figure 4.5: Possible accident outcomes – (a) collision between ships and (b) grounding of ship(s) with the underwater rocks. ....	72
Figure 4.6: A schematic diagram of simple reflex agent (Russel et al., 2010). ....	75
Figure 4.7: Agent based perception-action concept. ....	76
Figure 4.8: Definition of ship agent. ....	78
Figure 4.9: Definition of Captain agent.....	79
Figure 4.10: Definition of SOOW agent. ....	80
Figure 4.11: Definition of JOOW agent. ....	80
Figure 4.12: Definition of Helmsman agent.....	81
Figure 5.1: Example 1 – case one.....	86
Figure 5.2: Example 1 – case two. ....	87
Figure 5.3: Example 1 – case three. ....	88
Figure 5.4: Example 1 – case four.....	89
Figure 5.5: Example 2 – case one.....	92
Figure 5.6: Example 2 – case two. ....	94
Figure 5.7: Example 2 – case three. ....	95
Figure 5.8: Ship’s path in two cases: Following small-scale chart and following large scale chart.....	102

# **Abstract**

Maritime accident is a big problem. Since the early days of sea voyage by sailing to the modern days of satellite navigation, maritime accidents are nevertheless a huge concern for all. It appears that even though there have been so many technological breakthroughs in the past century, the nature of maritime accidents in many cases are still the same. As an example, in between the accident of the Titanic (1912) and the accident of the Costa Concordia (2012) one hundred years have passed, nevertheless, the events that unfolded before the accident are similar in many instances. The individual and organizational factors that affect the events toward an accident exist in harmful manner as several research works have pointed out. The traditional approaches of risk analysis have been developed and applied extensively in the last few decades, even so, the success of such application still questions whenever an accident occurs. Moreover, risk based approaches are less useful in discovering unknown sequence of events to accidents. This research work, therefore, attempts to develop a new tool for analysis of maritime accidents, which is called the Logic Programming Technique (LPT).

Chapter 1 of this thesis discusses the nature of maritime accident problem and approaches towards accident prevention. Maritime accidents are generally one-off events and rare while compared to other types of accidents (e.g. road accidents). Such accidents are practically impossible to simulate for further study and investigation. Therefore, historically, various maritime authorities have taken preventive measures or developed regulations in reactive approach rather than proactive approach, i.e. many significant

regulations were developed after major accidents occurred. The accident problem is further complicated by involvement of various organizations and individuals. In this connection, the problem space was studied and it was comprehended that different group of professionals are involved and can contribute in controlling maritime accidents. Hence, it can be concluded from the study of Chapter 1 that maritime accident is a complex socio-technical problem and the problem space is diversified and vast.

Chapter 2 discusses some notable maritime accidents. As the previous Chapter presents an overview of the maritime accident problem, this chapter gets into more detail. The accidents of MV Costa Concordia (2012), MV Bright Field (1996) and MV Planet V (2012) were studied in detail. The study reveals that maritime accidents can occur due to complex interactions between individual professionals including ship crew. These interactions often seem useful or harmless which eventually masks the necessary and sufficient causes of accidents. Therefore, predicting maritime accidents become difficult.

Chapter 3 reviews the accident theories and models. Since the earlier two chapters identified the nature of maritime accidents, it will be coherent to study the accident theories and models so that the present state of the knowledge is understood. The study reveals that accident theories and models are evolving over the past century and have shifted from individual faults to organization failures, epidemiologic concepts to the metaphoric model of ‘Swiss Cheese’ and many others. This implies an indication that the changes in society (technological, societal, economical and much more) have given birth to new ways of accident occurrence. Hence, accident theories and models became diversified and vast. Nevertheless, despite such significant developments in wider

perspective, very few (if none) accident theories or models are able to deduce computationally ‘how’ an accident may unfold for a given scenario.

The previous chapters describe the complexity and vastness of the accident problem, therefore, Chapter 4 introduces the Logic Programming Technique (LPT) with its advantages over the given problem space. LPT is a method of logic computation which has been proposed in this study for deducing chain of events leading to accidents. The monkey-banana problem is studied based on the logic programming concepts. In this connection an accident problem is seen as a ‘how’ problem rather than a ‘what’ problem. This kind of approach is cardinal in logic programming. The fundamentals of logic including definitions and classifications are studied and presented. The characteristics of arguments are discussed in relation to development of logic worlds. The concept of agent based perception-action of ship crew is introduced. Thereby, two different methods have been developed in this study: (1) Propositional Logic Based Technique and (2) Agent Based Perception-Action Technique. The fundamentals of these methods are described.

Chapter 5 presents the development of logic worlds and results of logic computations. Three different examples were shown and studied based on the fundamentals discussed in Chapter 4. These examples show the techniques for creating logic worlds and programming methods in Prolog programming language. Different scenarios are constructed using propositions which are utilized during logic deductions. The interaction of different agents (crews and ship) are shown and the results of such interaction on accidents are also visualized. The examples reveal that logic models can be constructed using theories from different disciplines (engineering, social science,

psychology etc.) and coded into a single program. This is an important aspect because accident studies require knowledge from multiple disciplines and LPT can offer a platform in this regard. The merits and demerits are also discussed later in the chapter.

Finally, Chapter 6 concludes the finding of this research work. It can be concluded that accident problems are multi-disciplinary. The problem of maritime accidents, in particular, is diversified and complicated. So far very few (if none) accident theories or models have been developed that can specifically deduce ‘how’ an accident may occur. However, the Logic Programming Technique (LPT) has the potential of deducing and revealing ‘how’ an accident may take place. There are many advantages of such approach and there exists significant scope for further development. In future, this approach may significantly contribute in building a safer and accident-free society.

# **Chapter 1: Understanding the Accident Problem**

## **1.1 Introduction**

This chapter seeks answers to some generalized questions such as “why accidents occur?”, “how does accident occur?”, “how to prevent accidents?”, “what is the problem space?” and so on. It points out the change in human civilization through history and its possible affects on the society, particularly the changes that cause accidents. In this connection, some major maritime accidents are highlighted in timeline and the reactive measures associated with these accidents are depicted. Later on the discussion shifts over to the problem space, which provides the diversified viewpoint of accident problem.

## **1.2 The Changing Human Civilization**

Since the dawn of human civilization, the human society has been changing perpetually in terms of culture, economy, medicine, technology and many others aspects. The aspiration for development has brought such changes, which has given better living in terms of safety, security, subsistence, sheltering and so on. Such changes are distinguished by names based on the significant achievements made by human civilization on that time, such as Stone Age, Bronze Age, Iron Age and so on (shown in Figure 1.1). These changes are more significant in the modern ages compared to middle ages or earlier. One significant change can be seen is the growth in population as shown in Figure 1.1. Such dramatic rise in population has been reshaping the society in various

aspects continuously. One may agree that this changing society is driven by new inventions and new technologies.

### World Population Growth Through History

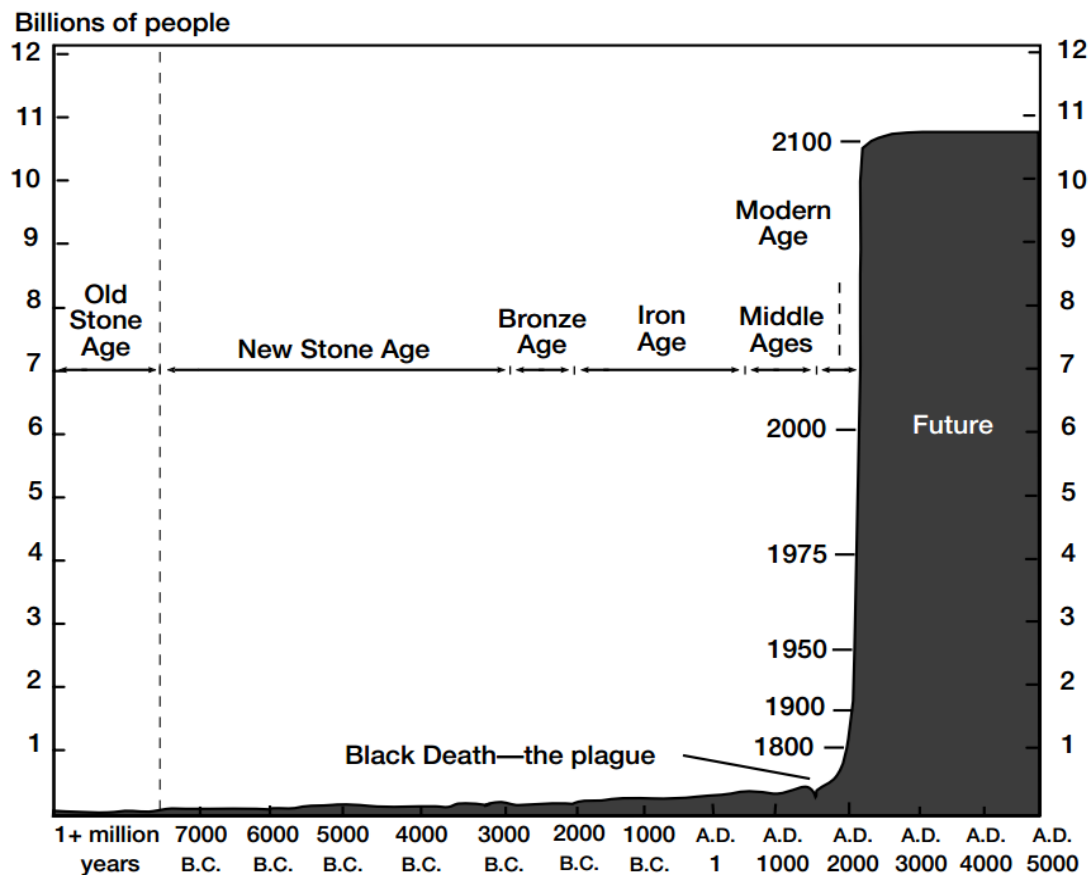


Figure 1.1: World population growth through history (McFalls, 2003).

Probably one of the most detrimental effects of the ever-changing civilization is on the adaptation of human beings' capabilities. Human civilization has been slowly changing prior to modern ages. Adaptation to the society perhaps was easier than at present. However, in modern ages, all aspects of living have been boosted tremendously. For example, industries roll products faster, constructors build faster, people travel faster – cars, ships, airplanes, trains all are moving faster than ever, and so on. In today's

technology, a standard car may be built in less than one day out of a production line; in an hour people can travel almost a thousand kilometers by jet planes; and communication signals can be sent and received around the globe instantly (fraction of a second). This gives some idea of how fast things can happen or move around in today's world. Therefore, people have very little time to adapt when they encounter something new.

Another example can be seen in Figure 1.2 that shows the evolution of human beings in terms of travel speed in history. This figure points out that in recent times, past century in particular, the travel speed has increased tremendously compared to the dawn of human civilization. This gives rise to a problem of perception as there are reasons to believe that human being is not evolved well enough to cope with this pace not just only in travel speed but in all aspects of life. Lack of information occurs when things move very fast or something happens quickly and perhaps this lack of information contributes to improper decisions and/or mistakes. Sometimes these mistakes lead to immediate accidents and sometimes lead to accidents at a later period.

A question may, therefore, arise that how the lack of information or knowledge is affecting the maritime community? Perhaps, the answer will be, "enormously!". Throughout the past century, maritime community has encountered accidents that had not been anticipated before. Each time an accident had taken place, the maritime community had to learn how it happened and thereby apply preventive measures. It is discussed further in the next section that so far the maritime community has been dealing safety reactively because of information or knowledge deficiency.



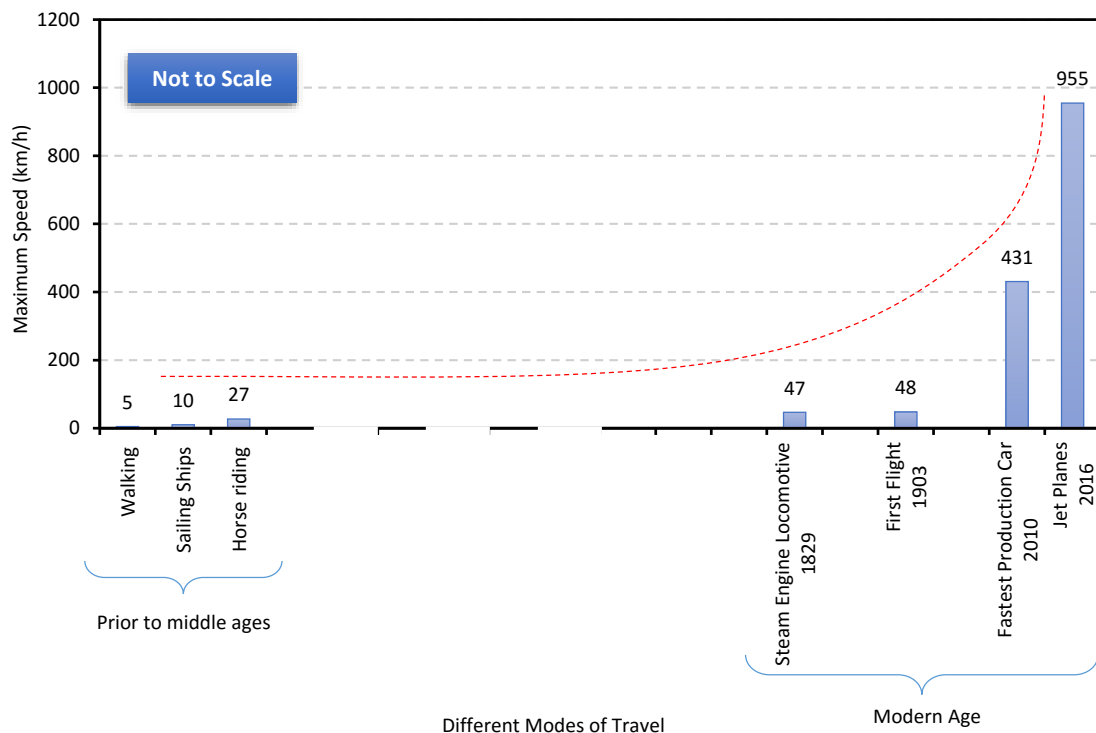


Figure 1.2: Evolution of human beings in terms of travel speed in history.

### 1.3 Maritime Accidents: The Problem

Maritime accidents have shocked the world every now and then. Since the early days of sea voyage by sailing ships to this modern day of satellite navigation, Automatic Identification System (AIS), and so many more technologies, maritime accidents are still a concern for all involved in this business. It appears that even though there have been so many technological breakthroughs, the nature of maritime accidents, in many cases, are still the same as before. Historically, the list of maritime accidents is quite extensive, and the number of casualties is grievously high. However, there are specific incidents that fell out over the last one hundred years, which forced to international agreements on safety, liability, and environmental controls, essentially reshaping the marine industry. Figure 1.3 shows a glimpse of some notable accidents that took place over the past century.



(a)



(b)



(c)



(d)



(f)



(g)



(h)

Figure 1.3: Some major maritime accidents - (a) Fire onboard SS Morro Castle in 1934 (SS Morro Castle (1930), 2016), (b) SS Torrey Canyon oil spill in 1967 (Oil spills, 2013), (c) Amoco Cadiz oil spill in 1978 (Amoco Cadiz oil spill, 2016), (d) Herald of Free Enterprise accident in 1987 (Early, 2016), (f) Piper Alpha disaster in 1988 (Piper Alpha, 2016), (g) Tanker ship Prestige after splitting in two in 2002 (Hamilos, 2013) and (h) Accident of Costa Concordia in 2012 (Derbyshire, 2012).

A timeline is constructed in Figure 1.4, which shows the notable maritime accidents and necessary measures taken by different authorities after the accident. The most famous disaster of all, the Titanic, struck the first blow for real international cooperation on safety regulations, known as the International Convention for the Safety of Life at Sea (SOLAS). Two years after the Titanic tragedy SOLAS was adopted in 1914. This is the primary safety book from which most other policies and regulations leaped. SOLAS is considered the safety bible for the maritime industries and is updated on a regular basis.

The Titanic may have made a tremendous impact, but the 1934 sinking of the Morro Castle off the New Jersey coast, which left one hundred and twenty-six dead, also left quite a safety legacy in its aftermath. The ship went up in flames and the accident not only led to new fire suppression, protection and control regulations and equipment requirements, it served as the impetus for both the U.S. Merchant Marine Act of 1936, which created the Maritime Commission, and the adoption of a significant upgrade to SOLAS in 1948. It also led to federally mandated officer training requirements and eventually, to the establishment of the federal maritime academy in the United States (Keefe, 2014).

The Torrey Canyon oil spill off French and Cornish coasts in 1967 led to the International Convention for the Prevention of Pollution from Ships (MARPOL) in 1973. This tragic event is sometimes credited with moving the IMO into environmental and legal issues with the Civil Liability Convention of 1969. Activated in 1975, that policy was adopted to ensure that adequate compensation is available to victims of oil pollution

resulting from maritime casualties involving oil-carrying ships, and places the liability for such damage on the owner of the polluting ship (Keefe, 2014).

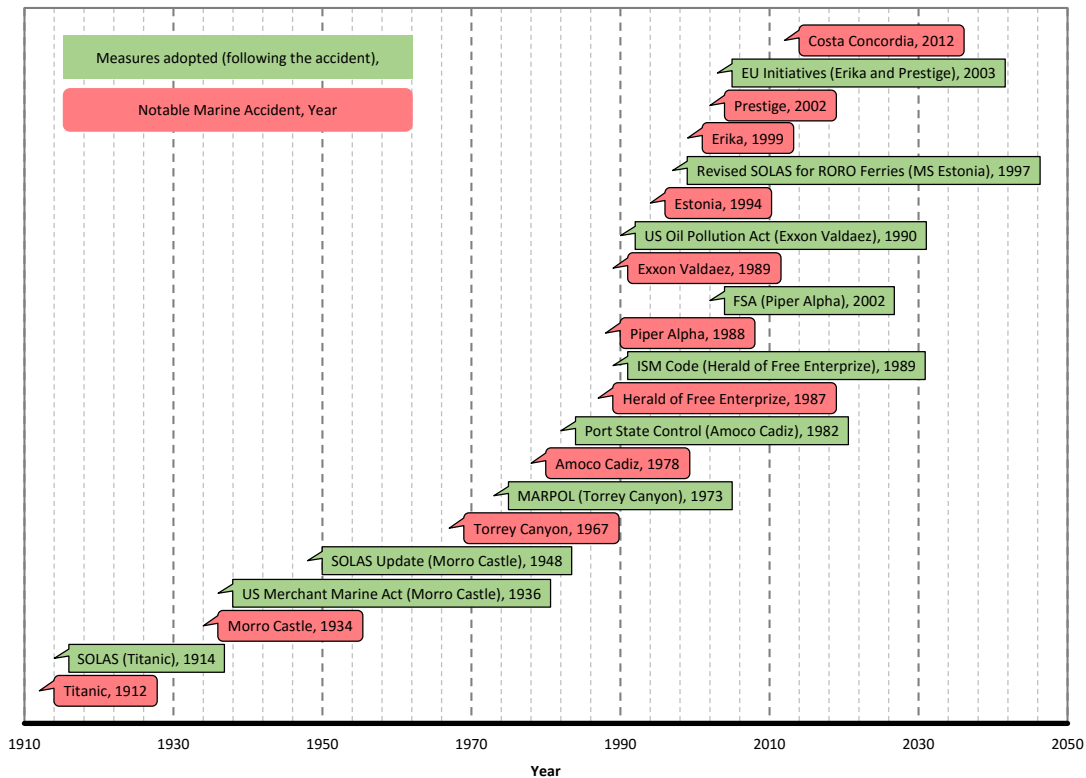


Figure 1.4: A timeline of notable marine accidents and actions taken by authorities.

On March 16, 1978, the Amoco Cadiz tanker ran aground three miles from the coast of Brittany, France due to a steering gear failure. It split in three before sinking, creating the largest oil spill of its kind in history to that date – more than one and half million barrels. Public outcry and political pressure resulted in significant updates to both MARPOL and SOLAS, and the addition of safety and pollution audits that led to in 1982 to the Paris Memorandum of Understanding (Paris MoU), which established Port State Control. The beauty of port state control is that it enabled an international port inspection

system that makes it impossible for non-compliant ships to hide. It also led to the International Convention on the Standards of Training, Certification and Watchkeeping for Seafarers (STCW) in 1978 (Keefe, 2014).

The remarkable capsizing of the Herald of Free Enterprise in 1987 took place minutes after leaving the harbor in Zeebrugge, Belgium. Incredibly, the bow door was left open, resulted in the loss of one hundred ninety three out of the five hundred and thirty nine passengers and crew. This accident led to the adoption of the Guidelines on Management for the Safe Operation of Ships and for Pollution Prevention, or the International Safety Management (ISM) Code. This code is designed to prevent damage to life and the environment at sea, by requiring each vessel to have a working, audited, Safety Management System (SMS). It also required shipping companies to have a license to operate (Keefe, 2014).

The Exxon Valdez ecological disaster of 1989 led to the first Port state establishment of policy with international repercussions - the Oil Pollution Act (OPA) of 1990 in the U.S., which mandated that all tankers entering U.S. waters be double-hulled – a requirement that eventually became the rule internationally, especially following several oil spills in European waters. OPA greatly increased federal oversight of maritime oil transportation, toughened liability and provided greater environmental safeguards. It also put the spotlight on drug abuse in the merchant marine and led to related programs and reforms (Keefe, 2014).

The RoRo ferry MS Estonia sank in heavy seas on September 28, 1994 in circumstances very similar to the Herald of Free Enterprise. In this case the bow door failed, letting in too much water, sinking the boat, and killing eight hundred and fifty-two people out of the one thousand on board. To improve the survivability of ferries, it also led to changes in the design parameters so that ferries can take up to a half meter of water on the car deck before the ship starts to list (Keefe, 2014).

Two accidents Prestige in 2002 and Erika in 1999 happened in almost the same spot off the northwest coast of France and Spain, and both became a huge political issue, leading to Eur-OPA, the European equivalent of OPA90. These incidents led to a monumental acceleration of the schedule to phase out single-hulled tankers. Incidents like these generated focus on environmental issues as spills mounted, each seemed worse than the last. There was a palpable shift in focus to combating pollution, and providing adequate training and certification of crews. Even the oil companies got into the act, proactively launching spill response coalitions to respond to catastrophes and providing training (Keefe, 2014).

In recent times several accidents have shocked the world as well. The accident of MV Costa Concordia and accident of MV Sewol are indeed mentionable in this regard. On January 13<sup>th</sup> 2012 the Italian cruise ship Costa Concordia capsized and sank after striking an underwater rock obstruction off Isola del Giglio, Italy. Thirty-two people lost their lives. The accident of MV Sewol in 2014 shocked the world when it capsized while carrying four hundred and seventy six lives and most of them were secondary school students. The sinking of MV Sewol resulted in widespread social and political reaction

within South Korea and the world as well. Criticism were raised on various aspects including crew operation, management and regulations as well. These two terrible incidents somehow remind that accidents seem to happen for the same underlying human and organizational reasons even though a century of improvements over technology and safety regulations have taken place since the accident of Titanic in 1912.

#### **1.4 The Problem Space**

Efforts to improve the safety of systems have often, some might say always, been predominated by hindsight. This means that the safety measures are taken once an accident had taken place. The reason may be found in Schröder-Hinrichs et al. (2012). According to the authors, maritime accidents in the past have demonstrated that socio-technical systems in the maritime industry have become too complex to be understood by means of linear or complex linear accident models, which often build the core for traditional risk assessments. The different interactions between operators and subsystems are so diverse and context-dependent that it becomes impossible to forecast a system's performance in its entirety.

However, traditionally maritime accidents, other accidents as well, are guarded or prevented by applying different barriers. Such barriers are believed to be effective until some new type of accidents occur. Hollnagel (2008) discussed that the characteristics of different barrier systems (physical, functional, symbolic, and incorporeal) and their relative advantages and disadvantages. The author argued that while barriers are necessary, they represent a reactive approach which is insufficient by itself to guarantee safety. In short, some events can be prevented by applying barriers while some barriers

cannot prevent all types of accidents. In this connection, Westrum (2006) proposed a distinction between three threats (or events): regular threats, irregular threats, and unexampled events. Regular threats are events that occur so often that the system can develop a standard response. An example is medication errors that only implicate a single patient and which potentially can be brought under control. Irregular threats are one-off events where their sheer number makes it practically impossible to provide a standard response. Even though they are imaginable, they are usually unexpected. Finally, unexampled events are those that are virtually impossible to imagine and which exceed the responders' collective experience. According to Hollnagel et al. (2006), irregular threats and unexampled events are infrequent and unusual, they cannot be treated in the conventional way, i.e., by designing barriers to avoid them. Their distinguishing feature seems to be that they emerge out of a situation. Thereby, accident becomes as emergent phenomena. Perhaps some maritime accidents fall under the later categories and therefore, applying different types of barrier cannot prevent the rare type of accidents.

Recent accident studies suggest that control theoretic approaches useful in understanding the accident problem within the social context. Rasmussen (1997) points out that injuries, contamination of environment, and loss of investment all depend on loss of control of physical processes capable of injuring people or damaging property. The propagation of an accidental course of events is shaped by the activity of people, which either can trigger an accidental flow of events or divert a normal flow. Safety, then, depends on the control of work processes so as to avoid accidental side effects causing harm to people, environment, or investment.



According to Rasmussen (1997) many levels of politicians, managers, safety officers, and work planners are involved in the control of safety by means of laws, rules, and instructions that are formalized means for the ultimate control of some hazardous, physical process. They seek to motivate workers and operators, to educate them, to guide them, or to constrain their behavior by rules and equipment design, so as to increase the safety of their performance. The socio-technical system actually involved in the control of safety is shown in Figure 1.5. It represents the problem space with several academic disciplines involved at each of the various levels. At the top, society seeks to control safety through the legal system: safety has a high priority, but so has employment and trade balance. Legislation makes explicit the priorities of conflicting goals and sets boundaries of acceptable human conditions. Research at this level is within the focus of political and legal sciences.

Next, the level of authorities and industrial associations, workers' unions and other interest organizations. Here, the legislation is interpreted and implemented in rules to control activities in certain kinds of work places, for certain kinds of employees. This is the level of management scientists and work sociologists. To be operational, the rules now have to be interpreted and implemented in the context of a particular company, considering the work processes and equipment applied. Again, many details drawn from the local conditions and processes have to be added to make the rules operational and, again, new disciplines are involved such as work psychologists and researchers in human-machine interaction. Finally, at the bottom level the engineering disciplines involved in the design of the productive and potentially hazardous processes and equipment and in

developing standard operating procedures for the relevant operational states, including disturbances.

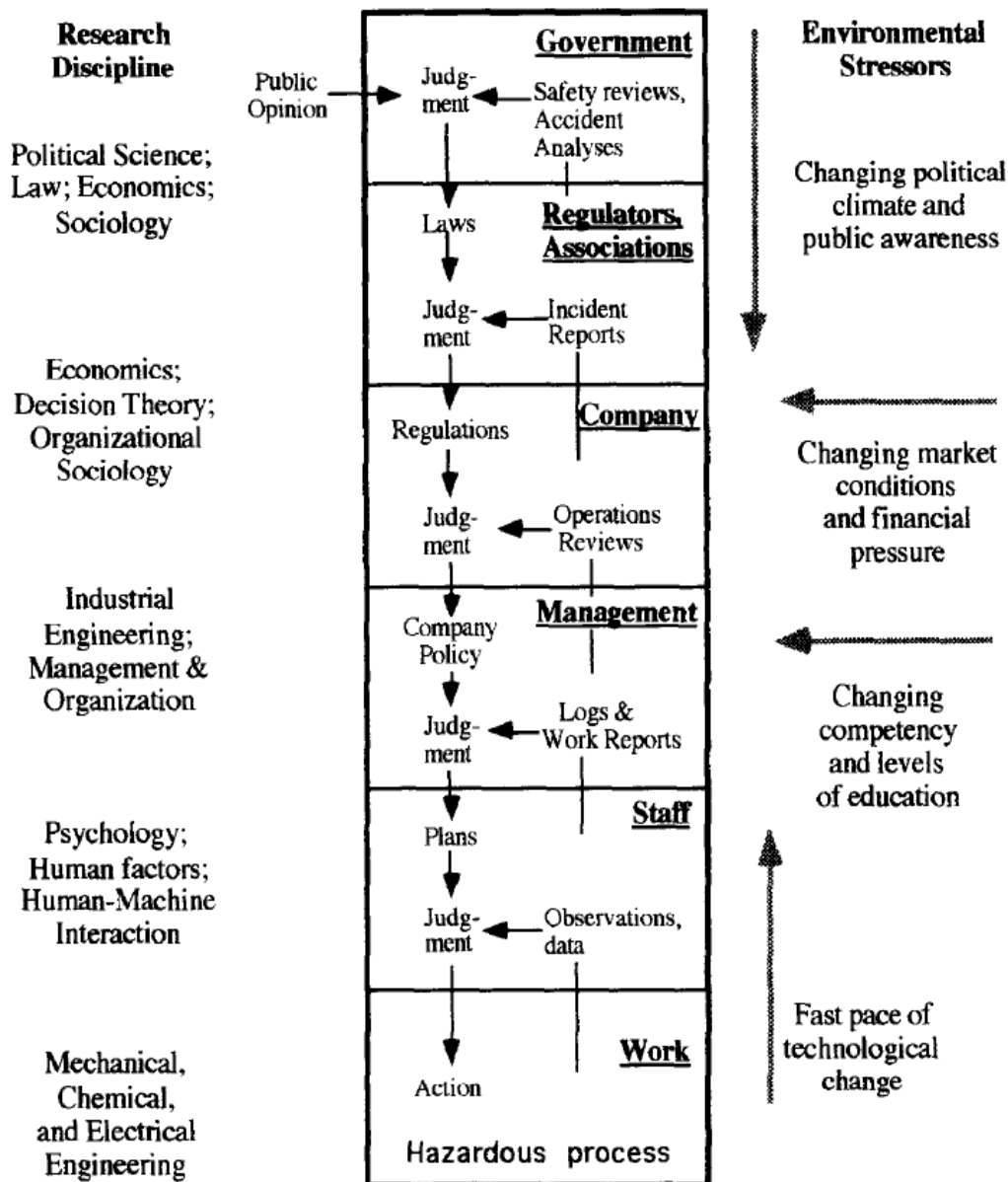


Figure 1.5: The socio-technical system involved in risk management (Rasmussen, 1997).

Therefore, it is comprehensible that controlling accidents is a multidisciplinary subject and actors in multiple levels within the society have different roles to play. The prevention of accidents using barrier is also challenged by the emergent nature of accident itself.

### **1.5 Chapter Summary**

This chapter attempted to show the nature of accident problem. It has argued that the way human civilization has evolved over the years is unique. Particularly in the past couple of century, the change is tremendous. Maritime community is also facing the same changes along with the society. In addition, the severity of maritime accidents have become bad to worse over the years. The complexity of the sociotechnical system associated with maritime community is too difficult to comprehend and resolve. However, it may be summarized that maritime accidents like all other accidents may be preventable if the problem space is properly understood and appropriate preventive measures/tools are developed and employed.

## **Chapter 2: Study on Notable Maritime Disasters**

### **2.1 Introduction**

This chapter investigates the occurrence of maritime accidents. The idea is to comprehend the question ‘how accidents have occurred?’. Therefore, three notable maritime accidents such as the accident of MV Costa Concordia in 2012, the accident of MV Bright Field in 1996 and the Accident of MV Planet V in 2012 are studied.

### **2.2 Accident of MV Costa Concordia**

The accident of MV Costa Concordia took place on 13<sup>th</sup> January 2012. The ship grounded on the rocks Le Scole, near Giglio Island, Italy (Marine Casualties Investigative Body, Italy, 2013). The ship operated by Costa Crociere – a subsidiary of Carnival Corporation – was on route from Civitavecchia to Savona, carrying over four thousand two hundred people on board. Thirty-two lost their lives and sixty were injured in the accident. With its gross tonnage of one hundred and fourteen thousand, thirteen decks, two hundred and ninety meters of length, thirty-five meters of beam and eight meters of draught, Costa Concordia was launched in 2006, and at the time, it was the largest Italian cruise ship ever built. The accident demonstrated that catastrophe may occur even with ships that are considered masterpieces of modern technology and despite more than hundred years of regulatory and technological progress in maritime safety since the accident of the Titanic. Figure 2.1 shows the final path of the MV Costa Concordia (Costa Concordia Disaster, 2016). A study by Lieto (2012) identifies several operational errors during the voyage,

which resulted in the accident. The author utilized Reason's Organizational Accident Model (Reason, 1997) to identify the errors. Table 2.1 shows the list of errors with necessary description. For the first error, the external influence of paying a tribute to the mentor and a request to change the voyage plan makes the Captain to settle to change in the voyage plan. However, the Captain decided to take informal procedure because of the regulatory limitations from the company.

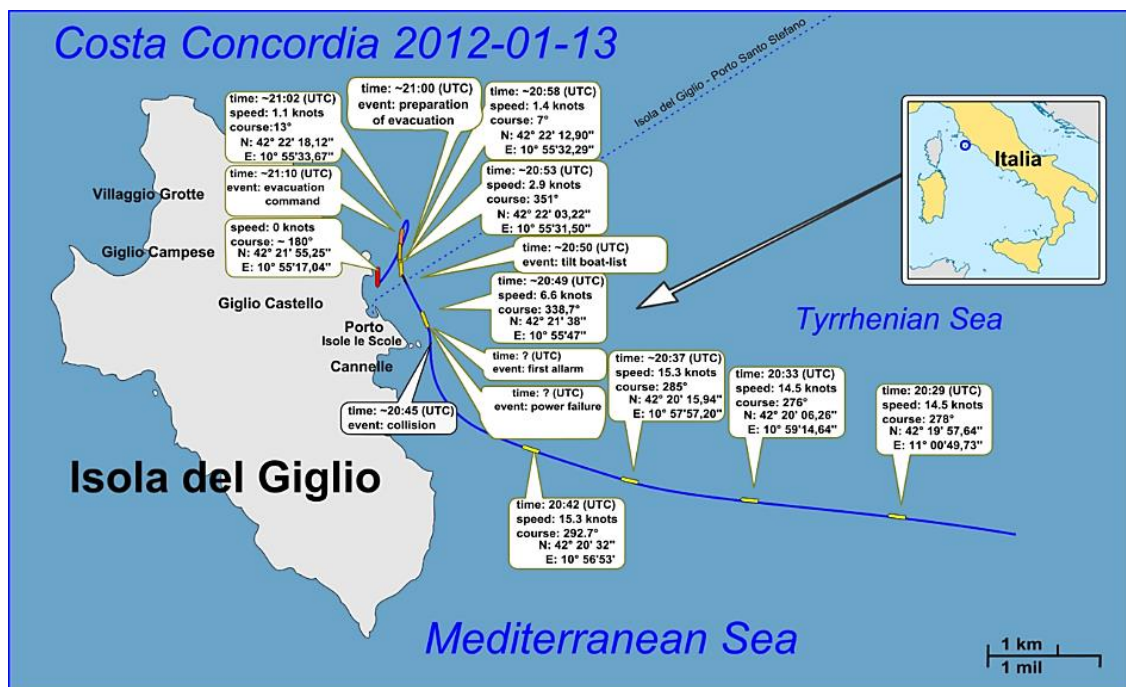


Figure 2.1: The final path of MV Costa Concordia before the accident in timeline (Costa Concordia Disaster, 2016).

Secondly, limited time for modifying the voyage plan, and captain's reliance on Senior Officer of the Watch (SOOW) resulted in a decision of planning the voyage on large-scale charts. Here the Captain could have intervened SOOW to draw the voyage on small-scale charts where the danger of grounding could have been spotted. But the limited

time and informal procedure resulted both the Captain and the SOOW to decide to plan the voyage on large scale charts.

The third error triggered when there was no proper route monitoring. This happened in two cases along the voyage. Firstly, the Junior Officer of the Watch (JOOW) did not have 'planned larger scale charts' to fix ship's position. Therefore, JOOW could not detect any danger. As there was no observed danger and there is informal procedure, the JOOW decided not to challenge. Secondly, in another case JOOW left route monitoring and went to assist the Helmsman, as there was language/communication barrier between the Captain and Helmsman.

The fourth error was regarding to route monitoring on Integrated Navigation System (INS). The chart alarm was set to go on if the radar distance is 2000 meter or less from the ground. It was not set for crossing the 10-meter bathymetric line. If it was selected, the captain might have received a warning alarm and could take actions much earlier (as soon as 10-meter draft compromised).

At the final stage of the approach, the Captain took over command from SOOW. So far, SOOW or JOOW did not challenge the Captain for any decision in any form. Captain's intentions and expected outcomes were not clear. Because of the presence of guests and hotel manager, his role as a team leader was not fulfilled. The lack of challenge from the ship crew could be the fifth error.

Table 2.1: List of errors took place before the accident of MV Costa Concordia.

Error	Logical Description
<b>1<sup>st</sup> Error</b>	<u>Voyage Planning</u>  <i>Captain Decision: Change in voyage plan.</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>1. The mentor of the Captain is in the Giglio Island.</li> <li>2. The Hotel manager also requests the Captain to sail past.</li> </ol>
<b>2<sup>nd</sup> Error</b>	<u>Route Planning on Paper Charts</u>  <i>Senior Officer Decision: Plan route on large scale paper charts - incomplete route planning.</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>3. Limited time to start the voyage.</li> <li>4. Informal procedure of Captain.</li> </ol>
<b>3<sup>rd</sup> Error</b>	<u>Route Monitoring on Papers Charts</u>  <i>Junior Officer Decision: Faulty route monitoring – no danger observed on the chart.</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>5. On large scale paper charts the rocks are invisible.</li> <li>6. On the final stage of approach to the island, the Junior Officer left route monitoring and went to assist Helmsman to translate Captain's commands.</li> </ol>
<b>4<sup>th</sup> Error</b>	<u>Route Monitoring on INS</u>  <i>Senior Officer Action: Wrongly set INS chart alarm.</i> <i>No challenge from Captain (Captain's Decision).</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>7. The danger of rocks was not perceived due to planning route on the large scale charts.</li> <li>8. Informal voyage procedure reduced the formal attitude.</li> </ol>
<b>5<sup>th</sup> Error</b>	<u>Bridge Team Management</u>  <i>Senior Officer Action: No challenge on Captain's Decision.</i> <i>Junior Officer Action: No challenge on Captain and Senior Officer decision.</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>9. Captain adopted informal procedure.</li> </ol>
<b>6<sup>th</sup> Error</b>	<u>Ship Handling</u>  <i>Faulty execution of Captain's commands.</i>  <u>Because:</u> <ol style="list-style-type: none"> <li>10. Helmsman could not clearly understand Captain's Command.</li> </ol>

When the Captain took over the control from SOOW, valuable time was lost. Within that very short span of time, the ship crossed safety contour from 0.5 Nautical mile to 0.28 nautical mile. During this period, the captain was giving verbal orders to the Helmsman but due to language barrier, the Helmsman could not execute the commanded orders accurately in time. Therefore, JOOW came to assist in translating the orders. At this point of voyage, it was very crucial not to make any errors while executing the navigational orders. Yet, the Helmsman misunderstood some of the orders and it was too late to correct it. This was the final error. Hence, a series of perceptions-actions of ship crew resulted in an accident.

### **2.3 Accident of MV Bright Field**

The accident of MV Bright Field took place shortly after 1400 hrs on 14<sup>th</sup> December 1996 (National Transportation Safety Board, 1998). The fully loaded Liberian bulk carrier temporarily lost propulsion power as the vessel was navigating outbound in the Lower Mississippi River at New Orleans, Louisiana. The vessel struck a wharf adjacent to a populated commercial area that included a shopping mall, a condominium parking garage, and a hotel. No fatalities resulted from the accident, and no one aboard the Bright Field was injured; however, four serious injuries and fifty eight minor injuries were sustained during evacuations of shore facilities, a gaming vessel, and an excursion vessel located near the impact area. Total property damages to the Bright Field and to shore side facilities were estimated at about twenty million dollars (NASA Safety Center, 2010). A schematic diagram of the ships final path and the surrounding location is shown in Figure 2.2, which gives an overall idea on the accident site. Also using the information available



from the final six minutes before the accident a time history of events with logical description can be constructed as shown in Table 2.2.

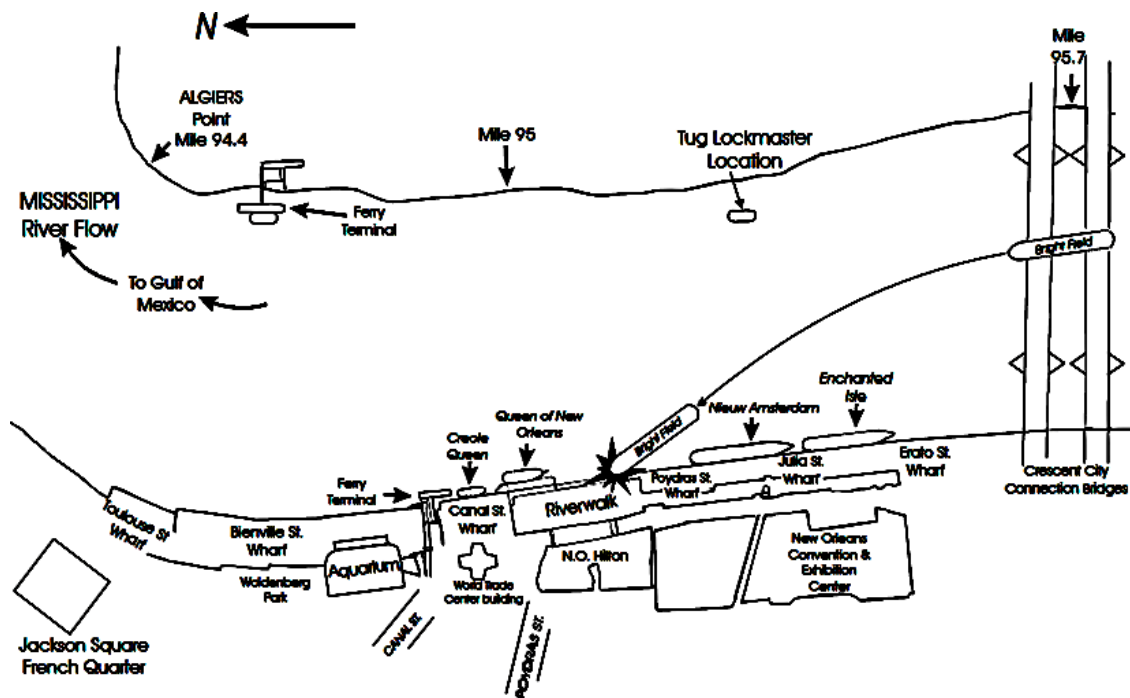


Figure 2.2: Allision of MV Bright Field (National Transportation Safety Board, 1998).

According to the investigation report (NASA Safety Center, 2010) it was found that the ship had problems with its engine lube oil system prior to few days of the accident. On the open sea, in good weather, temporary malfunctions in the vessel's main engine may be tolerable; however, in the close quarters of the Mississippi River, where safe maneuvering is directly dependent upon a responsive main engine, a loss of power can, as it did in this instance, present an immediate threat to other vessels and to shore side facilities. Hence, a combination of engine failure and series of wrong perception-actions of the crew resulted the accident of MV Bright Field.

Table 2.2: Events occurred prior to the accident of MV Bright Field.

Comments	Time	Person	Observation/ Activity/Decision	Situation
Without Engine Power (3 Minutes till impact)	1406		Engine power drops.	Bright Field passing under Crescent City Connection Bridge.
	1406+	Master	Asks his mate to call engine room and demand an increase in power.	
	1406+	Chief Engineer	Thinks except for the low rpm everything is normal.	He possibly thinks the low rpm is from the bridge control.
	1406+	Second Mate	The second mate calls the Chief Engineer and demands increase power. But he doesn't relay the information of ship's heading and maneuvering situation to the Chief Engineer.	It seems the danger of collision or allision is not comprehended. Perhaps both the Master and the Second Mate thought the engine power would be back soon.
	1406+	Chief Engineer	As the Chief Engineer does not perceive any danger, he suggests transfer of engine control from wheelhouse to engine control room as a usual practice.	
	1406+	Master	As he does not know about the particular cause of the problem, The Master agrees to transfer the control to the engine room.	This decision seems right in the sense that previously the engine showed starting problem and it was started from the engine room.
	Waste of valuable time: This transfer of control takes usually 20-30 seconds and must be completed before engine stopped. As soon as the lube oil pressure reached desired state, the engine could have been operable from the engine room.			
The Allision	1407+	Chief Engineer	The Chief Engineer could have increased engine rpm at this stage.	But the Master cannot determine his course of action. Due to language barrier, he was not fluent with the pilot who was navigating the ship.
	1411	Engine power came back on 1408. However, the crew realized very late that allision is inevitable. The port bow of Bright Field strikes a wharf adjacent to a populated commercial area including a shopping mall, a condominium parking garage and a hotel.		

## 2.4 Accident of MV Planet V

The accident of MV Planet V took place on 26<sup>th</sup> May 2012 at the Westerschelde, The Netherlands. The motor vessel lost its engine power and collided with a towed (by tug MTS vantage) pontoon while an Able-Bodied seaman (AB) lost his life trying to reduce the ship speed by dropping anchor (Dutch Safety Board, 2013). Figure 2.3 shows a snapshot from the wheelhouse of the tug taken just moments before the collision between MV Planet V and the pontoon. Table 2.3 shows a list of major events in terms of crew perception and crew action that took place prior to the occurrence of the accident (Dutch Safety Board, 2013). The final path of MV Planet V and MTS Vantage is also shown in Figure 2.4 (Dutch Safety Board, 2013).



Figure 2.3: A Snapshot of MV Planet V very close to the pontoon of MTS Vantage  
(Dutch Safety Board, 2013).

Table 2.3: Final events before the MV Planet V accident.

Time	Event
16:30	The Chief Officer carried out a routine test of the navigation systems on the bridge deck. Nothing unusual observed.
Next 40 mins	Voyage preparation was made using a Voyage Plan ( <i>Before departing for sea, the captain has to draw up a voyage preparation document, which is referred to as Voyage Plan</i> ).
17:10	A tugboat MTS Vantage leaves for its destination with its pontoon tow.
Next 8 mins	The Pilot of the MTS Vantage contacts the Pilot of MV Planet V by VHF to inform about the tugs intentions.
17:18	Main engine of MV Planet V is started.
Next 6 mins	At this time two auxiliary engines for the auxiliary generators were running. The shaft generator was also running which was used to provide power for the bow thruster.
17:24	The ship departs the harbor.
Next 17 mins	The Captain informed the engine room crew that the bow thruster was no longer required. The Chief Engineer, therefore, shut down the auxiliary engines and used the shaft generator for necessary power.
17:41	MTS Vantage passes the Sloehaven harbor entrance with a speed of 6 knots.
17:45	MV Planet V passes the harbor entrance. The speed was 11 knots.
17:48	MV Planet V is along the starboard side of the pontoon.
17:48:23	The main engine of MV Planet V fails. Immediately the electrical systems onboard failed and the ship went into total blackout.
Next 16 seconds	The ship started to turn port after the electrical failure.
	The crew and the Pilot observed that the rudder angle indicator showed starboard rudder angle.
	The Pilot of MV Planet V informs the Pilot of MTS Vantage about the situation and requests 'full speed ahead' for the tug to prevent collision.
17:48:39	The Captain of Planet V instructs AB to return to forecandle, and prepare the anchor.
17:49:34	The Captain orders to drop the anchor via VHF. The pilot was not consulted with about this. The intention of the Captain is to slow down the ship and accelerate its turn to the port in an attempt to pass the tug and the tow at its stern.
Next 21 seconds	The tug started increasing speed and turning to port in an attempt to increase its distance from MV Planet V.
	The Captain orders AB not to run out of chain any further.
	AB tightens the anchor winch brake. Despite this the anchor chain continues to run out at high speed.
	To apply additional force AB climbed onto the electrical motor of anchor winch.
17:50:05	MV Planet V hits the pontoon amidships on its starboard side.
	After collision, MV Planet V moved along the pontoon while the anchor chain continued to run out. The end of the chain flew out of the sparling pipe and fell overboard.  AB standing on the electric motor was hit and fatally injured by the anchor chain.

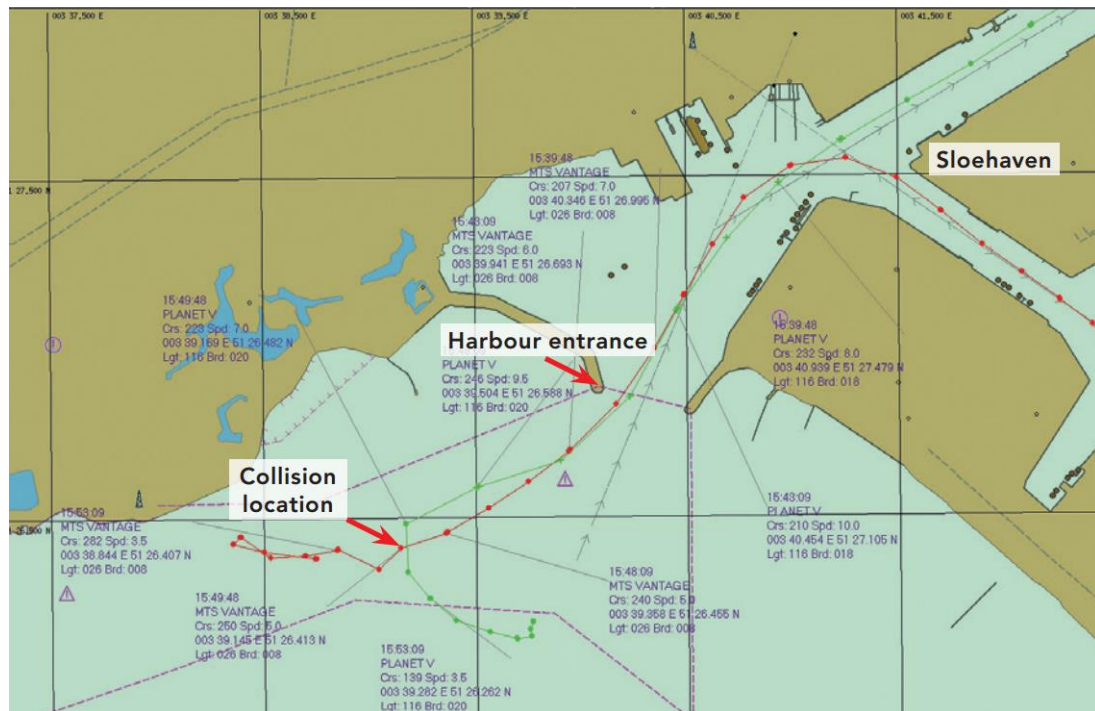


Figure 2.4: The final path of MV Planet V (green line) and MTS Vantage (red line)

(Dutch Safety Board, 2013).

The tragic incident of MV Plate V was preventable and the same goes with the cases of MV Costa Concordia and MV Bright Field. The hindsight of these accidents establishes that if the crew made alternate decisions at the right time the accidents could have been avoided. For example, the Costa Concordia accident could have been prevented if the helmsman could have executed the command in time or the other crewmembers avoided the errors. The allision of MV bright field could have been prevented if the Chief Engineer knew about the danger ahead and took emergency restart of the engine. In the case of MV Planet V, if the auxiliary generators were kept running then the bow thruster could have been used to avoid the collision and the AB could have saved his life by avoiding the emergency anchor maneuver or standing in a different spot. Therefore, the point of argument is that to prevent an accident it is important to comprehend ‘how’ an

accident may evolve and ‘how’ the perception-action sequence of the crew result in an accident.

## **2.5 Chapter Summary**

This chapter showed how maritime accidents unfold in timeline. Three major accidents are studied. It is seen that every decision that results in an action can be reasoned back to some justified events or situations. The study reveals that maritime accidents can occur due to complex interactions between individual professionals. These interactions often seem useful or harmless which eventually masks the necessary and sufficient causes of accidents. Therefore, predicting maritime accidents become difficult.

## **Chapter 3: Review of Accident Theories and Models**

### **3.1 Introduction**

This chapter reviews some of the major accident theories and models. The objective is to comprehend and compare different accident theories and models. It is, therefore, vital to understand the science and essence of each accident theory/model. In the review, three questions were kept in mind while studying the theories/models:

- i. What type of theory/model/hypothesis/method/work is it?
- ii. What is the context under which this work belongs?
- iii. What is the cause(s) of accident?

In order to make the discussions simple and comprehensible, timelines are utilized wherever possible. The following sections describes the above in detail.

### **3.2 Classification of Accident Theories and Models**

A theory is a collection of propositions to illustrate the principles of a target subject. On the other hand, a model is a simplified description, especially a mathematical one, of a system or process, to assist calculations and presentations. Therefore, an accident theory is a collection of propositions that illustrate the principles of accident causation and an accident model is a simplified description of accident occurrence that is based on an accident theory.

Different researchers classified different types of accident theories/models. Six research works were found which constructed comparable review on accident theories, models and investigation techniques. Figure 3.1 shows the review works in timeline. Benner (1985) conducted an extensive review and found that accident models and investigation methodologies are quite diverse. He suggested that more exhaustive research into the measurement and rating of accident models and accident investigative methodologies is required.

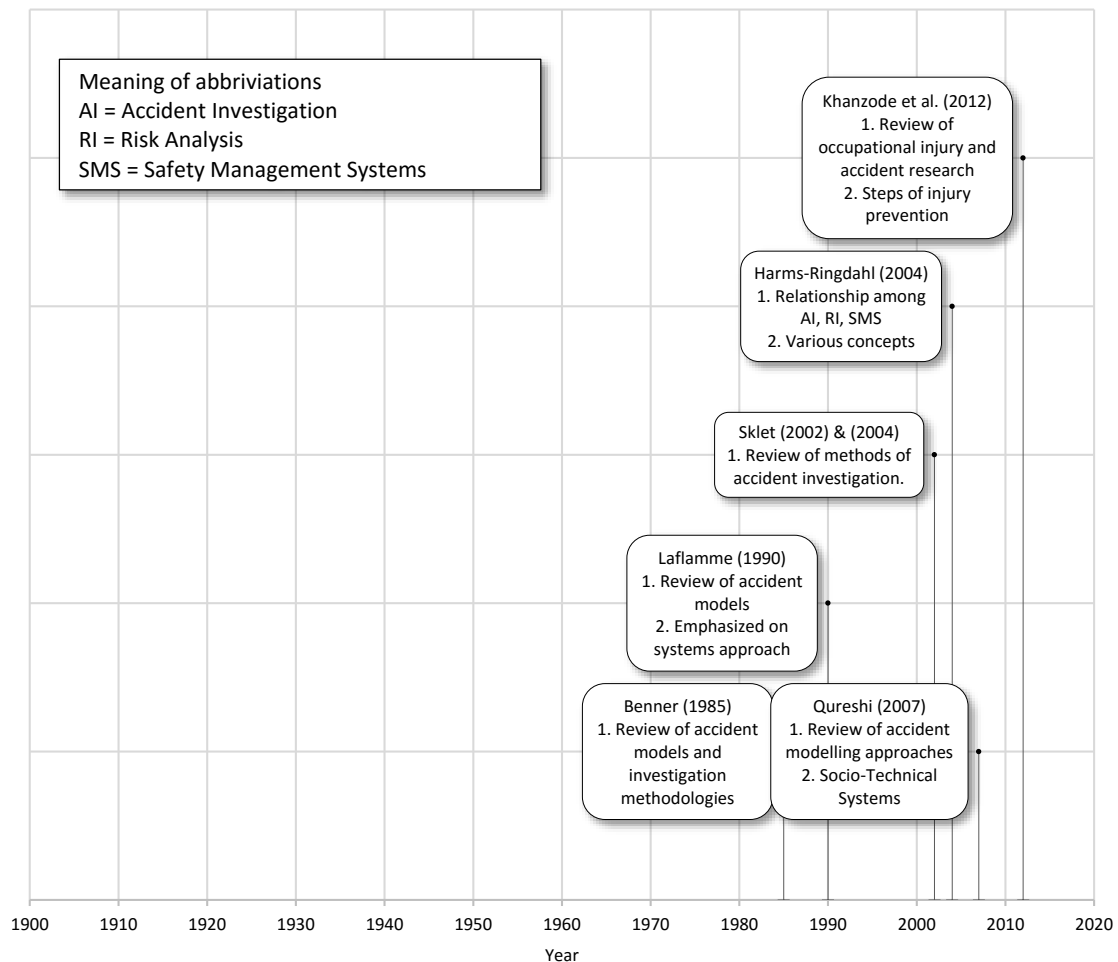


Figure 3.1: Review of accident theories and models by different researchers in timeline.



Laflamme (1990) studied different occupational accident models and found four different types of accident models:

- i. Decision model
- ii. Sequential model
- iii. Energetic and sequential model and
- iv. Organizational models

However, Laflamme emphasized on further studies for the development of systems approach for accident studies. One of several reasons is that systems approach enhances that dynamic and interactive nature of the phenomena. Another advantage is that systems approach has the merit of putting aside guilt or blame. Instead, it draws attention towards many different ways of adjusting and correcting non-optimal work-situations.

Sklet (2002, 2004) reviewed different methods of accident investigation. The studies argue that during an investigation process, different methods might be used in order to analyze arising problem areas. Among a multi-disciplinary investigation team, there should be at least one member having good knowledge about the different accident investigation methods, being able to choose the proper methods for analyzing the different problems. Just like the technicians have to choose the right tool in order to repair a technical system, an accident investigator has to choose proper methods analyzing different problem areas.

Harms-Ringdahl (2004) reviewed and compared the relationship among accident investigation (AI), risk analysis (RA) and safety management systems (SMS), which clearly established the differences and similarities among these three subjects.

Qureshi (2007) made extensive review and classified accident modelling approaches. According to Qureshi there are four main approaches to accident modelling:

- i. Traditional approaches to accident modelling
- ii. Systemic Accident Models
- iii. Formal Methods and accident analysis
- iv. Sociological and Organizational Analysis

At first the traditional approaches to accident modeling which include Sequential Accident Models (e.g Domino theory), chain of time ordered events models (Multi-linear event sequencing models), risk analysis models, and epidemiological accident models. The second type of models are Systemic Accident Models. It includes system theoretic approach, cognitive systems engineering approach, Rasmussen's socio-technical framework, AcciMap accident analysis technique and System Theoretic Accident Model and Process (STAMP) approach. The third type is the Formal Methods and Accident Analysis which include Logic formalisms to support accident analysis, Why-Because Analysis (WBA), Probabilistic models of causality. The fourth type is the Sociological and Organizational Analysis of Accident Causation. This type of analysis originated from recent disasters like the Bhopal and Challenger accidents which involved decisions of group of people and social behavior. It also discusses how normalizing social deviance can cause accidents.

Khanzode et al. (2012) reviewed occupational injury and accident research where he classified accident models in Chronological order. According to Khanzode there are four generation of accident models:

- i. First Generation: Accident proneness
- ii. Second Generation: Domino theories
- iii. Third Generation: Injury epidemiology
- iv. Fourth Generation: System models

First generation theories hold a primitive viewpoint towards accident causation. These theories hold a person's traits and unsafe behavior as responsible for accident. Second generation theories (domino theories) conceptualize a chain of sequential events leading to an accident, and call these events as dominos. Removal of any one domino from the chain would break the chain of accident events. Injury epidemiology models (originated in 1960s) represent third generation of accident research. Injury epidemiology approach holds that accident prevention efforts do not necessarily lead to injury control in a work system. This approach focuses on energy transfer involved in injury incident, and tries to minimize it in order to minimize the losses. System approach to accident causation (the fourth generation) emerged in 1970s as a response to the challenge of maintaining safety in increasingly complex work systems.

In an attempt to review accident theories, Awal and Hasegawa (2015a) made a simple classification of accident models which is shown in Figure 3.2. According to this

study there are three types of accident models: (i) Sequential Accident Models, (ii) Epidemiologic Accident Models and (iii) Systemic Accident models.

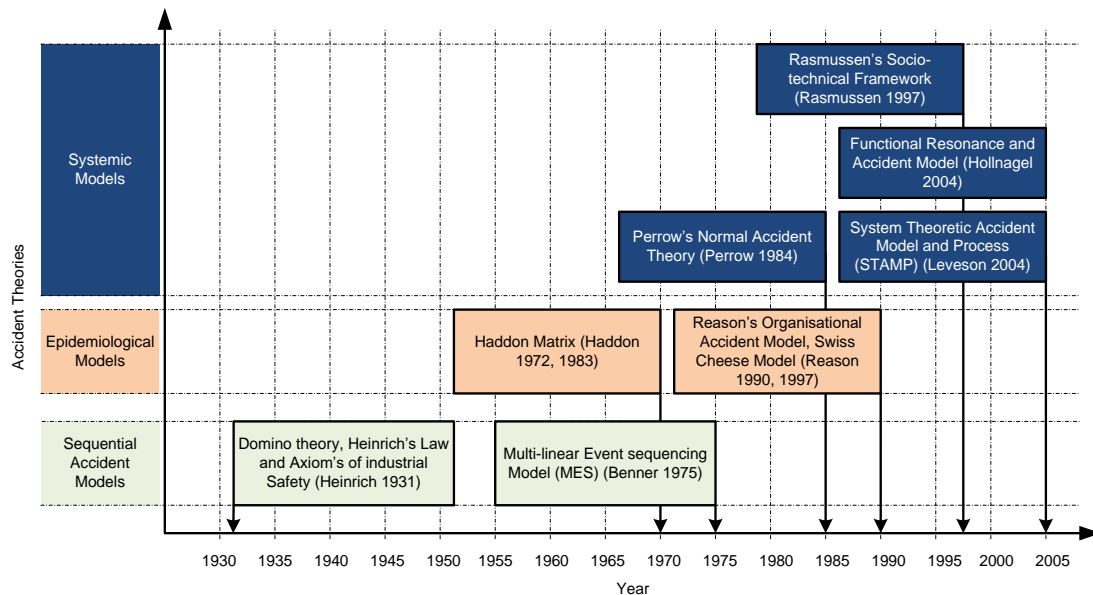


Figure 3.2: Major accident theories and their chronological appearance (Awal and Hasegawa, 2015a).

In this thesis this classification has been elaborated further. The accident theories, models, investigation methods etc. are classified into eight groups. A summary of the literature review is portrayed in Figure 3.3 where the timeline indicates the publication year of the papers and number of papers studied within each specific group. The years indicate a range of publication where the preceding year indicates the first publication and the later indicates the recent publication that has been read. The following sections elaborates this classification and distinguishes their respective characteristics, merits and demerits. It is pertinent to mention that Group 6 Education for accident prevention is not included in this thesis since it falls beyond the scope of current research work.

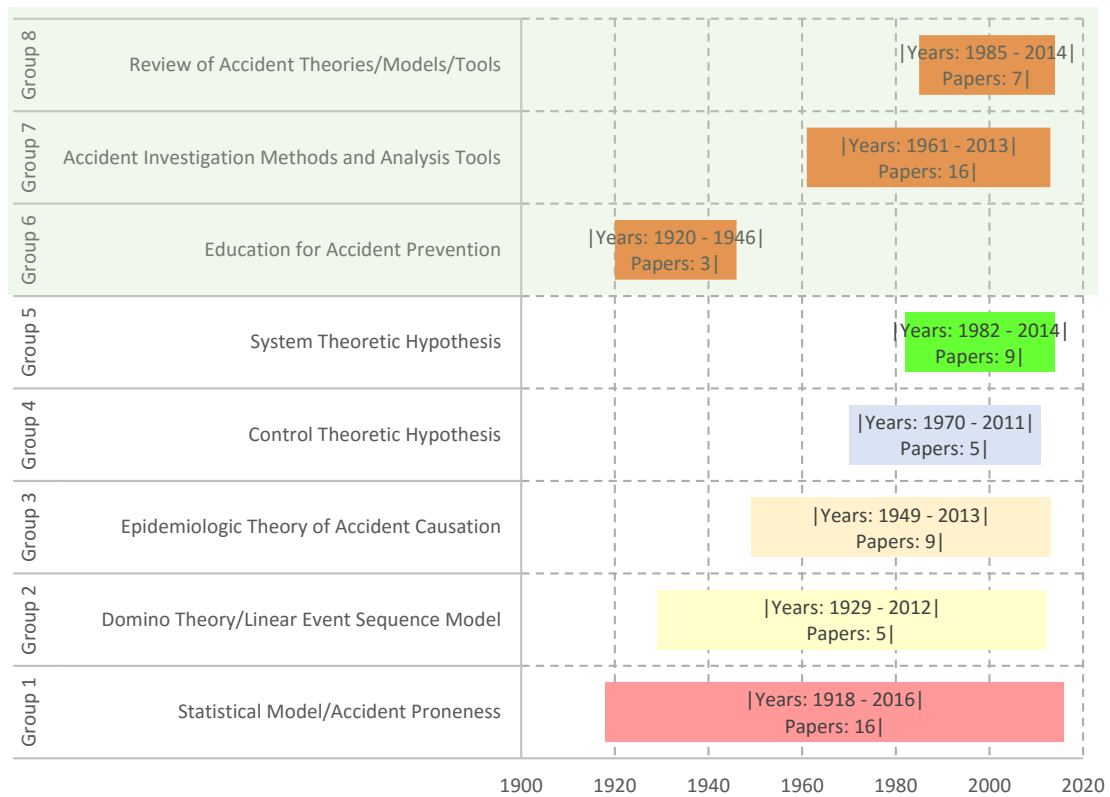


Figure 3.3: Classification of accident theories/models/etc in timeline.

### 3.3 Accident Proneness and Statistical Models

In this group fifteen research works are considered which are shown in Figure 3.4. Vernon (1918) is perhaps the one of the earliest researcher in modern accident science who investigated the accident causations mathematically in light of industrial context. Vernon studied industrial accidents in United Kingdom in relation to the following:

- i. Speed of production
- ii. Fatigue
- iii. Physical influence and alcohol consumption
- iv. Nutrition

v. Natural and artificial lighting

vi. Temperature

Vernon's study concluded that accidents are very largely due to carelessness and inattention, so they could be diminished by preventing the workers from talking to one another in the shops. Kossoris (1939) also studied industrial accidents and concluded that unsafe conditions and unsafe practices generate accident causation factors.

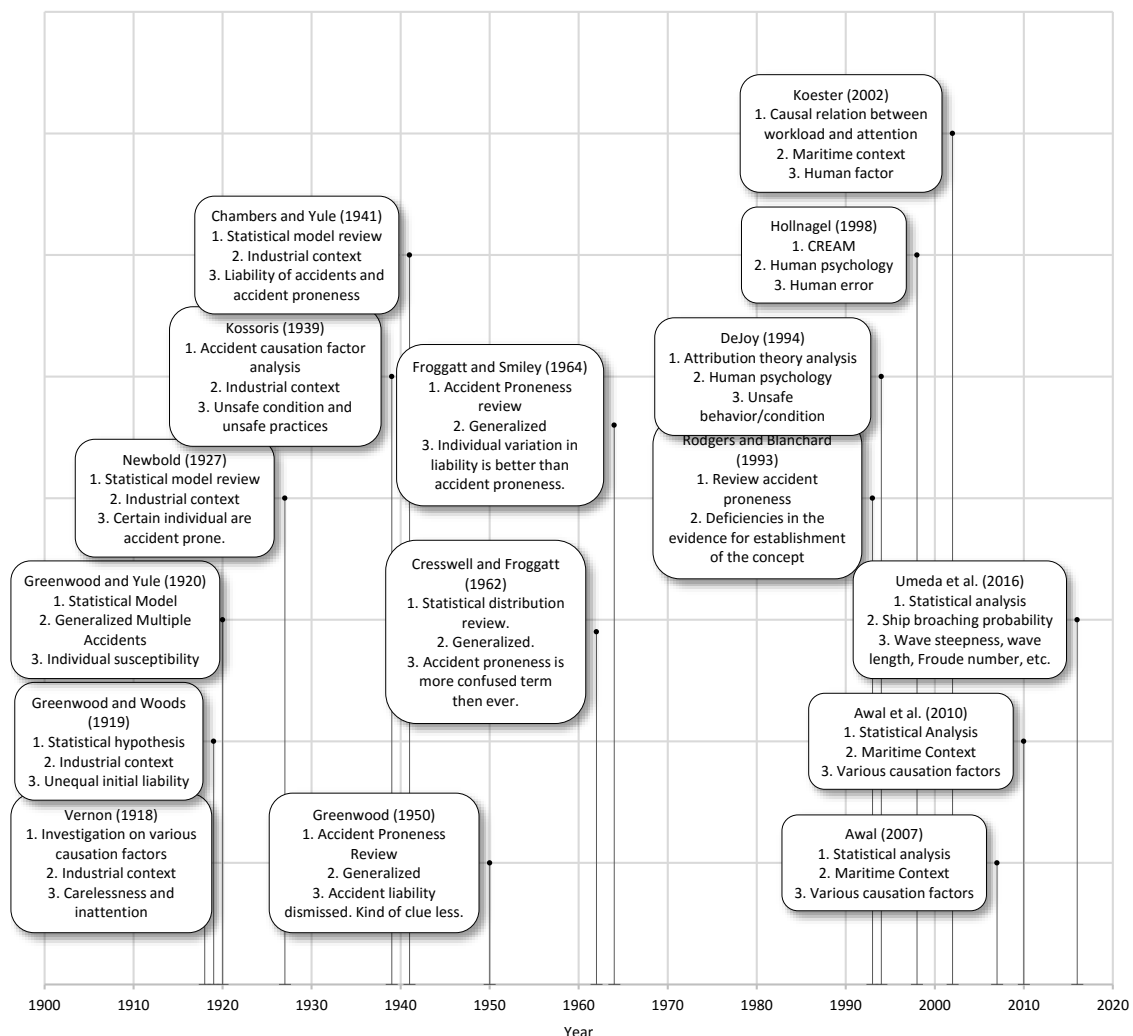


Figure 3.4: Research works on accident proneness and statistical models.

An empirical research by Greenwood and Woods in 1919 established some statistical hypothesis about accidents in the industrial context. Their studies suggested some strong grounds for thinking that the bulk of the accidents occur to a limited number of individuals who have special susceptibility to accidents, and that the explanation of this susceptibility is to be found in the personality of the individual. The aim is to consider and investigate the relations of the hours of labor and of other conditions of employment including methods of work, to the production of fatigue and so on. The study utilized some statistical distributions such as:

- i. Simple Chance Distribution (SD)
- ii. Biased Distribution (BD)
- iii. Distribution of unequal liabilities (UD)

Similarly, Greenwood and Yule (1920) studied the nature of frequency distributions of accidents.

Newbold (1927) studied accidents statistically in industrial context and concluded that certain individuals are accident prone. Chambers and Yule (1941) also support such study. According to Cresswell and Froggatt (1962) and Froggatt and Smiley (1964), the designation of accident prone implies, irrespective of environment, that individual is more likely at all times to incur an accident than his colleagues even though exposed to equal risk, and that this is due to some characteristics or summation of some characteristics associated with corporal dexterity, sensory-motor skill, personality, or higher cognitive function. In short, accident proneness is conceived as an immutable load to which the unfortunate possessor is chained like some Ixion to his wheel. The corollary is, that the

population can be dichotomized on the basis of the possession or non-possession of the characteristic, or at least ranked in terms of its severity.

However, Greenwood (1950) reviewed the concept of accident proneness and the idea of accident liability. Cresswell and Froggatt (1962) also supported this study of Greenwood (1950) and concluded that individual variation in liability is better than accident proneness. Rodgers and Blanchard (1993) reviewed accident proneness and identified the deficiencies in the evidence for establishment of the concept of accident proneness.

DeJoy (1994) discussed the attribution theory analysis and within the human psychology context where he concluded that unsafe behavior/condition is the reason of accident occurrence. Hollnagel (1998) proposed the renowned Cognitive Reliability and Error Analysis Method (CREAM) which utilized the science of human cognition. The premise of this method is that accidents occur due to human error hence by preventing human errors accidents can be prevented. Koester (2002) studied the causal relation between workload and attention within the maritime context and identified the human factor that contribute towards accidents.

Perhaps one of the most acclaimed branch of statistical analysis of accidents is risk analysis, which gained a popularity after the introduction of Fault Tree Analysis (Watson 1961 and Erricson, 1999). Some examples can be found in Hossain et al. (2010, 2014). Also Awal (2007) and Awal et al. (2010) utilized statistical analysis of maritime accidents in Bangladesh where the authors identified various accident causation factors.



Similarly, Umeda et al. (2016) studied ship broaching probability due to wave steepness, wavelength, Froude number and other parameters. One may ask the question why risk analysis is so popular in real engineering problems and academic exercises? The answer perhaps can be found in the technical facility of risk analysis and fault tree analysis rather than the theoretical justification. The most attractive feature of such analysis is undoubtedly the wide range technical applicability. The literature of Li et al. (2012) on quantitative maritime risk assessment can be referred and considered as a foundation of in this regard.

### **3.4 Domino Theory**

Heinrich (1931) proposed that accident occurs from chain of events after conducting studies on statistical accident analysis (Heinrich, 1929). Heinrich elaborated that the individual fault can be related to other factors in sequence, just like a domino. Heinrich explains that undesirable personality traits can be passed along through inheritance or develop from person's social environment and both inheritance and environment contribute to faults of person. This can be considered as the first domino. The second domino deals with worker personality traits. Heinrich explains that inborn or obtained character flaws contribute to accident causation. According to Heinrich (1931), natural or environmental flaws in the worker's family or life cause these secondary personal defects, which are themselves contributors to unsafe acts or the existence of unsafe conditions. The third domino is the direct cause of incidents. All incidents directly relate to unsafe conditions or acts, which Heinrich defines as, "unsafe performance of persons ... and removal of safeguards". However, Heinrich defines four reasons why people commit unsafe acts:

- i. Improper attitude
- ii. Lack of knowledge or skill
- iii. Physical unsuitability
- iv. Improper mechanical or physical environment

Heinrich later goes on subdivide these categories into “direct” and “underlying” causes and concludes that combination of multiple causes create a systematic chain of events that leads to accident. Figure 3.5 shows the five research works of domino theory in timeline.

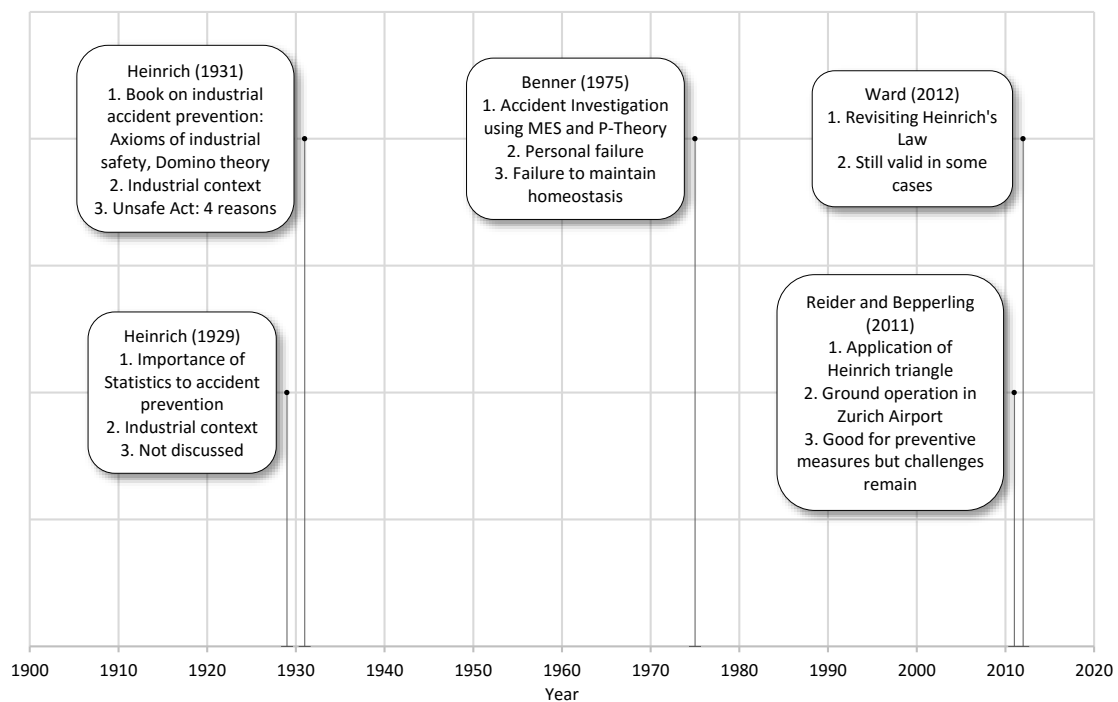


Figure 3.5: Research works related to domino theory in timeline.

The work of Heinrich gained wide acceptance in the early part of twentieth century and influenced other researchers to propose linear event sequence accident models. An example can be Benner (1975) who proposed multilinear event sequencing

methods and P-Theory for accident causation. Benner's proposal states that during an activity events may occur in series or parallel. Perturbation may try to break the homeostasis and the actors involved in the activity may fail to maintain the homeostasis. Then the accident may take place. Although the linear event sequence idea proved inapplicable in some cases yet Reider and Bepperling (2011) found its application useful in Ground operation in Zurich Airport. They concluded that this approach is good for preventive measures but there remain challenges to overcome. Ward (2012) revisited Heinrich's Law and concluded the same.

### **3.5 Epidemiologic Theory**

Gordon (1949) is the proposer of epidemiologic theory of accidents. Gordon considers accidents as an ecologic problem. In this research he states that the causative factors in accidents have been seen to reside in agents, in the host and in the environment. The mechanism of accident production is that process by which the three components interact to produce a result, the accident. Therefore, the cause of accident comes from the interaction between the host, agent and environment.

The hypothesis is if home accidents are primarily a public health problem then the problem is reasonably to be approached in the manner and through techniques that have proved useful for other mass disease problems. This includes first an epidemiologic analysis of particular situation, an establishment of causes, the development of specific preventive measures directed towards those causes, and finally a periodic evaluation of accomplishment from the program instituted. Figure 3.6 shows the accident studies based on the epidemiologic concept.



Figure 3.6: Accident studies in timeline based on epidemiologic concept.

Haddon in 1968 proposed a 2D dimensional matrix for injury control which was mainly developed for car crash. Such matrix helps to identify the host, agent and environmental factors in temporal order (present, during event and post event) and helps identifying preventive measures. Haddon later in 1970 gave ten strategies for reducing

losses which were based on the energy transfer concept. The reason behind this energy transfer concept is because that major class of ecologic phenomena involves the transfer of energy.

Reason (1990) proposed that there are latent human failures which cause accidents without any visible causes. The proposal included the following three concepts:

- i. Latent failure
- ii. Local triggering event
- iii. System defenses

The study discussed the differences between active and latent human failure and a framework for the dynamics of accident causation. Reason (1997) developed the organizational Accident Model or the Swiss cheese model. Organizational accidents occur within modern complex technological societies having multiple causes. The accidents involve many people operating at different levels of their respective companies. Hazard cause losses and barriers are there to prevent. Like a Swiss Cheese there are holes in the barriers. When all the holes align then hazards pass through and cause losses. Each barrier represent each level of organizational defenses against losses.

Woods et al.s' report (1994) on human error studied the interaction between man and machine and attempted to relate active failure and latent failure to accident phenomena. Loimer et al.'s review (1996) suggested that some accidents may be unstoppable yet injury could be manageable. Schröder-Hinrichs et al. (2012) compared the accidents of RMS Titanic and MV Costa Concordia and highlighted hundred years of lessons not learned. The authors further discussed how human and organizational factors

still remain in maritime industry. Recently, Underwood and Waterson (2013) reviewed the Australian Transportation Safety Board (ATSB) model which is based on Swiss Cheese Model (SCM) that provided a general framework that can guide data collection and analysis activities during an investigation.

### **3.6 Control Theoretic Hypothesis**

Suchman (1970) proposed the social deviance hypothesis for accident causation. In this study an accident is considered as a social problem and rejection of social constraint is considered as the cause behind accident. Kjellen and Larsson (1981) and Kjellen (1984a, 1984b) proposed Deviation Concept for occupational accident control. The proposal stated that deviation from norm causes accidents in production process. According to the studies there are two levels of analysis: (i) Accident sequence and (ii) determining factors. Their model may be considered as a tool for practical investigation of accidents. More precisely, occupational accidents within the industrial company.

Tuominen and Saari's (1982) method of accident analysis states that linear event sequence occur in temporal order. A work accident is a state of disturbance in an organized dynamic system of man and his technical environment. Accident occurs when uncontrolled contact takes place between a person and an injuring factor.

Rasmussen (1997) discussed the risk management in dynamic society and proposed AcciMap for accident investigation. Rasmussen's work fundamentally considers accidents as control problem which to some extent resembles with control engineering structures. Vicente and Christoffersen (2006) tested the Rasmussen's

framework on E. Coli outbreak. Branford (2011) applied AcciMap and studied loss of control at various levels of the Überlingen mid-air collision. Figure 3.7 shows some of the control theoretic studies of accident causation in time line.

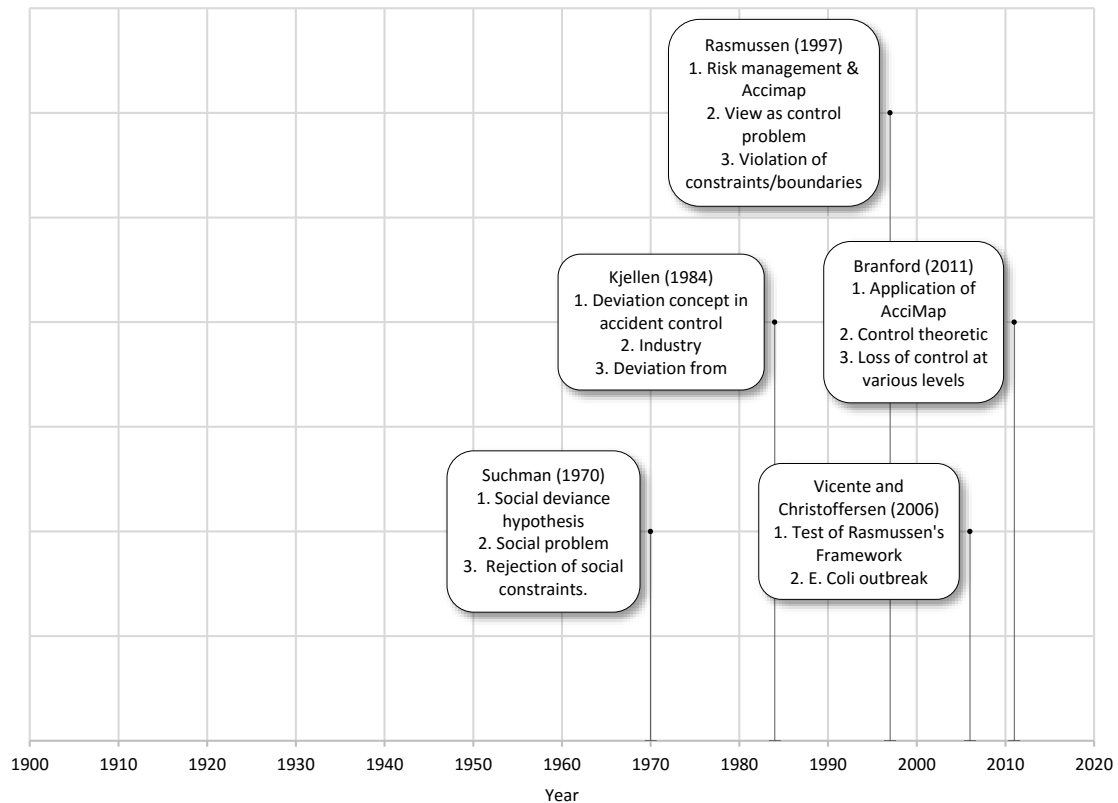


Figure 3.7: Control theoretic accident models in timeline.

### 3.7 System Theoretic Hypothesis

Perrow (1984) proposed the term ‘Normal Accident’ which became very popular in various high tech industries. According to Perrow, normal accident is a characteristic of a system. Given the characteristics of the system, multiple and unexpected interactions of failures are inevitable. The interactive complexity and tight coupling – the system

characteristic – inevitable produces an accident; therefore, it is called ‘Normal Accident’ or ‘System Accident’. The premises of this idea are:

- i. People make mistake
- ii. Big accidents begin from small beginnings
- iii. Many failures originate due to organization/technology

Tuominen and Saari (1982) proposed a linear analysis model based on systems concept. The reason of accident was proposed to be uncontrolled contact. Collins and Thompson (1997) presented Systemic Failure Modes (SFMs) based on Normal Accident Hypothesis (NAH). They conclude that in high risk society the failures are inevitable. Figure 3.8 shows the system theoretic research works in timeline.

Hollnagel (2001) studied the performance condition's effect on individual error which was based on System condition/Human error. Hollnagel concluded that there is a need to understand the complexity of joint system performance, and the dynamics of human-machine interaction. Instead of focusing on discrete actions taking place at single points in time, one should focus on how actions develop over time. Hollnagel and Goteman (2004) proposed Functional Resonance Accident Model (FRAM). The idea suggested that accidents occur due to functional resonance within a system. Leveson (2004) introduced System Theoretic Accident Model and Process (STAMP) which is essentially a model of control system. She suggested that accidents occur when there is a lack of constraints. Sammarco (2005) applied Normal Accident Theory (NAT) to safety related computer systems. Hollnagel (2013a) applied of FRAM based on resilience engineering the study illustrated the risk assessment due to organizational changes.



Leveson (2015) studied the risk management by leading safety indicators using the STAMP based idea.

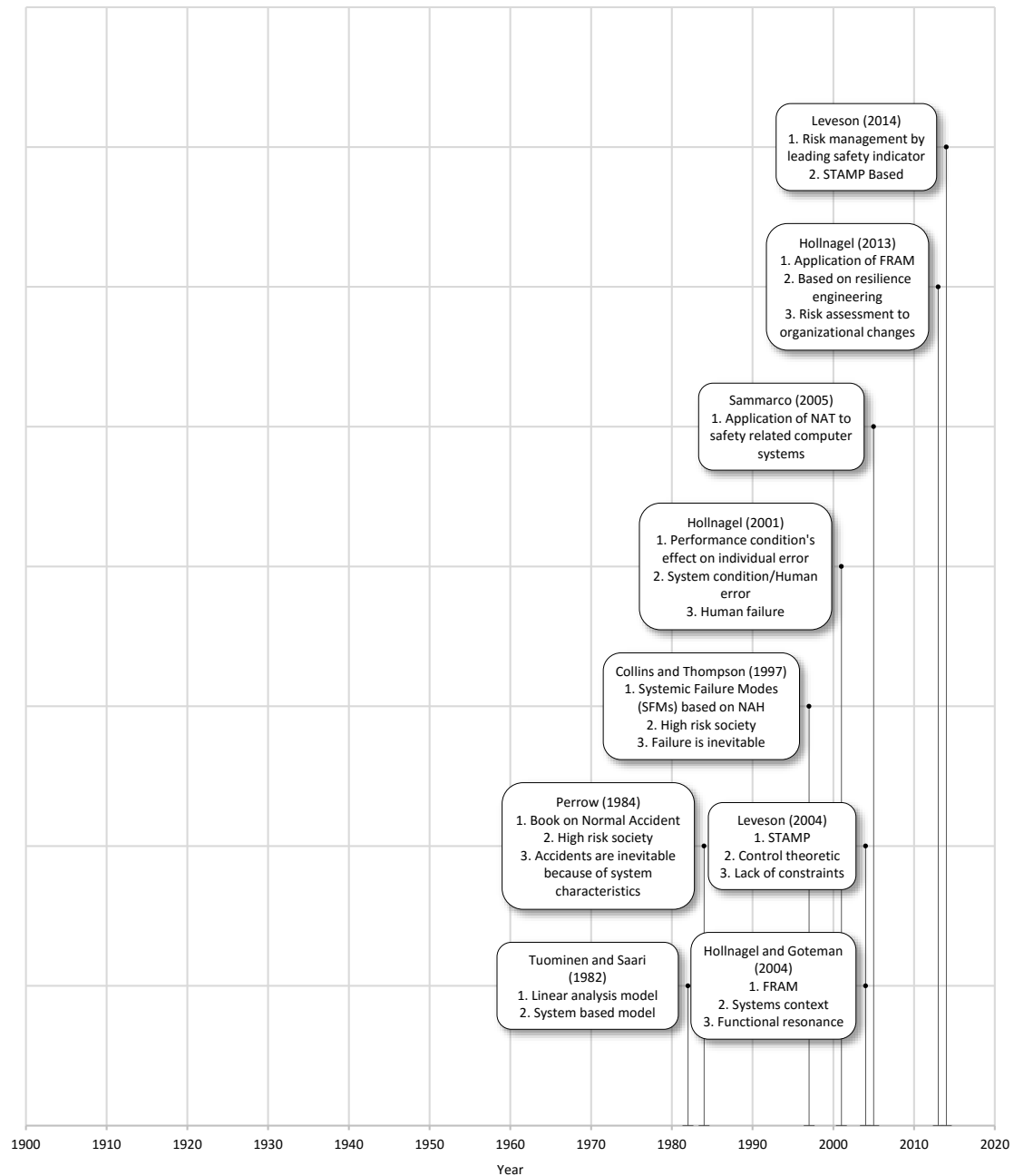


Figure 3.8: Systems theoretic research works in timeline.

### **3.8 Tools for Accident Analysis and Investigation**

One of the earliest and most successful tool for accident analysis is Fault Tree Analysis (Watson, 1961). It was originally developed for space mission which was later generalized for all purposes. A comprehensive literature review on Fault Tree Analysis can be found in Ericson (1999). Hendrick and Benner (1987) wrote a book on STEP (Sequentially Timed Events Plotting) which is an accident investigation technique. Figure 3.9 shows various accident analysis and investigation tools.

Johnson et al. (1995) showed the use of Formal Language in accident reports. Burns et al. (1997) demonstrated Sorted First Order Action Logic (SFOAL) usage in reasoning and find out the human contribution in accidents. Johnson (1997) discussed the usage of formal methods in human factor/system failure. Botting and Johnson (1998) discussed the task analysis in accident model. Burns (2000) discussed the analysis of accident reports in light of formal methods. Johnson (2000) showed the application of mathematical proof technique for accident analysis. Johnson and Holloway (2003) presented an overview of mishaps logic such as: classical logic, material condition, etc.

Rasmussen and Svedung (2000) demonstrated proactive risk management and extension of AcciMap. They argued that functional abstraction is more effective than structural decomposition in accident prevention. Vernez et al. (2003) showed potentials of colored petri nets in risk analysis and accident modelling. Harms-Ringdahl (2009) reviewed accident investigation methods and discussed about safety function and barrier.

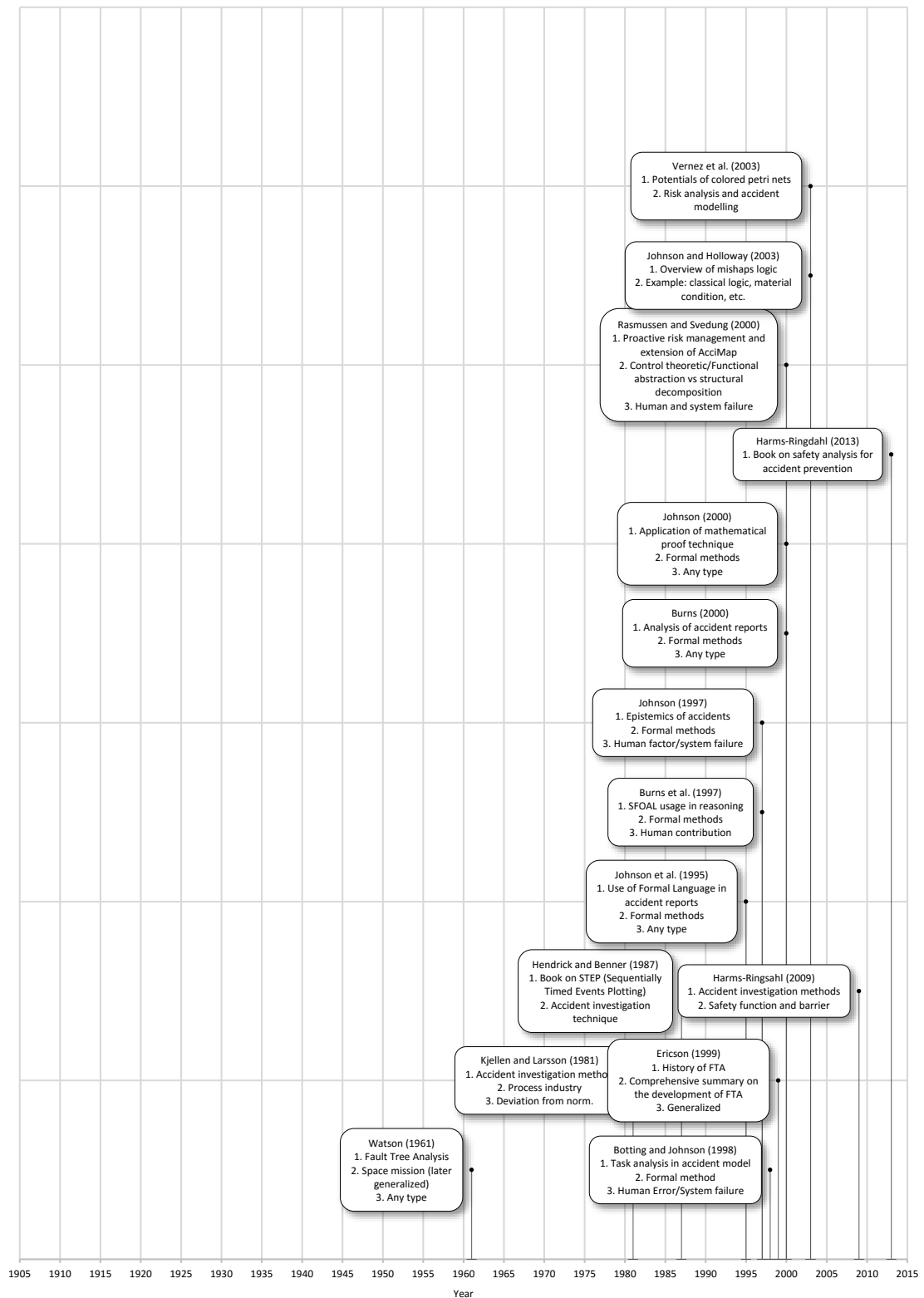


Figure 3.9: Various tools for accident analysis in timeline.

### **3.9 Chapter Summary**

Over sixty papers/books/reports on different theories and models have been studied. Most of these are journal papers that represent the first appearance of the contribution or significant development. However, this is not exhaustive; there are numerous relevant research works which could only be mentioned to increase the volume of examples.

Looking from the beginning of industrialized era – the dominating trend has been that systems become gradually more difficult to understand and control. This goes for technical systems, socio-technical systems and organizations alike. Therefore, regarding accidents, the thinking has gone from single factor models (such as ‘error proneness’), to simple linear models (such as the Domino model), to composite linear models (such as the Swiss cheese model), and to complicated multi-linear models (such as STAMP).

## **Chapter 4: Fundamentals of Logic Programming**

### **Technique (LPT)**

#### **4.1 Introduction**

This chapter discusses the fundamentals of Logic Programming Technique (LPT). Several important issues such as ‘what is logic programming?’, ‘what are the advantages of logic programming?’, ‘how logic programming is applied in accident problems?’ etc. are discussed in this chapter. At first, the classic Monkey-Banana Problem is elaborated in order to understand the concept ‘how’ in logic programming.

Before proceeding to further discussions, it is however, necessary to distinguish between Logic Programming and Logic Programming Technique (LPT) which are frequently used in this thesis. Logic Programming refers to a programming paradigm which is fundamentally a subject of computer science. On the other hand, Logic Programming Technique (LPT) is the subject matter of this thesis about which the introduction and fundamentals are proposed. Logic Programming Technique (LPT), therefore, refers to an accident analysis technique which utilizes logic programming.

#### **4.2 The Monkey-Banana Problem**

Monkey-Banana Problem is a famous problem in artificial intelligence. In this problem, in a room there is a monkey, a stool, and bananas that have been hung from the center of the ceiling of the room. The bananas are clearly out of reach from monkey (Figure 4.1).

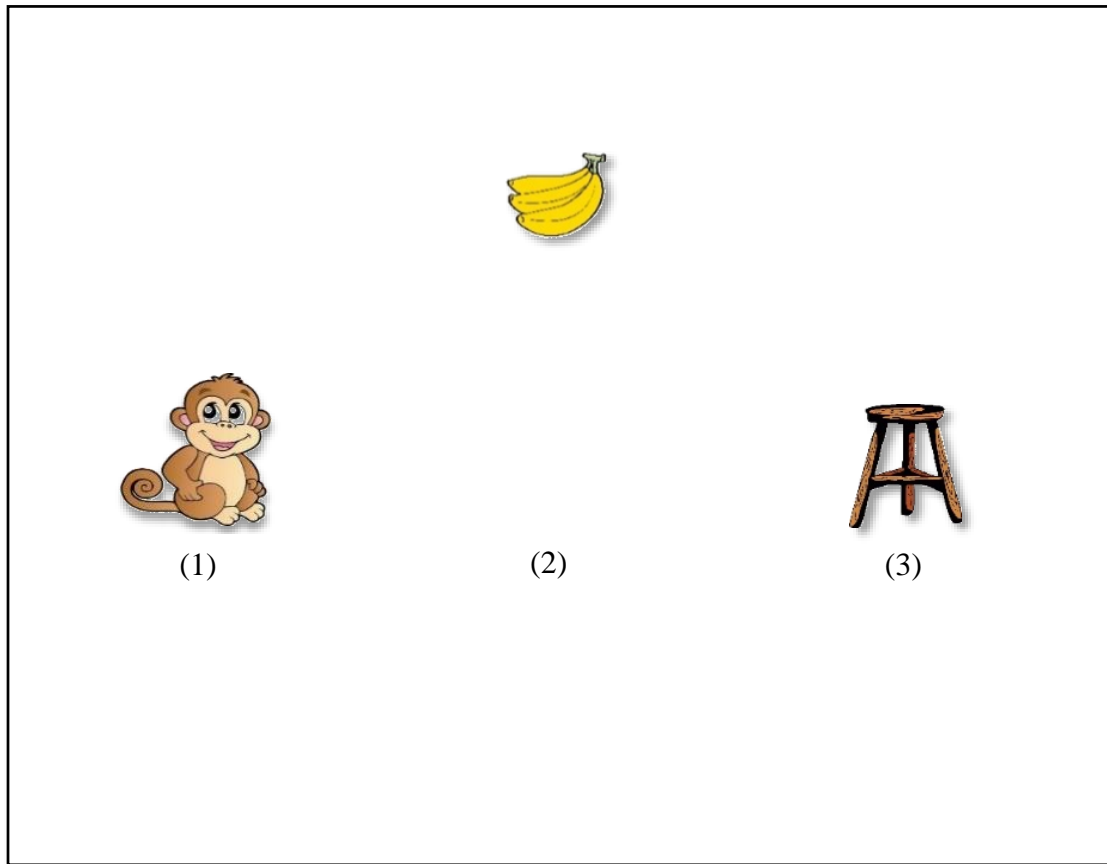


Figure 4.1: Classic monkey-banana problem: ‘how’ to get the banana?

Under this situation, the problem is ‘how’ the monkey gets the banana from the ceiling. If the monkey is clever enough, he can reach the bananas by placing the chair directly below the bananas and climbing on the top of the chair. Therefore, the solution to the problem is the ‘measures’ the monkey takes to get the banana. The measures could be like this in sequence: monkey moves to position (3) from position (1), gets the stool, moves the stool to position (2), gets on top of the stool and grabs the banana. This sequence of action is the solution to the problem that is shown in Figure 4.2.

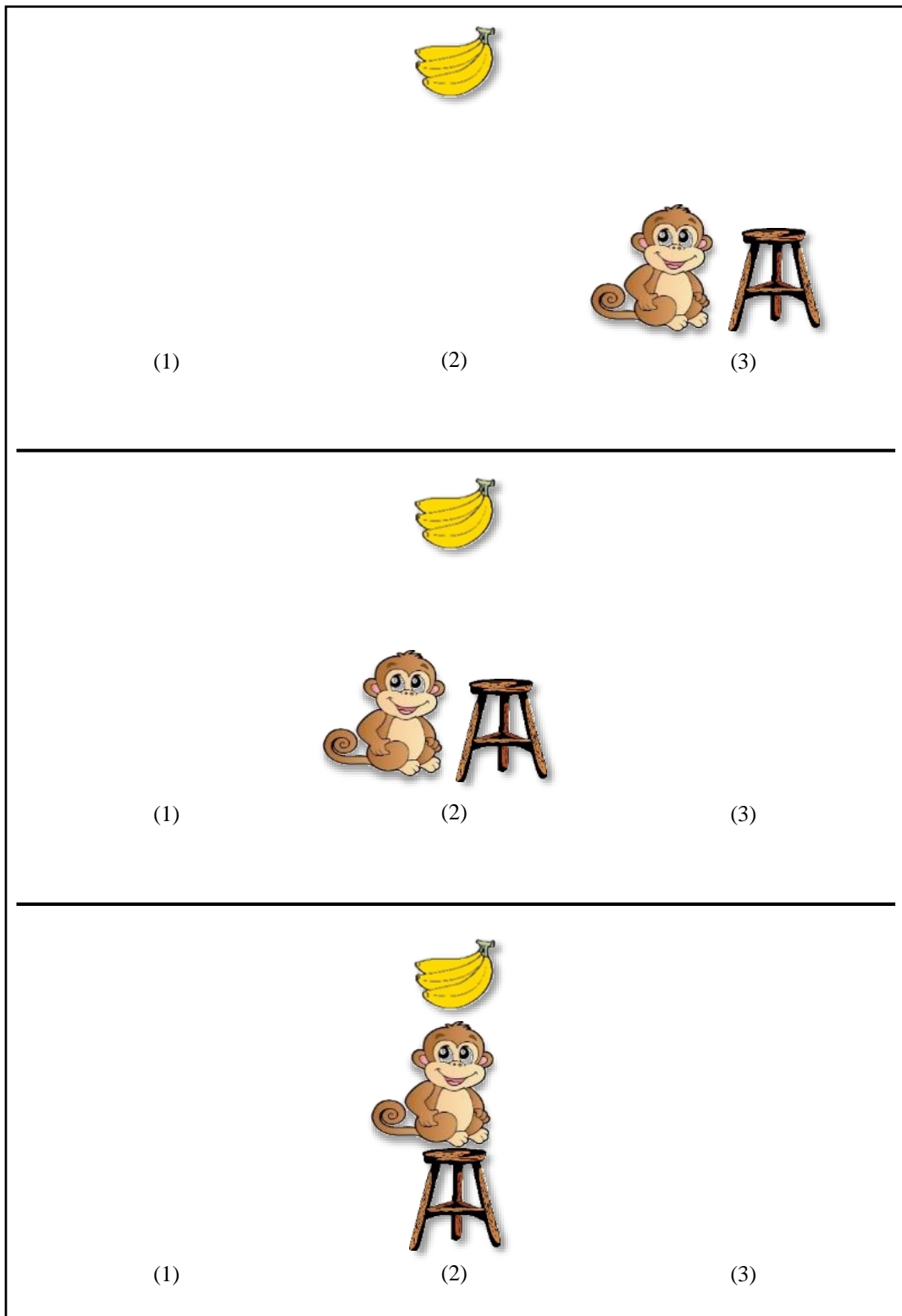


Figure 4.2: Solution of the monkey-banana problem.

In logic programming such kinds of solution is obtainable (shown in Appendix A and Appendix B). In this particular problem, the monkey is given several abilities such as:

- i. Ability to grab the banana.
- ii. Ability to move forward and backward in the horizontal direction.
- iii. Ability to push the given tool forward and backward in the horizontal direction.
- iv. Ability to climb up and climb down from the tool.

Once the abilities are given, then the monkey can be taught to use these ability in sequence. In this particular example the monkey tries to grab the banana first from position (1) as shown in Step 1 of Appendix B. This may be regarded as the monkey's first strategy. Nevertheless, it fails to grab from position (1) so he moves forward to position (2) and tries to grab. This can be regarded as second strategy. However, the monkey, in Step 2, fails to accomplish this strategy as well. Therefore, in Step 3 the monkey moves to the next possible option, tries to climb up and tries to grab. In this third strategy, the monkey has two options to move: position (1) and position (3). In position (1) the monkey fails because there is no tool to climb. So the monkey takes position (3) where there is a tool. Nevertheless, the monkey fails to complete the strategy or step because it fails to grab the banana. Therefore, the monkey takes the fourth strategy: climb down from the tool, push the tool, climb up the tool and try to grab the banana. In this step the monkey succeeds to accomplish all the goals because the banana is in position (2) and the monkey moves the tool from position (3) to position (2) to climb up and get



the banana. Hence, by searching through the given abilities the monkey reaches to the banana.

It might be comprehensible from the above solution that the idea is not to find out ‘what’ rather than to find out ‘how?’. Such kind of a problem, therefore, suits more with the logic programming domain rather than procedural programming domain. This also reveals a clear distinction between procedural programming and logic programming. Nevertheless, in a brief comparison, logic programming is about finding ways to reach to a particular ‘goal’ (within the logic model or logic world), like getting the banana, while procedural programming is about finding a particular value or set of values (e.g. various numerical models). However, at this stage, it is pertinent to study the basics of logic before proceeding to the advanced sections of this chapter.

### **4.3 Definition and Types of Logic**

Logic may be defined as the science of reasoning. Logic is a non-empirical science like mathematics, which is different from an empirical (i.e., experimental or observational) science like physics, biology or psychology. Logic may be also considered as the science of arguments. Logic sets out the important properties of arguments, especially the ways in which arguments can be good or bad.

Distinguishing the correct reason from incorrect reasoning is the task of logic. Hence, logic investigates and classifies the structure of statements and arguments, both through the study of formal systems of inference and through the study of arguments in natural language. It deals with propositions (declarative sentences, used to make an

assertion, as opposed to questions, commands or sentences expressing wishes) that are capable of being true and false. It is not concerned with the psychological processes connected with thought, or with emotions, images and the like. The summary of the definition and characteristics of logic is given in Table 4.1.

There exist several types of logics namely: formal logic, informal logic, symbolic logic and mathematical logic as shown in the Figure 4.3. Formal logic is generally thought of as traditional logic or philosophical logic, namely the study of inference with purely formal and explicit content. A formal system (also called a logical calculus) is used to derive one expression (conclusion) from one or more other expressions (premises). These premises may be axioms (a self-evident proposition, taken for granted) or theorems (derived using a fixed set of inference rules and axioms, without any additional assumptions). Formalism is the philosophical theory that formal statements (logical or mathematical) have no intrinsic meaning but that its symbols (which are regarded as physical entities) exhibit a form that has useful applications.

Table 4.1: Summary of definition and characteristics of logic.

Define as	<ul style="list-style-type: none"> <li>• Science of reasoning</li> <li>• Science of arguments</li> </ul>
Characteristics	<ul style="list-style-type: none"> <li>• Investigates and classifies statement</li> <li>• Investigates and classifies arguments</li> <li>• Deals with propositions</li> <li>• Not concerned with psychology</li> </ul>

Informal Logic is a discipline which studies natural language arguments, and attempts to develop a logic to assess, analyze and improve ordinary language (or "everyday") reasoning. Natural language here means a language that is spoken, written or signed by humans for general-purpose communication, as distinguished from formal languages (such as computer-programming languages) or constructed languages (such as Esperanto). It focuses on the reasoning and argument one finds in personal exchange, advertising, political debate, legal argument, and the social commentary that characterizes newspapers, television, the Internet and other forms of mass media.

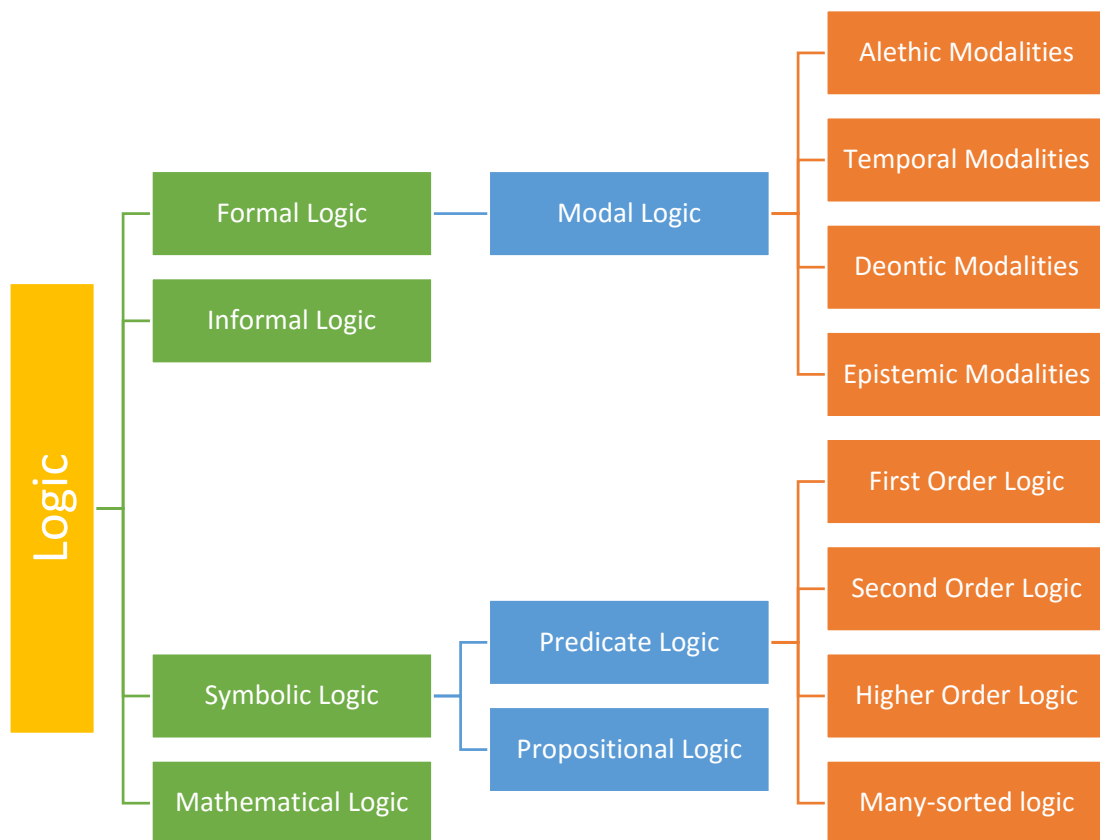


Figure 4.3: Classification of different types of logic.

Symbolic Logic is the study of symbolic abstractions that capture the formal features of logical inference. It deals with the relations of symbols to each other, often using complex mathematical calculus, in an attempt to solve intractable problems traditional formal logic is not able to address. It is often divided into two sub-branches, such as - Predicate Logic: a system in which formulae contain quantifiable variables. And Propositional Logic (or Sentential Logic): a system in which formulae representing propositions can be formed by combining atomic propositions using logical connectives, and a system of formal proof rules allows certain formulae to be established as theorems. The predicate logic is classified into four types such as: (i) First Order Logic, (ii) Second Order Logic, (iii) Higher Order Logic and (iv) Many-sorted logic. This classification is not utilized in this study and therefore, ignored in the rest of the thesis.

Mathematical logic is a subfield of mathematics exploring the applications of formal logic to mathematics. It is a set of mathematical disciplines (such as Boolean algebra, predicate calculus, and propositional calculus) employed in reducing the rules of formal logic to the rules of algebra. Its major objective is to eliminate ambiguities caused by the use of natural languages. Further study can be found in Hardegree (1999) and in Mastin (2008).

In this thesis predicate logics, propositional logics and mathematical logics are utilized for the development of Logic Programming Technique (LPT). Fundamentally, predicate logics and mathematical logics are utilized in the programming codes (Prolog in particular). While propositional logics are used during theoretical construction of the

logic world. However, the similarities and difference between predicate logics and propositional logics are discussed in detail in the following sections.

## **4.4 Propositional Logic vs Predicate Logic**

### *4.4.1 Propositional Logic*

The area of logic which deals with propositions is also called propositional calculus. A proposition is a declarative sentence (that is, a sentence that declares a fact) that is either true or false, but not both. The following three examples may clarify the concept of propositional logic:

- i. Tokyo is the capital of Japan
- ii.  $2 + 2 = 3$
- iii.  $x + y = z$

All of the above three examples can be considered as propositions. However, according to the order of appearance, the propositions are true, false and true or false (depending on assigned value). Propositional logic, additionally known as sentential logic and statement logic, also studies ways of joining and/or modifying entire propositions, statements or sentences to form more complicated propositions, statements or sentences. Using sentential connectives and logical operators (such as "and", "or", "not", "if ... then ...", "because" and "necessarily"), statements or sentences can be formed into more complex propositions, statements or sentences. In propositional logic, the simplest statements are considered as indivisible units.

#### *4.4.2 Predicate Logic*

Predicate logic is the generic term for symbolic formal systems. This formal system is distinguished from other systems in that its formulae contain variables which can be quantified. Predicate Logic allows sentences to be analyzed into subject and argument in several different ways. Predicate Logic is also able to give an account of quantifiers general enough to express all arguments occurring in natural language, thus allowing the solution of the problem of multiple generality. The following example can be considered in this regard:

Three Propositions:

- i. Awal lives in Ibaraki City.
- ii. Jyotsna lives in Minoo City.
- iii. Cessar lives in Minoo City.

Three Predicates:

- iv. live(Awal, Ibaraki).
- v. live(Jyotsna, Minoo).
- vi. live(Cessar, Minoo).

General form predicate:

- vii. live(Person, City).

In the above example, it can be seen that there are three propositions (i to iii). These proposition describes that Awal, Jyotsna and Cessar lives in Ibaraki, Minoo and

Minoo city respectively. These three propositions can be written in the predicate form as well. The name of the predicate is given 'live' and it contains name of the person and the city where the person lives. These three predicates can be written in general form as shown in the example.

Predicate logic is designed as a form of mathematics, and as such is capable of all sorts of mathematical reasoning beyond the powers of term or syllogistic logic. In first-order logic (also known as first-order predicate calculus), a predicate can only refer to a single subject, but predicate logic can also deal with second-order logic, higher-order logic, many-sorted logic or infinitary logic. It is also capable of many commonsense inferences that elude term logic, and has all but supplanted traditional term logic in most philosophical circles.

#### *4.4.3 A Comparison*

The above sections defined and explained both predicate logic and propositional logic. In this section a short tabular for difference between two types of logic is shown in Table 4.2.

### **4.5 Logic Programming**

Logic Programming is the name of a programming paradigm, which was developed in the 1970s. Rather than viewing a computer program as a step-by-step description of an algorithm, the program is conceived as a logical theory, and a procedure call is viewed as a theorem of which the truth needs to be established. Thus, executing a program means searching for a proof (Flach, 1994).

Table 4.2: Difference between propositional logic and predicate logic.

<b>Propositional Logic</b>	<b>Predicate Logic</b>
<p>1. A proposition is a statement of language that formulates something about an external world. A proposition can either be true or false. Thus questions, commands, promises, etc., are not propositions.</p>	<p>1. Predicate logic assumes that the world consists of individual objects which may have certain properties and between which certain relations may hold (the general name for a property or a relation is predicate).</p>
<p>2. Propositional logic simply uses symbols without the ability to do predication. For example: the proposition “If the captain observes danger then the captain alerts all crew” can be expressed as IF X THEN Y. Where, X is “the captain observes danger” and Y is “the captain alerts all crew”</p>	<p>2. Predicate logic is the general term for all logics that use predicates, e.g. <math>p(x)</math>. Here, <math>p</math> is a predicate; we say that <math>p</math> is predicated of <math>x</math>. For example, captain(PERCEPTION, ACTION) means ‘captain’ takes ACTION based on his PERCEPTION. Here the PERCEPTION and ACTION are variables.</p>
<p>3. Propositional logic consists of a set of atomic propositional symbols which aren't variables. These symbols are joined together by logical operators (or connectives) to form sentences.</p>	<p>3. Predicate logic supports the ability to have variables, and quantifiers over variables.</p>



In logic programming, a program consists of a collection of statements expressed as formulas in symbolic logic. There are rules of inference from logic that allow a new formula to be derived from old ones, with the guarantee that if the old formulas are true, so is the new one. Because these rules of inference can be expressed in purely symbolic terms, applying them is the kind of symbol manipulation that can be carried out by a computer. This is what happens when a computer executes a logic program: it uses the rules of inference to derive new formulas from the ones given in the program, until it finds one that expresses the solution to the problem that has been posed. If the formulas in the program are true, then so are the formulas that the machine derives from them, and the answers it gives will be correct. This kind of declarative programming allows the programmer to disregard the precise sequence of actions that takes place when a program is executed, to a much greater extent than is made possible even with high-level imperative programming languages (Spivey, 1996).

Logic program contains no explicit instructions for finding a solution to the problem. In fact, it may seem fanciful to call what we have written a program at all, since it does not seem to describe a computational process; but this absence of explicit instructions is one of the attractions of a declarative style of programming. It turns out that there are powerful, general strategies for finding solutions to problems that have been expressed as logic programs. Each implementation of logic programming includes such a strategy as a central component – for example, many implementations of the logic programming language use a strategy known as ‘SLD–resolution with depth-first search’. Whilst this strategy is not the most powerful one, it is relatively easy to implement efficiently (Spivey, 1996).

These powerful and exceptional characteristics of logic programming can be useful in solving accident problems. Solving accident problems can be regarded as an exercise of uncovering the causes of accidents and determining ways to stop the accident from taking place. Uncovering the cause can be synonymous to 'how' accidents occur. Therefore, if a system (or 'world' in logic programming terms) can be described using causal relations, then the logic programming can deduce new relations which may explain the cause of an accident.

#### *4.5.1 Programming with Relations*

Logic programming works by defining relations between data items. Some techniques can be used to define new relations in terms of existing ones. Drawing on database techniques, it is possible to examine various ways of combining relations to derive the answers to questions (Spivey, 1996). The simplest way to define a relation is to give an explicit list of facts; that is, to define the relation by a table. Such examples will be seen in the later part of the thesis.

#### *4.5.2 Use of Prolog: Clause, Facts and Rules*

In this study Prolog programming environment is utilized. A Prolog program consists of a succession of clauses. A clause can run over more than one line or there may be several on the same line. A clause is terminated by a dot character, followed by at least one 'white space' character, e.g. a space or a carriage return. There are two types of clauses: facts and rules (Bramer, 2005). Facts express unconditional truths, while rules denote conditional truths, i.e. conclusions which can only be drawn when the premises are known to be true. The later sections of this thesis will show more examples.

## 4.6 Fundamentals of Arguments

Arguments are the fundamental building blocks in Logic Programming Technique (LPT). In order to analyze accidents in logic programming domain, logic worlds are created using arguments. An argument is a collection of declarative sentences (propositions) one of which is called conclusion and the rest of which are called premise(s). In this thesis arguments and logic are sometimes used as synonym to each other. The following may be considered as an example of argument:

<i>Premise 1: Ship A heads south-east.</i>	}	<i>An argument</i>
<i>Premise 2: Ship A is sailing. (a variable fact)</i>		
<i>Premise 3: Dangerous underwater rocks are nearby.</i>		
<i>Conclusion: Ship A is on a grounding course.</i>		

### 4.6.1 Deductive Argument

An argument in which, without fail, if the premises are true, the conclusion will also be true. In deductive argument, the truth of the premises necessitates the truth of the conclusion. In deductive argument the task is to distinguish deductively correct arguments from deductively incorrect arguments. Nevertheless, it is necessary to keep in mind that, although an argument may be judged deductively incorrect, it may still be reasonable, that is, it may still be inductively correct. An example of valid deductive argument can be given as follows:

*Premise 1: It is dangerous to sail in stormy weather.*

*Premise 2: It is stormy now.*

*Conclusion: So it is dangerous to sail now.*

#### *4.6.2 Inductive Argument*

An inductive argument is that in which the premises provide good reasons for believing the conclusion. In inductive argument, the premises make the conclusion likely, but the conclusion may be false even if the premises are true. Inductive argument investigates the process of drawing probable (likely, plausible) through fail able conclusion from premises. Another way of stating this: inductive argument investigates arguments in which the truth of the premises makes it likely the truth of the conclusion. An example of inductive argument can be given as follows:

*Premise 1: The Captain of the ship does not follow Bridge Resource Management (BRM) practices.*

*Premise 2: The Senior Officer of the Watch (SOOW) does not follow Bridge Resource Management (BRM) practices.*

*Conclusion: No crew of the ship follow Bridge Resource Management (BRM) practices.*

#### *4.6.3 Consistency, Validity, Soundness and Completeness*

Logical systems or logic worlds are constructed of arguments. Once the system or world is created, it needs to be assessed. For this reason there are four foundation pillars by which logical systems or logic worlds are evaluated. These are:

- i. Consistency
- ii. Validity
- iii. Soundness
- iv. Completeness

Consistency means that none of the theorems or arguments of the system contradict one another. Validity is that one cannot arrive at false statements from true statements by applying theorems which are known to be true (including axioms that are taken as true). Soundness means that the system's rules of proof will never allow a false inference from a true premise. In other words, every statement that is proven is true. Finally, completeness means that there are no true sentences in the system that cannot, at least in principle, be proved in the system. In other words, every true statement can be proven.

Based on the above discussions on the fundamentals of arguments it is possible to construct logic worlds. This design of logic world is the essence of Logic Programming Technique (LPT). It appears that there are, however, many ways of constructing logic worlds. Awal and Hasegawa (2013, 2014a, 2014b, 2015a, 2015b, 2015c, in press) and Hasegawa and Awal (2013) have been investigating this area of research, which have been elaborated in this study. This thesis, thereby, proposes two methods: (i) Propositional Logic Based Technique and (ii) Agent Based Perception-Action Technique. The later method is fundamentally an advancement of the former one. The following sections describe the techniques in detail.

#### **4.7 Technique 1: Propositional Logic Based Technique**

This technique is founded on the description of the world in terms of propositional logic or arguments. One may create a world using any number to arguments. Once the world is created, a search predicate is constructed that searches through the arguments. If, the

arguments describing an accident prove to be true then the search matches the with the true arguments and results in a description of ‘how’ accident is going to take place. We may continue the discussion with an example world where there are two ships: Ship A and Ship B in a water body and where there is an area with underwater rocks. The world is a simple flat space with no numerical description and only ruled by logics. Figure 4.4 shows the world. The ships can change their orientation or heading as their captain orders them. In this world, it is assumed that when ships are sailing they sail at a constant speed. However, the speed value is not considered in this world.

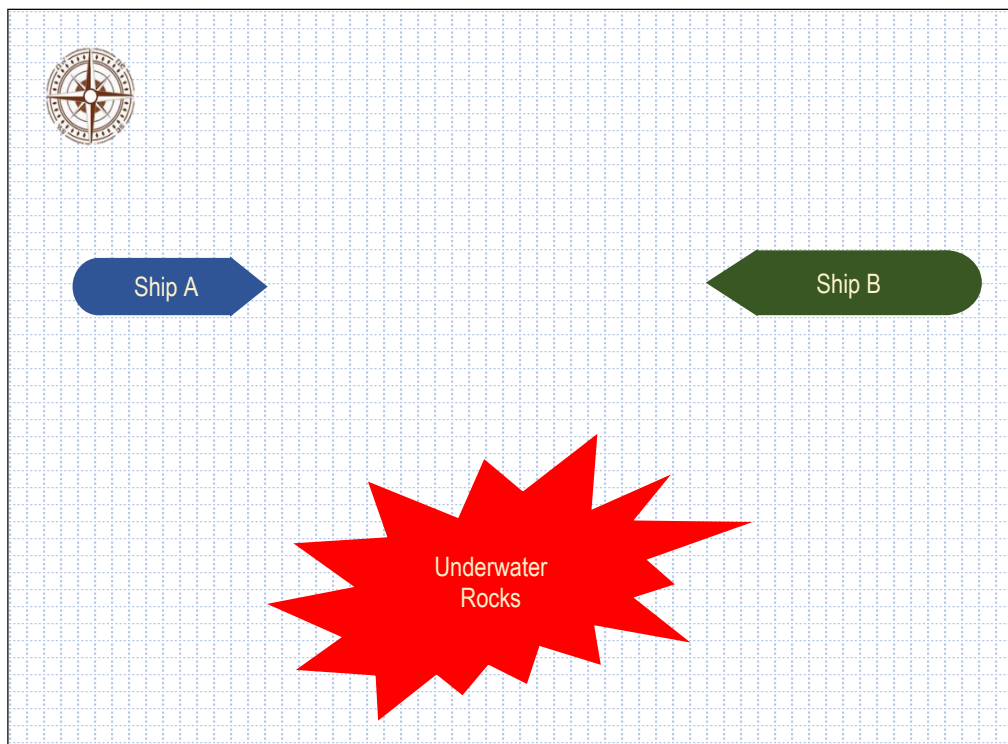


Figure 4.4: Simple world with two ships and underwater rocks.

Now, in order to translate the arguments of the world to logic program (Prolog code) we may utilize systems theory concept. Systems theory is the transdisciplinary

study of the abstract organization of phenomenon, independent of their substance, type, or spatial or temporal scale of existence (Bertalanffy, 1968). It investigates both principles common to all complex entities and the (usual mathematical) models which can be used to describe them. A system can be said to consist of four things:

- 1) Objects: the parts, elements or variables within the system
- 2) Attributes: qualities or properties of the system and its objects
- 3) Relationships: internal relationships among its objects
- 4) Environment: system exists in an environment

Here the objects are Ship A, Ship B and underwater rocks. The environment is the water body. The attribute of ship A and Ship B is such that Ship A can move to north-east or south-east if it sails east while Ship B can move to south-west or north-west if it sails to the west. The relationships of the objects is such that if the objects are placed in the same space then there will be an accident. Thereby, we can simplify the definition of accidents, for example, when Ship A heads to south-east then the ship runs aground. When Ship A heads east and Ship B heads west then there is a collision between the two. When Ship B heads south-west then Ship B gets grounded. However, these rules also govern the ships' heading characteristics.

It is intuitively possible to realize that there are three possible outcomes of accident in this world. For example, the ships may collide with each other if they keep their heading as shown in Figure 4.4 and Figure 4.5 (a). On the other hand, if both or any of the ships change the heading according to the Figure 4.5 (b) then there will be accidents.

Now, using the above-discussed propositions a logic world can be constructed. The next chapter discusses this further and the prolog code can be seen in Appendix C and Appendix D. The general structure of the arguments is shown below:

```
logic(Conclusion, Premise1, Premise2, Premise3):-  
    Premise1    = _____,  
    Premise2    = _____,  
    Premise3    = _____,  
    Conclusion = _____.
```

After constructing the arguments in the above mentioned procedure, it is possible to post a query asking whether there will be any accident or not for a given set of facts. The structure of the query is given as follows:

```
how:-  
    logic(C, P1, P2, P3).
```

Using this methodology, it is possible to deduce some simple logic computations, which are described in Chapter 5 section 5.1 and section 5.2.



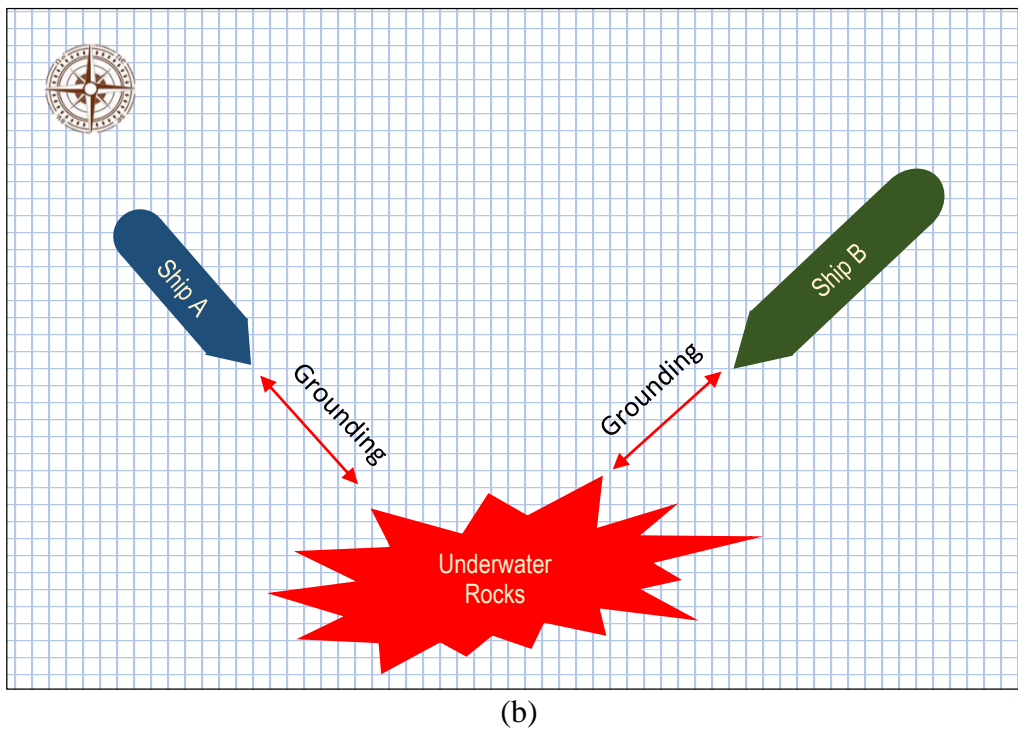
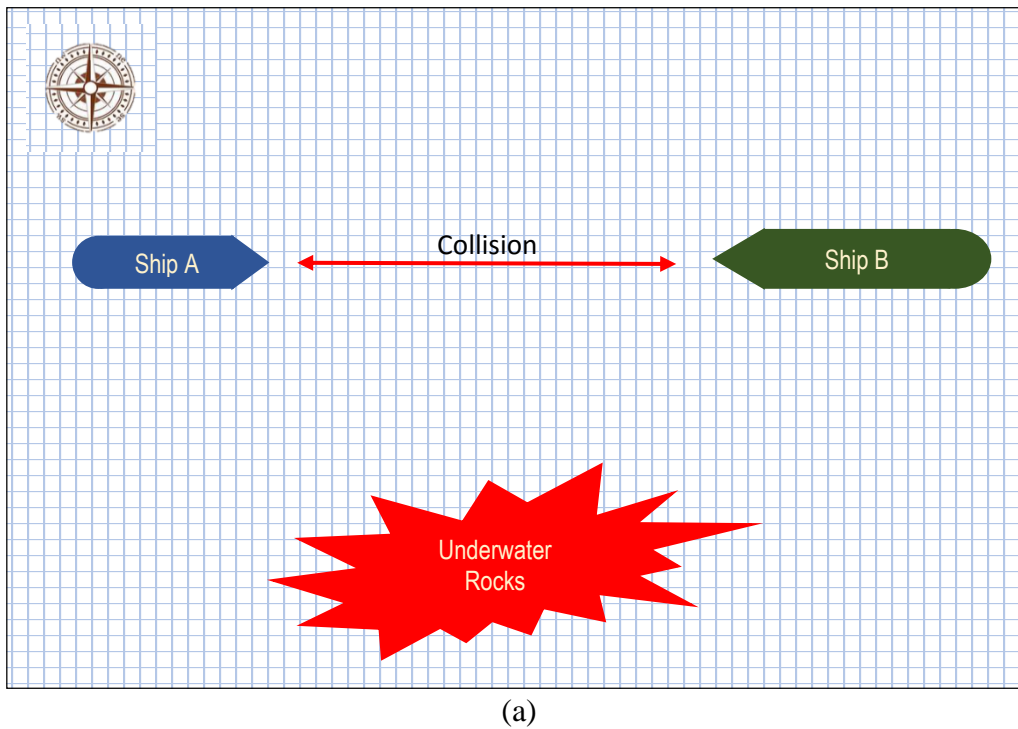


Figure 4.5: Possible accident outcomes – (a) collision between ships and (b) grounding of ship(s) with the underwater rocks.

#### **4.8 Technique 2: Agent Based Perception-Action Technique**

In the previous chapter (Chapter 2) it was discussed maritime accidents can be explained in perception-action sequence of ship crew. Several other notable accidents, such as the Three Mile Island nuclear disaster can be explained in terms of the operators perception-action sequence. Therefore, in this technique it is hypothesized that Logic Programming Technique (LPT) can be used to analyze and deduce the perception-action of human agents using deductive logic along with simulation of the concerned system in order to find out the unknown causes of a particular type of accident.

Fundamentally, an agent can be anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators (Russel et al., 2010). For example, a software agent receives keystrokes, file contents and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files, and sending network packets. In general, for an agent, choice of action at any given instant may depend on the entire percept sequence observed to date but not on anything that it has not perceived. Mathematically, an agent's behavior is described by the agent function that maps and given percept sequence to an action. According to Russel et al., (2010) there are several types of agents with different characteristics:

- i. Simple reflex agent
- ii. Model-based reflex agent
- iii. Goal-based agent
- iv. Utility-based agent
- v. Learning agent

In this method, simple reflex agents are considered for construction of the logic world. The next section describes this further.

#### *4.8.1 Design of Simple Reflex Agent*

The characteristic of a simple reflex agent is that such an agent selects action(s) based on the current percept, ignoring the rest of the percept history. The agent uses the condition-action rule or situation-action rule. The simple reflex agent needs to have a library of rules so that if a certain situation should arise and it is in the set of condition-action rules the agent will know how to react with minimal reasoning.

A schematic diagram of simple reflex agent is shown in Figure 4.6. The agent may perceive through sensors from the environment. The environment can be a software environment or the natural environment. Based on the perception and condition-action rule the agent decides its action and through its actuators it can take an action which may change the environment.

An important aspect of simple reflex agent is that it does not consider the consequences of its action in advance. Such an example of simple reflex agent could be the reaction of a person to fire. A person pulls his or her hand away without thinking about any possibility that there could be danger in the path of his/her arm. This is called reflex action. Similar to a person's reaction to fire, a simple reflex agent behaves relative to the situation and does not consider previous percept.

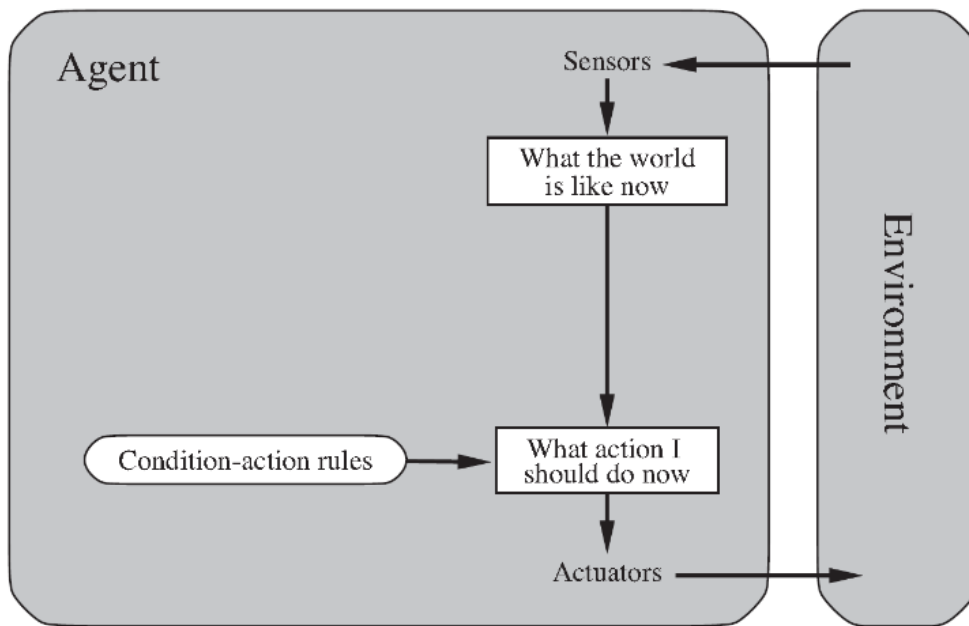


Figure 4.6: A schematic diagram of simple reflex agent (Russel et al., 2010).

In this method, it is important to relate the simple reflex agent to the ship crew. The principal idea is that all crew takes action based on his or her perceptions just like a simple reflex agent. For a particular crew, the perceptions develop from surrounding world parameters and actions from other crew members.

Figure 4.7 shows a schematic diagram of this concept. In this figure four crew members are visualized – a Captain, a Senior Officer of The Watch (SOOW), Junior Officer of the watch (JOOW) and a Helmsman. The Captain may perceive an event from the actions of his co-workers. At the same time the Captain may perceive something from the surrounding environment as well. This perception may lead to an action of the Captain. Once the captain takes an action, the world changes. This change may result in perception(s) for the crew, which may lead to action(s) of the crew(s). Hence the world

changes and such perception-action cycle may continue. The following sections describes each of the agents in detail.

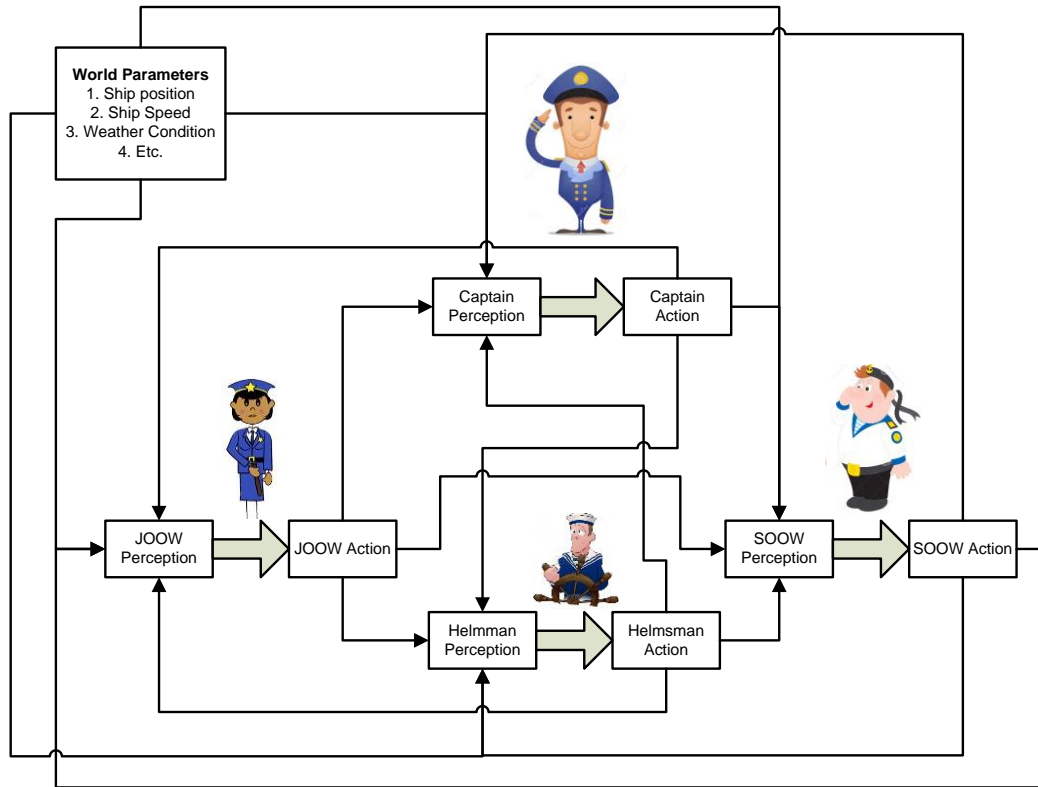


Figure 4.7: Agent based perception-action concept.

#### 4.8.2 Performance, Environment, Actuator and Sensors (PEAS)

An initial yet most significant step for agent design is to specify the task environment as fully as possible. Task environments are essentially the ‘problems’ to which the rational agents are the ‘solutions’ (Russel et al., 2010). It is a general practice to define or describe PEAS (Performance, Environment, Actuators and Sensors) as fully as possible for designing agents. Table 4.3 depicts a description of the agents in terms of PEAS. In this table, five simple reflex agents are considered, as an example, including the ship itself and four ship crewmembers, such as a Captain, a Senior Officer of the Watch (SOOW),

a Junior Officer of the Watch (JOOW) and a Helmsman. The following sections briefly describe the properties of these agents.

Table 4.3: Example of PEAS definition of different agents.

Name of Agent	Performance	Environment	Actuator	Sensor
Ship	Calculate ship position and heading, evaluate status (sailing, grounded, etc.)	Coastal water Underwater rocks	Rudder angle and speed	Rudder command and Speed command
Captain	Visual observation inside and outside the ship, listen to ship crew, Command to ship crew	Bridge deck	Verbal command and manual operation	Vision and hearing
SOOW	Visual observation inside and outside the ship, communicate with ship crew.	Bridge deck	Verbal command and manual operation	Vision and hearing
JOOW	Visual observation inside and outside the ship, communicate with ship crew and monitor route.	Bridge deck	Exchange information and manual operation	Vision and hearing
Helmsman	Visual observation inside and outside the ship, communicate with ship crew and execute command from Captain at the helm.	Bridge deck	Exchange information and manual operation	Vision and hearing

#### 4.8.3 Ship Agent

A ship agent is a mathematical model of ship maneuvering. In this study ship is considered as a simple reflex agent because the ship behaves according to its given commands and does not behave based on its behavior history. For example, the ship receives the rudder command given by helmsman and using this rudder command the ship agent computes its next position in the water, considering the speed, heading and turning rates are initially given. The ship will always compute its next position based on the given inputs and will not consider the new position based on old input values. Thereby

the ship agent behaves like a simple reflex agent. Figure 4.8 shows the definition of ship agent.

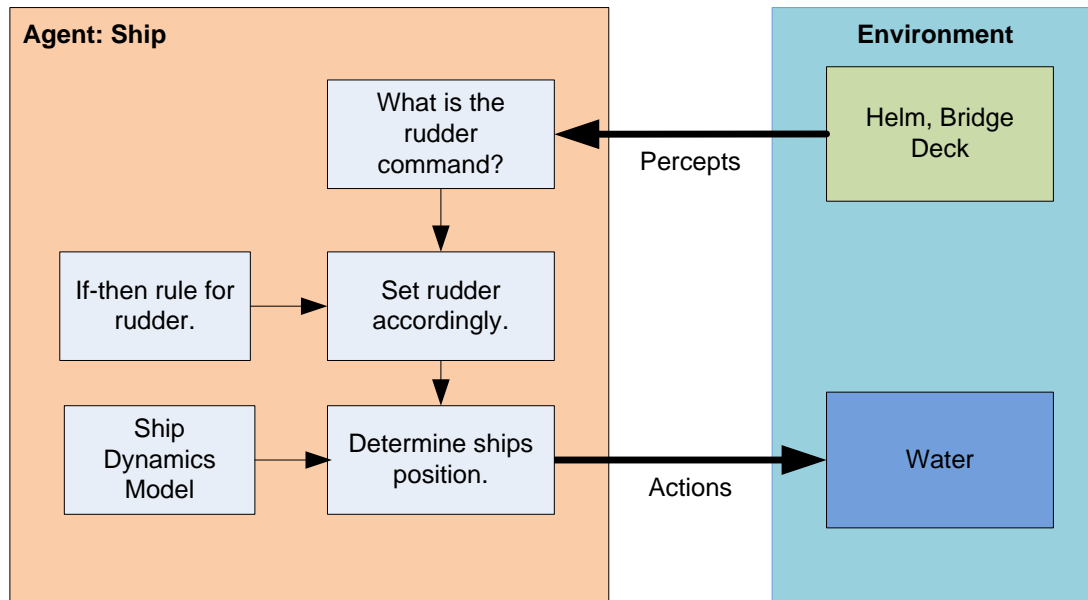


Figure 4.8: Definition of ship agent.

#### 4.8.4 Captain Agent

The Captain of a ship is responsible for every action and its consequences that occur on-board. The Captain must control all the crew and the ship itself. In this study, the captain agent perceives the actions of ship crew and the action of the ship agent itself. Based on this perceptions and simple if-then rules the captain agent takes actions. Actions usually involve giving commands to other crew and manual operations such as controlling the engine rpm. The captain agent necessarily requires to have a set of situation-action rules based on which the agent can perceive and take action. These rules may be derived from the existing regulations and practices. Figure 4.9 defines the captain agent.

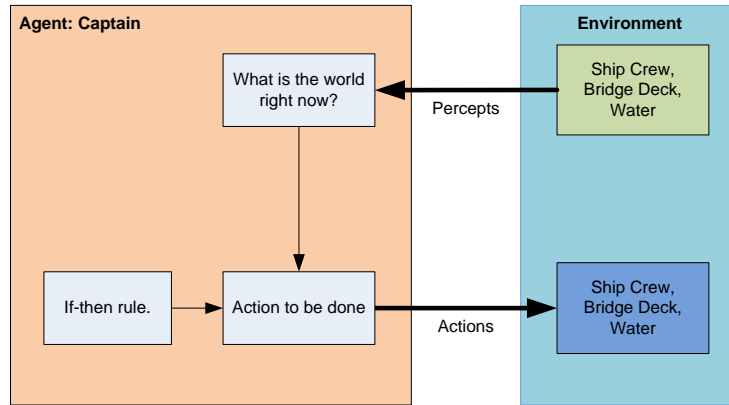


Figure 4.9: Definition of Captain agent.

#### 4.8.5 SOOW Agent

In a ship, the senior officer of the watch (SOOW) needs to follow the tasks assigned by the Captain. For example, in the case of MV Costa Concordia, the SOOW was assigned to conduct ship maneuvering and route monitoring at different times during its voyage. In this example, the SOOW agent works under the captain and his working environment is inside the bridge deck. The agent perceives from the actions of other ship crew and visual observation from bridge deck gadgets. He may order the JOOW or Helmsman and conduct manual operations (e.g. route planning). Figure 4.10 defines SOOW agent.

#### 4.8.6 JOOW Agent

In a ship, the Junior Officer of the Watch (JOOW) usually works under the Captain and the SOOW and executes the orders of his or her superiors. For example, the JOOW may conduct route monitoring on the paper chart during a voyage or may execute any other command given by the Captain. In this study, the JOOW agent can perceive from the orders and actions from the ship crew. His own actions will be executing the orders from



his superiors and ordering to his juniors. He may perceive from the surrounding world as well. Figure 4.11 defines the JOOW agent.

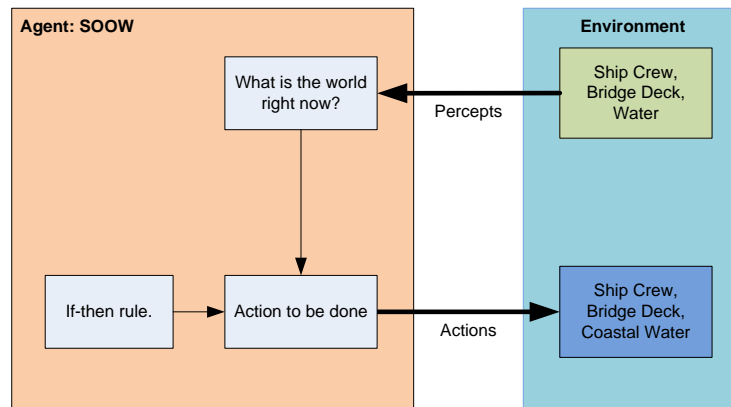


Figure 4.10: Definition of SOOW agent.

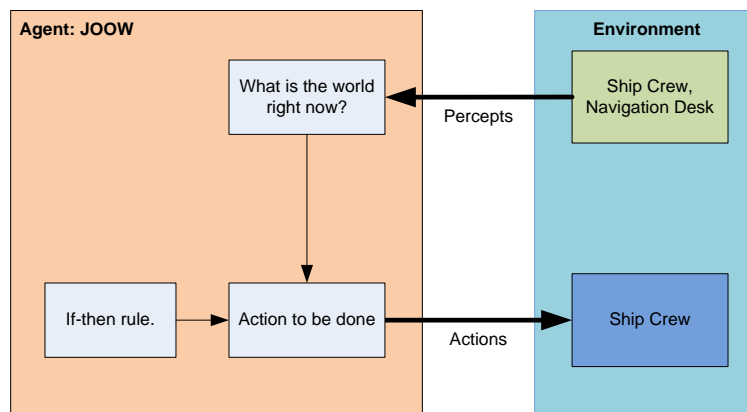


Figure 4.11: Definition of JOOW agent.

#### 4.8.7 Helmsman Agent

The helmsman of a ship is the crewmember who executed the rudder command given by the ship and usually is stationed at the helm of the ship. The helmsman is often responsible for executing the engine rpm command depending on the circumstances and environment.

In this example, the Helmsman agent only executes the rudder command given by the Captain. Figure 4.12 defines Helmsman agent.

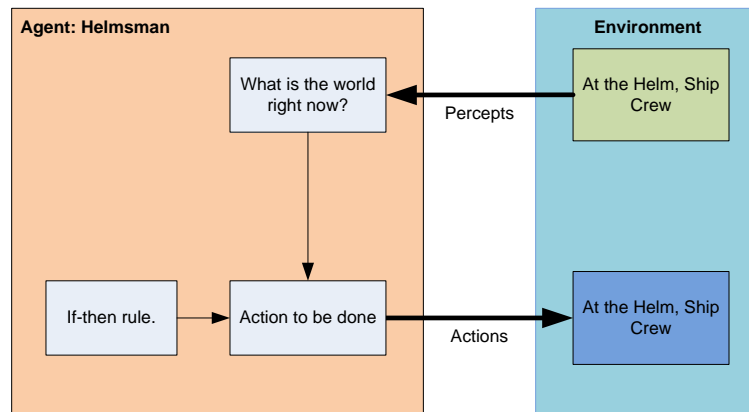


Figure 4.12: Definition of Helmsman agent.

Table 4.4 shows some of the perception-action arguments as example. The first item in Table 4.4 is the captain perception. Here if the captain perceives that the engine rpm is low then the captain concludes that he or she needs to transfer the engine control from bridge deck to the chief engineer who is in the engine room. Similarly, item no 2 shows that chief engineer perception. If the ship has a history of engine starting problems, then the chief engineer may ask the captain to transfer the engine control in the case of an engine trouble during a voyage.

Using the above mentioned concepts of different agents, perception-action arguments can be constructed which are discussed further in the next chapter. An example of coding in prolog is also demonstrated in Appendix E.

Table 4.4: Example of perception-action arguments.

No.	Crew	Arguments
1.	Captain Perception	<i>Premise:</i> Engine RPM Low. <i>Conclusion:</i> Transfer Control To Chief Engineer
2.	Chief Engineer Action	Premise: Engine has history of starting problem. Conclusion: During a voyage if the engine fails then request transfer of control from bridge deck to engine room.

#### 4.9 Chapter Summary

This chapter discussed the fundamentals of logic programming and logic programming technique. The definition and classification of logic is shown. Two different types of methods were developed and demonstrated: (i) Propositional Logic Based Technique and (ii) Agent Based Perception-Action Technique. Propositional logic Based Technique demonstrates utilization of simple arguments and query through these arguments. On the other hand the Agent Based Perception-Action Technique demonstrates utilization of simple reflex agents.

## **Chapter 5: Development of Logic Worlds and Analysis**

### **5.1 Introduction**

This chapter discusses some results obtained from posting queries into the logic worlds. Three different examples were given based on the methods described in Chapter 4. Example 1 and Example 2 were created using the Propositional Logic Based Technique and the third example was created using the Agent Based Perception-Action Method. The following sections discuss these further.

### **5.2 Example 1 – Propositional Logic Based Technique**

In this example an obvious scenario is constructed where one can easily find out (with some simple judgement) the possible outcomes. The idea is to demonstrate that by posting queries to the logic world it is possible to find out the probable accidents and the way it may take place. This logic world is constructed using ten propositional arguments. The logics are presented in terms of simple arguments (premise-conclusion) format as shown in Table 5.1.

In this world, there are several limitations. For, example the captain of the Ship A may command to head north-east only if the ship heads to north. Similarly, the captain may command head south-east only if the ship heads to south. In this world the captain of Ship A has no other commands. In case of heading to east or west no other commands were given as arguments.

Table 5.1: Ten propositional logics for Example 1.

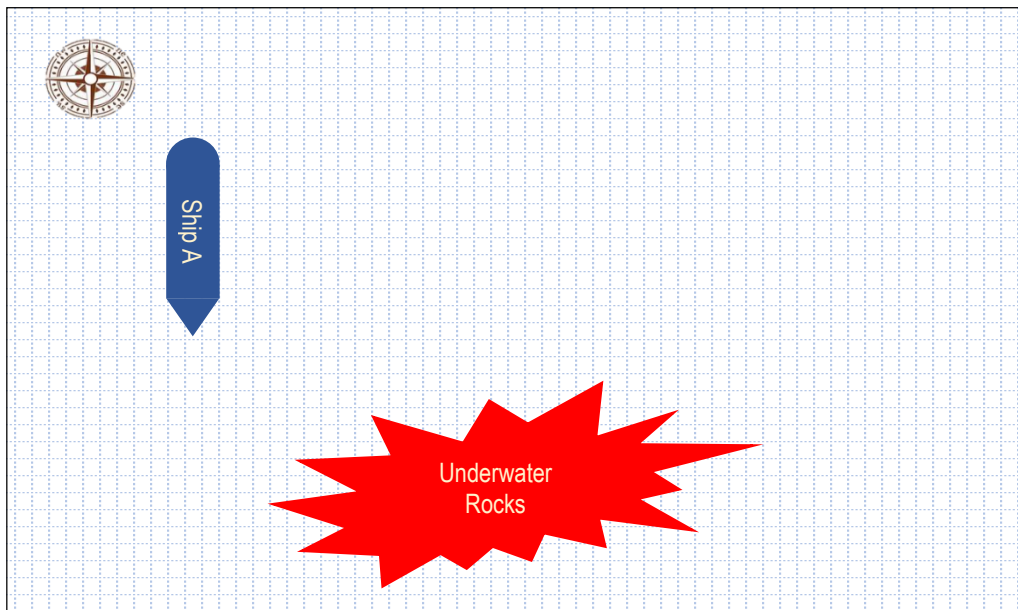
Logic Number	Related to	Logic
1	Ship A	Premise 1: Ship A heads north. Premise 2: Captain of Ship A commands head north-east. Conclusion: Ship A heads north-east.
2	Ship A	Premise 1: Ship A heads south. Premise 2: Captain of Ship A commands head south-east. Conclusion: Ship A heads south-east.
3	Ship B	Premise 1: Ship B heads north. Premise 2: Captain of Ship B commands head north-west. Conclusion: Ship B heads north-west.
4	Ship B	Premise 1: Ship B heads south. Premise 2: Captain of Ship B commands head south-west. Conclusion: Ship B heads south-west.
5	Ship A and underwater rocks	Premise 1: Ship A heads south-east. Premise 2: Ship A is sailing. (a variable fact) Premise 3: Dangerous underwater rocks are nearby. Conclusion: Ship A is on a grounding course.
6	Ship B and underwater rocks	Premise 1: Ship B heads south-west. Premise 2: Ship B is sailing. (a variable fact) Premise 3: Dangerous underwater rocks are nearby. Conclusion: Ship B is on a grounding course.
7	Ship A and Ship B	Premise 1: Ship A heads east. Premise 2: Ship B heads west. Premise 3: Ship A is sailing. (a variable fact) Premise 4: Ship B is sailing. (a variable fact) Conclusion: Ship A and Ship B are in a collision course.
8	Ship A and underwater rocks	Premise 1: Ship A is on a grounding course. Premise 2: Ship A is sailing. (a variable fact) Premise 3: Dangerous underwater rocks are nearby. Conclusion: There will be an accident.
9	Ship B and underwater rocks	Premise 1: Ship B is on a grounding course. Premise 2: Ship B is sailing. (a variable fact) Premise 3: Dangerous underwater rocks are nearby. Conclusion: There will be an accident.
10	Ship A and Ship B	Premise 1: Ship A and Ship B are in a collision course. Premise 2: Ship A is sailing. (a variable fact) Premise 3: Ship B is sailing. (a variable fact) Premise 4: Ship A heads east. (given condition) Premise 5: Ship B heads west. (given condition) Conclusion: There will be an accident.

Similarly, for Ship B, the captain is given limited commands. It is however, pertinent to mention that why the captain decides the heading is beyond to scope of this

example. This issue is discussed in Example 3 where crew perception-actions are depicted. Another feature of this world related to the ships is that it is possible to remove any of the ship by inactivating any of the facts related to sailing of ships. Such cases can be seen in the examples given later.

Using these ten arguments it is possible to consider and analyze several cases. For example, in case one (Figure 5.1) only one ship (Ship A) and the underwater rocks are considered. Ship A in this case heads south. The query ‘how’ results in the logical deductions shown in the figure. In this case there will be an accident. The logic model reveals that the captain of Ship A may command to head south-east the ship will encounter a grounding course with the underwater rocks, thereby, there will be an accident.

In case two (as shown in Figure 5.2) there is the single ship ‘Ship A’ and the underwater rocks. However, in this case the query ‘how’ is unable to deduce any accident within the given arguments. The reason is because the captain of Ship A has no arguments for sailing to east. Therefore, there is no change in heading of the ship and thereby no accident.



Input
Output
<pre> Ship A heads south-east. Because:   1. Captain of Ship A commands head south-east. true ;  Ship A is on a grounding course. Because:   1. Ship A heads south-east.   2. Dangerous underwater rocks are nearby. true ;  There will be an accident. This is because:   1. Ship A is on a grounding course. true ; false. </pre>

Figure 5.1: Example 1 – case one.

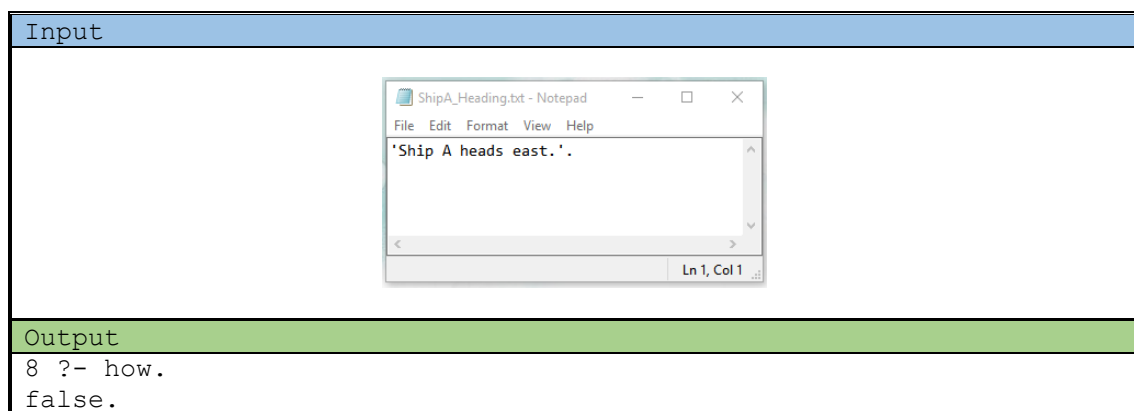
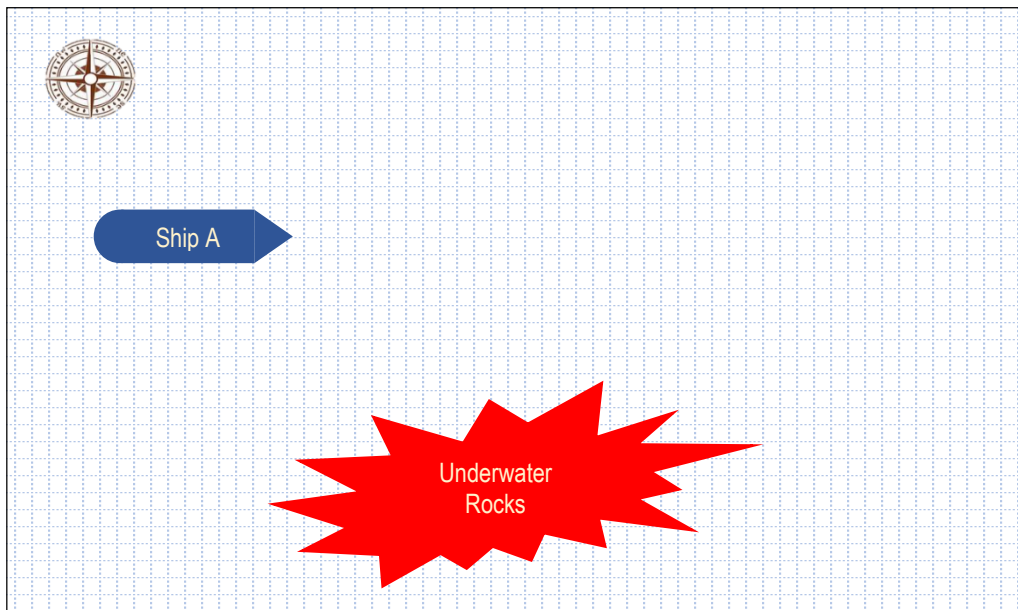


Figure 5.2: Example 1 – case two.

In case three, both ships, Ship A and Ship B, are now considered. Ship A is heading north and ship B is heading south as shown in Figure 5.3. Under this circumstance, the query ‘how’ results in an accident for Ship B but no accident for Ship A. since, in this world there is no argument for accident for ships heading north, therefore, there will be no accident. However, the captain of Ship A may change the heading from north to north-east, but there is no argument for accidents. Anyhow, Ship B will encounter a grounding accident as the captain of Ship B commands to change heading from south to south-east.





Input

ShipA\_Heading.txt - Notepad

File Edit Format View Help

'Ship A heads north.'.

Ln 1, Col 1

ShipB\_Heading.txt - Notepad

File Edit Format View Help

'Ship B heads south.'.

Ln 1, Col 1

Output

```

3 ?- how.

Ship A heads north-east. Because:
  1. Captain of Ship A commands head north-east.
true ;

Ship B heads south-west. Because:
  1. Captain of Ship B commands head south-west.
true ;

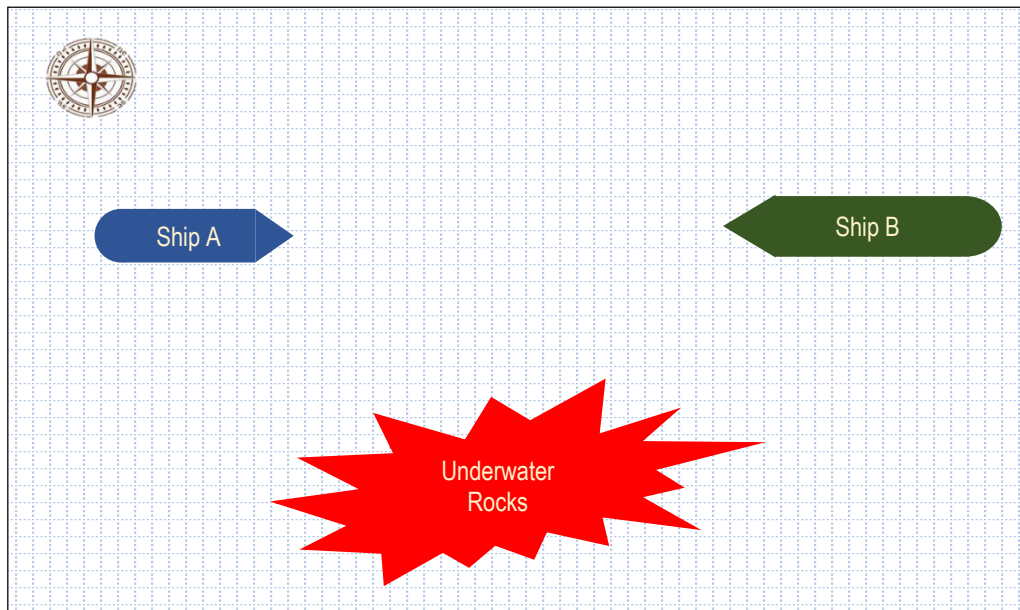
Ship B is on a grounding course. Because:
  1. Ship B heads south-west.
  2. Dangerous underwater rocks are nearby.
true ;

There will be an accident. This is because:
  1. Ship B is on a grounding course.
true ;
false.

```

Figure 5.3: Example 1 – case three.

In case four, Ship A and Ship B are heading to each other. Since neither Ship A nor Ship B is given arguments for changing heading from east or west, there will be an accident of head on collision. The logical deductions are shown in Figure 5.4.



Input
<div><div>ShipA_Heading.txt - Notepad File Edit Format View Help 'Ship A heads east.' Ln 1, Col 1</div><div>ShipB_Heading.txt - Notepad File Edit Format View Help 'Ship B heads west.' Ln 1, Col 1</div></div>
Output
<pre>4 ?- how.  Ship A and Ship B are in a collision course. Because:   1. Ship A heads east.   2. Ship B heads west. true ;  There will be an accident. This is because:   1. Ship A and Ship B are in a collision course. true.</pre>

Figure 5.4: Example 1 – case four.

In this world, simple and obvious arguments are utilized from which probable accidents are easily deducible. The example cases demonstrated that such logic models are capable of deriving how an accident may unfold. However, a practical world model would require lot more parameters for consideration and very precise computations of engineering systems in order to produce applicable results. From this viewpoint, further developments are necessary. Hence, the next sections discusses Example 2 where arguments that are more complex and accident is less obvious.

### **5.3 Example 2 – Propositional Logic based Technique**

This world is constructed using fourteen arguments. The arguments are shown in Table 5.5. These arguments are constructed primarily for describing the ship's propulsion power and controllability. The events occurred during the accident of MV Bright Field and Planet V have been studied and utilized for developing these arguments. In this world the events are complicated and less easier to visualize. Therefore, unlike the Example 1, no pictures are drawn. Nevertheless, the aim is to test complicated arguments using the same format as described in the previous example. This world has be following assumptions:

- i. The crew of the ship is ideal i.e. they exercise all the regulations as it is and do not disobey any rule or conduct any crime.
- ii. 'Ground is nearby.' means the crew is able to see ground by bare eye.
- iii. 'Ship has speed.' means that the ship is in forward motion.
- iv. 'Ship is uncontrollable.' means there is no possible way of keeping its desired ship's speed and heading.
- v. In case of emergency bow thruster is able to change course and avoid collision with another ship.

Table 5.2: Fourteen propositional logics for Example 2.

Logic No.	Related to	Premises and Conclusion
1	Ship and environment	Premise 1: Ground is nearby. Premise 2: Ship has speed. Premise 3: Ship is uncontrollable. <i>Conclusion: Ship will hit ground.</i>
2	Ship vs Ship	Premise 1: Ship has speed. Premise 2: Another ship is in collision course. Premise 3: Ship is uncontrollable. <i>Conclusion: Ship will collide with another ship.</i>
3	Ship, engine and bow thruster	Premise 1: Ship has speed. Premise 2: Engine shutdown. Premise 3: Bow thruster shutdown. <i>Conclusion: Ship is uncontrollable.</i>
4	Ship, engine and rudder	Premise 1: Engine not delivering enough power. Premise 2: Rudder is not functional. <i>Conclusion: Ship is uncontrollable.</i>
5	Ship and engine	Premise: Engine not delivering enough power. <i>Conclusion: Ship is uncontrollable.</i>
6	Engine	Premise: Engine automatic shutdown. <i>Conclusion: Engine not delivering enough power.</i>
7	Engine	Premise: Engine manual shut down. <i>Conclusion: Engine not delivering enough power.</i>
8	Engine	Premise: Lubricating oil pressure low. <i>Conclusion: Engine automatic shutdown.</i>
9	Engine	Premise: Lubricating oil pump fails. <i>Conclusion: Lubricating oil pressure low.</i>
10	Engine	Premise: Engine shutdown. <i>Conclusion: Faulty regulator.</i>
11	Engine	Premise: Engine shutdown. <i>Conclusion: Shaft generators shutdown.</i>
12	Rudder	Premise: Rudder is not functional. <i>Conclusion: Ship is uncontrollable.</i>
13	Generator	Premise: Commanded to shutdown auxiliary generators. <i>Conclusion: Auxiliary generators shutdown.</i>
14	Bow thruster	Premise 1: Shaft generators shutdown. Premise 2: Auxiliary generators shutdown. <i>Conclusion: Bow thruster shutdown.</i>

Three different sets of cases are presented in this world for simple demonstration. The first case is where a ship is in forward motion which is given as a fact ‘Ship has speed.’. The ship is sailing through inland waters where the crew can easily see the ground. The ship is considered to have functional rudder and will remain functional

during the logic computation. Under the circumstance, a query on how an accident may occur will result in a set of logical outputs which as shown in Figure 5.5.

Input
<pre>fact('Ship has speed.'). fact('Ground is nearby.'). fact('Rudder is functional.').</pre>
Output
<pre>1 ?- how. Ship will hit ground. This is because of the following premises:   1. Ground is nearby.   2. Ship has speed.   3. Ship is uncontrollable. true ;  Ship is uncontrollable. This is because of the following premise:   1. Engine not delivering enough power. true ;  Engine not delivering enough power. This is because of the following premise:   1. Engine automatic shutdown. true ;  Engine automatic shutdown. This is because of the following premise:   1. Lubricating oil pressure low. true ;  Lubricating oil pressure low. This is because of the following premise:   1. Lubricating oil pump fails. true ; false.</pre>

Figure 5.5: Example 2 – case one.

The output of the logic model is executed through ‘how’ predicate which is discussed in the earlier section. This predicate attempts to find a match within the constructed logic world with the given facts. At first it obtains a match and delivers the first logical conclusion that the ‘ship will hit ground’. The predicate generates the reasoning based on three premises 1. Ground is nearby, 2. Ship has speed and 3. Ship is

uncontrollable. Then the ‘how’ predicate backtracks and attempts to find another logic which may match with the facts. Hence it concludes that ‘Ship is uncontrollable’ because ‘Engine not delivering enough power’.

In this way the ‘how’ predicate continues until all the logic predicates are exhausted. This analysis suggest that the ship crew may comprehend the possible danger and if possible may take necessary action which are allowable within the regulations to avoid an accident. For example, this case is similar to the accident of Bright Field, a manual restart of the engine from the engine room could have restored power little earlier and that could prevent the accident.

In the second case the input facts are changed as shown in Figure 5.6. It is considered that there are two ships in collision course. One of the ship has a faulty engine regulator and that ship has shut down its auxiliary power units after leaving port. The ship has a bow thruster which are usually powered using the auxiliary power units and can also be powered using engine shaft generator.

Now by posting a query ‘how’ the accident may occur will result in a set of outputs in form of arguments. At first the ‘how’ predicate obtains a match and delivers the first logical conclusion that the ‘ship will collide with another ship’ because 1. Ship has speed, 2. Another ship is in collision course and 3. Ship is uncontrollable. Then the how predicate backtracks and attempts to find another logic which may match with the facts. Hence it concludes that ‘Ship is uncontrollable’ because 1. Ship has speed, 2. Engine shutdown and 3. Bow thruster shutdown. Similarly the logical arguments are deduced which are

different from case 1. The analysis suggest that ship became uncontrollable because of the failure of engine regulator. Since the auxiliary power units were shut down, the bow thruster was not operational.

Input
<pre>fact('Ship has speed.'). fact('Another ship is in collision course.'). fact('Commanded to shutdown auxiliary generators.'). fact('Faulty regulator.').</pre>
Output
<pre>2 ?- how.  Ship will collide with another ship. This is because of the following premises:     1. Ship has speed.     2. Another ship is in collision course.     3. Ship is uncontrollable. true ;  Ship is uncontrollable. This is because of the following premises:     1. Ship has speed.     2. Engine shutdown.     3. Bow thruster shutdown. true ;  Engine shutdown. This is because of the following premise:     1. Faulty regulator. true ;  Bow thruster shutdown. This is because of the following premise:     1. Shaft generators shutdown.     2. Auxiliary generators shutdown. true ;  Auxiliary generators shutdown. This is because of the following premise:     1. Commanded to shutdown auxiliary generators. true ;  Shaft generators shutdown. This is because of the following premise:     1. Engine shutdown. true.</pre>

Figure 5.6: Example 2 – case two.

In this hypothetical model world it is assumed that the bow thruster action is sufficient to maneuver the ship out of collision course. Therefore, if the auxiliary power units were kept running, it can be logically deduced that the ship will not be uncontrollable anymore and hence the ship may avoid a collision. Figure 5.7 shows case three. In this analysis the fact('Commanded to shutdown auxiliary generators.') is no longer true in the input section. Therefore, logically it can be deduced that the bow thruster is operable and emergency maneuver is no longer necessary. As the crew are ideal crew, they will apply the bow thruster to change course and avoid a collision. Therefore, the 'how' predicate could not match any of the logic that can prove the truth of an accident. Hence, the output deduces nothing i.e. no accidents in this world.

Input
fact('Ship has speed.'). fact('Another ship is in collision course.').  %fact('Commanded to shutdown auxiliary generators.').
Output
3 ?- how. false.

Figure 5.7: Example 2 – case three.

From the above discussions it is evident that logical deductions can be utilized in identifying how accidents may occur. However, apparently it seems that constructing logic worlds will be challenging due to several reasons. Firstly, construction of realistic and precise logic worlds may become tedious and time consuming. It will require a significant number of arguments for coding. Secondly, models from multiple disciplines



will be needed to be put together. For example to model the behavior of the ship crew it will be necessary to build a social behavior model. In order to model the ship's performance, the ship maneuvering models will have to be translated into logic programming domain. Although such study might seem challenging yet this gives an advantage of writing codes of different disciplines and utilize it in the same platform (e.g. Prolog), which perhaps has never been done before. Hence, the above results shows that it is possible to predict and analyze accidents with possible causes in the logic programming domain. The next world, Example 2 advances from Example 2.

#### **5.4 Example 3 – Agent Based Perception-Action Technique**

In this world, the agent based perception-action technique is utilized. One unique aspect of this world which stands out from the rest of the two worlds is that in this world numerical computations of ship maneuvering is incorporated along with the behavior of ship crew. Therefore, at first, the assumptions are discussed briefly. The knowledge of the human agents are discussed in tabular form where the arguments are presented. Each argument is presented using with one premise and one conclusion. In this particular study the agents are given very limited knowledge of perceptions and actions so that the reader can easily comprehend and develop further in the future.

##### *5.4.1 Assumptions*

In this study a simplified scenario is considered such as the following:

- i. Action-perception cycle of three crew members are studied in this simulation:  
(1) Captain, (2) Senior Officer of the Watch (SOOW) and (3) Junior Officer of the Watch (JOOW)

- ii. The ship's original starting position in space is considered as (0, 0) where the vertical axis represents advance distance of ship and horizontal axis represents transfer distance of ship.
- iii. There is a zone of scattered rocks visible from 2000 m in clear daylight but not visible at night. If the ship enters that zone, grounding accident is assumed to take place.
- iv. The scattered rocks are located at a coordinate of (0, 3000), that is vertically 3 kilometer away from the starting position.
- v. The captain agent of may see the scattered rocks at night from a distance of 500 meter or less.

#### *5.4.2 Ship Agent*

A ship agent is a mathematical model of ship maneuvering. In this study ship is considered as a simple reflex agent because the ship behaves according to its given commands and does not behave based on its behavior history. For example, the ship receives the rudder command given by helmsman and using this rudder command the ship agent computes its next position in the water, considering the speed, heading and turning rates are initially given. The ship will always compute its next position based on the given inputs and will not consider the new position based on old input values. Thereby the ship agent behaves like a simple reflex agent.

The mathematical model for ship response to rudder commands is determined by Nomoto's linear K-T model (Tzeng and Chen, 1999; Journée and Pinkster, 2002; Nomoto et al., 1957). The cardinal equations are given as follows:

$$T\ddot{\psi} + \dot{\psi} = K\delta_r \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad \dots \quad (5.1)$$

Where,

$\psi$  = Course angle

$\delta_r$  = Rudder angle

$T = \hat{T} \frac{U_0}{L}$

$K = \hat{K} \frac{U_0}{L}$

$U_0$  = Initial forward speed

$L$  = Ship length

$\hat{T}$  &  $\hat{K}$  are non dimensional maneuvering coefficients

For the ship maneuvering motion, the transition phase between dead stop to full ahead speed is not considered. The initial conditions are given in Table 5.3.

Table 5.3: Assumptions for ship maneuvering model.

No.	Item		Value	Unit
1.	Initial position in X axis		0	Meter
2.	Initial position in Y axis		0	Meter
3.	Initial heading		0	Degree
4.	Initial yaw rate		0	Degree/second
5.	Initial rudder angle		0	Degree
6.	Steady state speed		3	Meter/second
7.	Maneuvering indices	K	0.005	
		T	300	Second

#### 5.4.3 Captain's Knowledge

The captain agent's knowledge of perceptions are presented in Table 5.4. The knowledge is shown in terms of arguments where there are two parts: a premise and a conclusion.

The actions of captain are shown in Table 5.5. Here the captain agent plays the role of overall command.

Table 5.4: Captain's perceptions.

Logic No.	Statements	
1	Premise	Conduct route planning on small-scale chart.
	Conclusion	Ship is ready for voyage.
2	Premise	Conduct route planning on large-scale chart.
	Conclusion	Ship is ready for voyage.
3	Premise	Declare danger ahead.
	Conclusion	Need to change heading.
4	Premise	Lift anchor.
	Conclusion	Anchor lifted.
5	Premise	Declare danger ahead.
	Conclusion	Danger ahead.

Table 5.5: Captain's actions.

Logic No.	Statements	
1	Premise	Need to make a sail past
	Conclusion	Command SOOW to change voyage plan for sail past
2	Premise	Ship is ready for voyage
	Conclusion	Command JOOW to lift anchor
3	Premise	Anchor lifted
	Conclusion	Command JOOW - Full Ahead
4	Premise	Danger ahead
	Conclusion	Command JOOW 10 degree starboard

#### 5.4.4 SOOW's Knowledge

The SOOW agent's knowledge of perceptions and actions are presented in Table 5.6 and Table 5.7 respectively. The SOOW plays the role of route planning and monitoring on navigation charts.

Table 5.6: SOOW's perceptions.

Logic No.	Statements	
1	Premise	Command SOOW to change voyage plan for sail past
	Conclusion	Need to change voyage plan for sail past
2	Premise	Need to change voyage plan for sail past
	Conclusion	Need to conduct route planning

Table 5.7: SOOW's actions.

Logic No.	Statements	
1	Premise	Need to conduct route planning
	Conclusion	Conduct route planning on small scale chart
2	Premise	Need to conduct route planning
	Conclusion	Conduct route planning on large scale chart
3	Premise	Danger ahead
	Conclusion	Declare danger ahead

#### 5.4.5 JOOW's Knowledge

The JOOW agent is responsible for executing the commands from his/her superior such as lifting the anchor, speed of the ship and executing rudder command. The JOOW agent's knowledge of perceptions are presented in Table 5.8 and the knowledge of actions are shown in Table 5.9.

Table 5.8. JOOW's perceptions.

Logic No.	Statements	
1	Premise	Command JOOW to lift anchor
	Conclusion	Need to lift anchor
2	Premise	Command JOOW - Full Ahead
	Conclusion	Need to execute command - Full Ahead
3	Premise	Command JOOW 10 degree starboard
	Conclusion	Need to execute 10 degree starboard

Table 5.9: JOOW's actions.

Logic No.	Statements	
1	Premise	Need to execute 10 degree starboard
	Conclusion	Execute 10 degree starboard
2	Premise	Need to lift anchor
	Conclusion	Lift anchor
3	Premise	Need to execute command - Full Ahead
	Conclusion	Execute command - Full Ahead

#### 5.4.6 Model Run and Discussion

Based on the above mentioned assumptions and scenario settings the model is constructed and executed in Prolog environment. The objective is to find out which decision made by the crew may result in a possible accident. A scenario is considered where a voyage begins at night. The voyage had an original route planned but the route is required to be changed due to some reason. The reason is beyond the scope of this study. Figure 5.8 shows the ship path for of two cases where in one case the SOOW decided to use small-scale chart and in the other case the large-scale chart. The characteristics of these two charts are such that the small scale chart shows some scattered rocks and the large scale chart doesn't show the scattered rocks.

The logical deductions derived from the perception-action of agents are shown iteratively in Table 5.10 and Table 5.11. It is evident from Figure 5.8 that the ship following small scale chart easily avoids the scattered rocky zone. The logical deduction shown in Table 5.10 reveals the reason. In small scale charts the rocky region is clearly marked and SOOW who is following the route notices and declares the danger ahead (iteration no. 72). The captain perceives and responds to SOOW and orders JOOW for 10

degree starboard rudder command (iteration no. 73). The JOOW responds immediately and executes the rudder order. Hence the grounding is avoided.

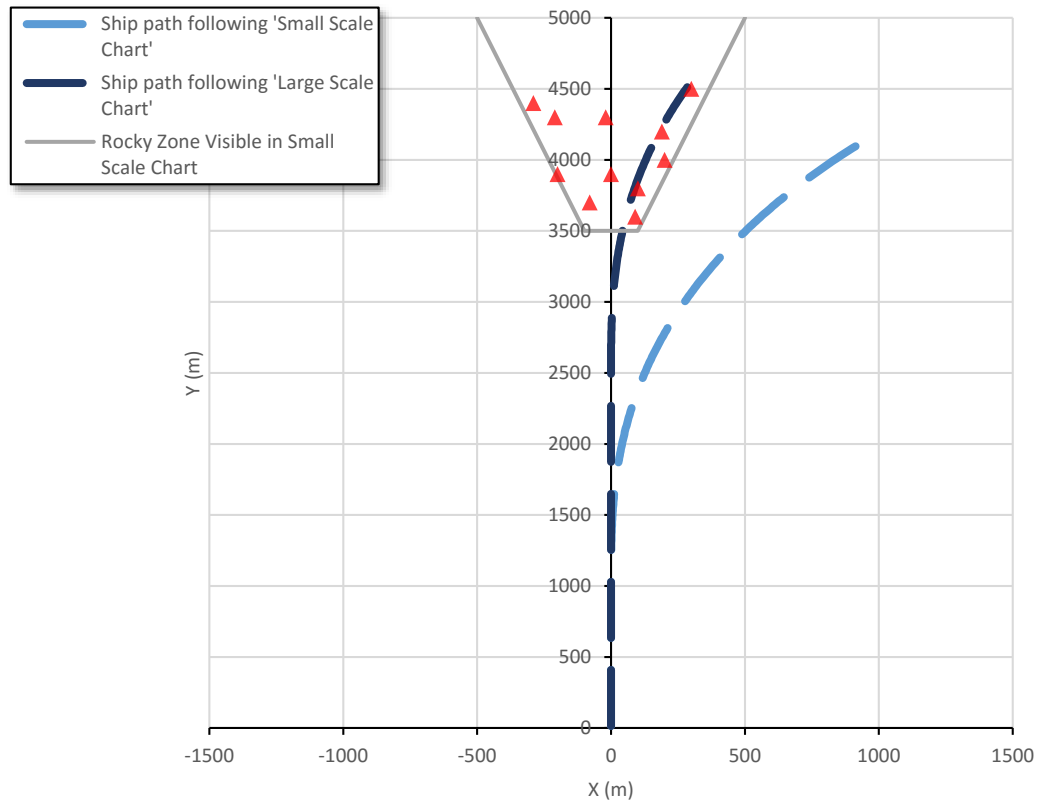


Figure 5.8: Ship's path in two cases: Following small-scale chart and following large scale chart.

On the other hand, when the SOOW decides to utilize large scale chart, the scenario is quite different. As it is shown in Table 10 that the danger is not observed by the SOOW on his chart. However, the Captain who was on the watch himself could look and anticipate the danger and order the JOOW for 10 degree starboard rudder order (iteration no. 172). Yet the decision was not sufficient enough to avoid the scattered rocky zone as shown in Figure 5.8.

Table 5.10: Results of logical deductions of crew perception-actions (small-scale chart chosen for navigation)

Ship following 'Small scale chart'										
Iteration No.	Captain Perception	Captain Action	SOOW Perception	SOOW Action	JOOW Perception	JOOW Action	Elapsed Time (sec)	Advance Distance (m)	Transfer Distance (m)	Heading (deg)
1	Need to make a sail past	Command SOOW to change voyage plan for sail past	Need to change voyage plan for sail past							
2			Need to change voyage plan for sail past							
3			Need to conduct route planning	Conduct route planning on small scale chart						
4	Ship is ready for voyage									
5	Ship is ready for voyage	Command JOOW to lift anchor			Need to lift anchor					
6					Need to lift anchor	Lift anchor				
7	Anchor lifted									
8	Anchor lifted	Command JOOW - Full Ahead			Need to execute command - Full Ahead	Execute command - Full Ahead				
9							6	15	0	0
72			Danger ahead	Declare danger ahead			341	1020	0	0
73	Danger ahead	Command JOOW 10 degree starboard		No Action	Need to execute 10 degree starboard	Execute command - Full Ahead	416	1245	1	0
300							1481	4196	1002	42



Table 5.11: Results of logical deductions of crew perception-actions (large-scale chart chosen for navigation)

Ship following 'Large scale chart'										
Iteration No.	Captain Perception	Captain Action	SOOW Perception	SOOW Action	JOOW Perception	JOOW Action	Elapsed Time (sec)	Advance Distance (m)	Transfer Distance (m)	Heading (deg)
1	Need to make a sail past	Command SOOW to change voyage plan for sail past	Need to change voyage plan for sail past							
2			Need to change voyage plan for sail past							
3			Need to conduct route planning	Conduct route planning on large scale chart						
4	Ship is ready for voyage									
5	Ship is ready for voyage	Command JOOW to lift anchor			Need to lift anchor					
6					Need to lift anchor	Lift anchor				
7	Anchor lifted									
8	Anchor lifted	Command JOOW - Full Ahead			Need to execute command - Full Ahead	Execute command - Full Ahead				
171							836	2505	0	0
172	Danger ahead	Command JOOW 10 degree starboard			Need to execute 10 degree starboard	Execute 10 degree starboard	841	2520	0	0
300							1516	4512	285	21

It is visible in Table 5.10 and Table 5.11 that out of 300 iterations not all iterations are shown. This is because of two reasons. Firstly, due to limited space. Secondly, not all iterations result in significant change in the simulation. For instance, in Table 5.11, iteration number 9 to iteration number 171 there is no change in the perception-action cycle except for the motion of the ship. Therefore, portraying all the iteration steps are unnecessary.

The iterations shown in the tables above provides a glimpse of the activity that takes place during a voyage. Although the results are hypothetical deduction and the knowledge of the agents is very limited, yet the idea presented in this study reveals the potentials of such analysis. It is needless to mention that with the increase in number to ship crew and intricate natural environment the problem space for accident analysis becomes very difficult and goes beyond human comprehension. Therefore, a computational technique as such could extend the capability for detailed accident analysis.

## **5.5 Chapter Summary**

This chapter demonstrated the examples of developing logic worlds and how logic programming technique can deduce possible accident outcomes. The first two examples, Example 1 and Example 2, demonstrates the usage of propositional logic based technique. This technique is very simple but plays an important role for the development of agent based perception-action technique, which is shown in Example 3. In this example the development of agents using list of perceptions and actions are shown. Hence, the results obtained in this chapter show how LPT deduces events that lead to accidents.

## **Chapter 6: Conclusion and Recommendation**

### **6.1 Some Important Features of Logic Programming Technique (LPT)**

In this study propositional logics are utilized to develop logic worlds and find out how accidents take place within the world. Propositions logics are simple sentences which have one or more premise(s) and one conclusion. This simple format allows modelling systems of various disciplines. This is a significant advantage for studying accidents in multidisciplinary platform.

The advancement of propositional logic based technique is the agent based perception-action technique. In this technique all agents search through respective perceptions and actions for given inputs. Thereby, logical deductions reveal all the possible sequence of events that may take place. Some of these events may lead to accidents which is of particular interest to achieve safety. Hence, Logic Programming Technique (LPT) can automatically deduce how accidents take place. Such deductions are generally difficult to make by human judgment alone because of the complexity and size of the problem space.

Traditional accident analysis also becomes very difficult when there is a change in the system; whereas logic programming can handle system changes easily. For example, an addition of an event or an agent is simply done by adding rules in the logic program; while in traditional accident analysis (e.g. fault tree analysis) such change

complicates the total structure (tree in this case) and requires reconstructing the structure again. Also in traditional accident analysis the (e.g. fault tree) the combination of events is known. However, in Logic Programming Technique (LPT) the combinations of events are deduced automatically. This gives Logic Programming Technique (LPT) a significant advantage over other accident analysis techniques.

Logic Programming Technique (LPT) can be utilized to gain hindsight. Hindsight means Understanding the nature of an event after it has happened. It is practically impossible to conduct experiment and study accident occurrence. Simulation is possibly the only way to study and gain hindsight. There is a general agreement that efforts to improve the safety of systems are dominated by hindsight. Which makes sense why safety measures are taken once an accident had taken place. Hollnagel et. al's (2011) book on resilience engineering, therefore, describes the importance of knowing 'how' things go right and 'how' things go wrong beforehand. Logic Programming Technique (LPT) can contribute in this area significantly by simulating all possible outcomes for a given scenario.

Logic Programming Technique (LPT) can be utilized in accident investigation as well. Theoretically, there are two groups of accident investigation methods: deductive methods and inductive methods. Inductive techniques help analysts to piece together the ways in which individual events combine during the course of an adverse event or near miss. The available evidence about these events drives the analysis. The success of an inductive approach, therefore, depends upon investigators gathering sufficient evidence to begin the analysis. Conversely, deductive techniques work back from the adverse event

or potential outcome. Investigators must then trace back a causal sequence to identify initial events. These approaches can be used in situations where evidence is missing or hard to gather (Johnson, 2003). In both of the accident investigation methods Logic Programming Technique (LPT) may contribute if logic worlds are developed and utilized.

There are however, a number challenges ahead for further development of Logic Programming Technique (LPT). For example, construction of logic world may become tedious and difficult at times. The description of the world needs to be precise and not all events may fit it a generalized structure of logic/argument. Secondly, deduction of probable accident outcomes may come in large number which might be impossible to comprehend. Therefore, engineering approaches for practical applications needs to be set up which may avoid infinite possibilities of accident outcomes.

## **6.2 Concluding Remarks**

This research work presented a completely new idea of accident analysis in the maritime perspective. The study reviewed various accident theories and models, which resulted in a conclusion that the problem space for solving accident problems is diverse and incorporation of knowledge from multiple disciplines is necessary. There are various perspectives of accident causation as well. For effective accident analysis and accident prevention this knowledge is indispensable.

An attempt was undertaken to understand the nature and characteristics of maritime accidents, specifically how the accidents unfold. The study revealed that the perceptions and actions of ship crew plays an important role behind accident causation.

Often certain perceptions or actions of ship crew appear logically correct yet those perceptions or actions result in the accident itself. Hence, the necessary and sufficient causes of accidents remain undercover.

The study demonstrated that accident theories are evolving with respect to sociotechnical context. There are wide range of perspectives over the causation of accidents. The literature review also suggests that most accident theories only provide conceptual hypothesis and does not suggest any computational technique for analysis. Therefore, this study attempts to fill in this weakness by suggesting Logic Programming Technique (LPT) for accident analysis. The fundamentals of Logic Programming Technique (LPT) are discussed and examples are given. In this technique, logic or argument is the fundamental element using which the logic worlds were created and analyzed for different cases. The most important characteristics of such model is that it results ‘how’ rather than ‘what’.

Multiple agent perception-action technique is proposed and demonstrated in this study. The technique is theoretically explained and examples are shown using famous maritime accidents. The study suggests that such a technique has the potential of bringing the human, social and technical factors in the same platform and provides the ability to analyze accidents in a single programming language.

In summary it is possible to mention a few unique characteristics of Logic Programming Technique (LPT) as follows:

- i. The concept is simple and elegant as it has the potentials of deducing how accident may take place.
- ii. Models can be developed using simple elemental logics/arguments which allow multiple models from different disciplines to be included in a single program.
- iii. Logic Programming Technique (LPT) can easily handle large problem space which are generally beyond human comprehension and judgment. This feature also gives advantage over other accident analysis techniques.
- iv. This method can also examine the perceptions and actions of ship crew agents in different scenarios where the real ship crew may utilize it by analyzing their possible intentions beforehand and conduct a safer voyage.

### **6.3 Recommendations**

This study shows very simple examples which can be further developed and elaborated in the future. The LPT approach to accident problems is new and appears to have a lot of potentials. Particularly in accident cases where the problem space is very large and complicated, this logic programming technique may become very useful for identifying the necessary and sufficient causes. In this view, the following recommendations are made for the future studies:

- i. Applying Logic Programming Technique (LPT) to Marine Traffic Simulation System (MTSS) is the next logical step and it is believed to be highly beneficial for marine safety analysis.
- ii. A multidisciplinary approach is recommended. Construction (or modelling) of more agents following actual world scenario is necessary. Also enriching

the agent's knowledge with more perception and action arguments will be realistic.

- iii. Utilizing more sophisticated ship maneuvering model where more accurate variables can be incorporated, such as wave, wind, drifting of ship, etc. can yield realistic study.
- iv. Further development of the methodology and framework for such kind of analysis will be interesting. Particularly, innovations in creating logic worlds will be beneficial.
- v. Identifying the barriers for practical application of this technique will be very important.
- vi. The future of this research work seems very promising because of its practical applications and importance in saving lives and resources. Therefore, research and development in this area should be continuously encouraged.



## References

- Analysis and Amoco Cadiz oil spill. (2016). In Wikipedia. Available at: [https://en.wikipedia.org/wiki/Amoco\\_Cadiz\\_oil\\_spill](https://en.wikipedia.org/wiki/Amoco_Cadiz_oil_spill). Accessed: 22<sup>nd</sup> June 2016.
- Awal, Z.I. (2007). A study on inland water transport accidents in Bangladesh: Experience of a decade (1995-2005), *Technical Note at The International Journal of Small Craft Technology (IJSCT)*, Vol. 149, Part B2, pp. 35–42. doi: 10.3940/rina.ijsc.2007.b2.5807.
- Awal, Z.I. and Hasegawa, K. (2013). Bridge Resource Simulator - A New Tool for Ship Accident, *Proceedings of the Japan society of Naval Architects and Ocean Engineers (JASNAOE), 27-28 May 2013, Hiroshima, Japan*, Vol. 16, pp. 51-54.
- Awal, Z.I. and Hasegawa, K. (2014a). Analysis of Marine Accidents by Logic Programming Technique, *Proceedings of the 10th International Symposium on Marine Engineering (ISME), 15-19 September 2014, Harbin, China*, In USB Drive, Paper No. ISME127.
- Awal, Z.I. and Hasegawa, K. (2014b). Application of Logic Programming Technique on Maritime Accident Analysis, *Proceedings of the International Conference on Ship and Offshore Technology (ICSOT 2014), 4-5 November 2014, Makassar, Indonesia*, pp. 59-66.

- Awal, Z.I. and Hasegawa, K. (2015a). Accident analysis by logic programming technique, *Safety and Reliability of Complex Engineered Systems: ESREL 2015*, pp. 13-21.
- Awal, Z.I. and Hasegawa, K. (2015b). A new approach to accident analysis: Multiple agent perception-action, *Proceedings of the 5th World Maritime Technology Conference (WMTC), 3-7 November 2015, Rhode Island, USA*, In CD-ROM.
- Awal, Z.I. and Hasegawa, K. (2015c). Analysis of Marine Accidents Due to Engine Failure – Application of Logic Programming Technique (LPT), Technical Information at the *Journal of the Japan Institute of Marine Engineering (JIME)*, Vol. 50, No., 6, pp. 39 – 46.
- Awal, Z.I. and Hasegawa, K. (In Press). A new approach to accident analysis: Multiple agent perception-action, *Transactions of the Society of Naval Architects and Marine Engineers (SNAME)*.
- Awal, Z.I., Islam, M.R. and Hoque, M.M. (2010). Collision of marine vehicles in Bangladesh: A study on accident characteristics, *Disaster Prevention and Management*, Vol. 19, No. 5, pp. 582–595. doi: 10.1108/09653561011091913.
- Benner, L. (1975). Accident investigations: Multilinear event sequencing methods, *Journal of Safety Research*, Vol. 7, No. 2, pp. 67-73.
- Benner, L. (1985). Rating accident models and investigation methodologies. *Journal of Safety Research*, Vol. 16, No. 3, pp. 105-126.
- Bertalanffy, V.L. (1968). General systems theory. New York: Braziller.
- Botting, R.M. and Johnson, C.W. (1998). A formal and structured approach to the use of task analysis in accident modelling, *International Journal of Human-Computer Studies*, Vol. 49, Issue 3, pp. 223-244.

- Bramer, M.A., 2005. *Logic programming with Prolog*. Springer.
- Branford, K. 2011. Seeing the Big Picture of Mishaps. *Aviation Psychology and Applied Human Factors*, Vol. 1, No. 1, pp. 31–37.
- Burns, C. P. (2000). *Analyzing accident reports using structured and formal methods* (Doctoral dissertation, University of Glasgow).
- Burns, C. P., Johnson, C. W. and Thomas, M. (1997). *Agents and actions: Structuring human factors accounts of major accidents*. Technical Report TR-1997-32, Department of Computing Science, October, Glasgow: University of Glasgow.
- Chambers, E.G. and Yule, G.U. (1941). Theory and observation in the investigation of accident causation. *Supplement to the Journal of the royal statistical society*, Vol. 7, No. 2, pp. 89-109.
- Collins, R. J., and Thompson, R. (1997). Systemic failure modes: a model for Perrow's normal accidents in complex, safety critical systems. *Advances in safety and reliability*, pp. 357-364.
- Costa Concordia Disaster. (2016). In Wikipedia. Available at: [https://en.wikipedia.org/wiki/Costa\\_Concordia\\_disaster](https://en.wikipedia.org/wiki/Costa_Concordia_disaster) (Accessed: 23 June 2016).
- Cresswell, W.L. and Froggatt, P. (1962). Accident Proneness, or Variable Accident Tendency? *Journal of the statistical and social inquiry Society of Ireland*, Vol. XX, No. V, pp. 152–171.
- DeJoy, D.M. (1994). Managing safety in the workplace: An attribution theory analysis and model, *Journal of Safety Research*, Vol. 25, No. 1, pp. 3-17.
- Derbyshire, D. (2012). So what DID cause the liner to hit the rocks? Human error, electrical failure and uncharted obstruction are all theories to be investigated. In

- Daily Mail. Available at: <http://www.dailymail.co.uk/news/article-2087133/Costa-Concordia-accident-So-DID-cause-cruise-ship-hit-rocks.html>. Accessed: 22 June 2016.
- Dutch Safety Board. (2013). *Fatal accident on board Planet V during emergency anchoring*, The Hague.
- Early, C. (2016). March 6, 1987: Zeebrugge ferry disaster claims 193 lives. Available at: <http://home.bt.com/news/uk-news/march-6-1987-zeebrugge-ferry-disaster-claims-193-lives-11363966092455>. Accessed: 22 June 2016.
- Ericson, C.A. (1999). Fault Tree analysis – A History, *Proceedings of the 17th International System Safety Conference, Florida, USA*, pp. 1-9.
- Flach, P.A. (1994). *Simply Logical intelligent reasoning by example*.
- Froggatt, P. and Smiley, J.A. (1964). The concept of accident proneness: A review, *British journal of industrial medicine*, Vol. 21, No. 1, pp. 1-12.
- Gordon, J.E. (1949). The epidemiology of accidents, *American Journal of Public Health*, Vol. 39, No. 4, pp. 504-515.
- Greenwood, M. (1950). Accident Proneness, *Biometrika*, Vol. 37, No. 1/2, pp. 24-29.
- Greenwood, M. and Woods, H.M. (1919). *The incidence of industrial accidents upon individuals*. Report No. 4, Industrial Fatigue Research Board, UK.
- Greenwood, M. and Yule, G.U. (1920). An inquiry into the nature of frequency distributions representative of multiple happenings with particular reference to the occurrence of multiple attacks of disease or of repeated accidents. *Journal of the royal statistical society*, Vol. 83, No. 2, pp. 255-279.
- Haddon, W. (1968). The changing approach to the epidemiology, prevention, and amelioration of trauma: the transition to approaches etiologically rather than

- descriptively based. *American journal of public health and the nation's health*, Vol. 58, No. 8, pp. 1431–1438.
- Haddon, W. (1970). On the escape of tigers: an ecologic note. (Strategy options in reducing losses in energy-damaged people and property). *Technology Review (MIT)*, Vol. 72, pp. 44-53.
- Hamilos, P. (2013). Spanish government cleared of blame for prestige oil tanker disaster. In Guardian. Available at: <https://www.theguardian.com/world/2013/nov/13/spanish-prestige-oil-tanker-disaster>. Accessed: 22 June 2016.
- Harms-Ringdahl, L. (2004). Relationships between accident investigations, risk analysis, and safety management. *Journal of hazardous materials*, Vol. 111, No. 1, pp. 13-19.
- Harms-Ringdahl, L. (2009). Analysis of safety functions and barriers in accidents. *Safety Science*, Vol. 47, No. 3, pp. 353-363.
- Hasegawa, K. and Awal, Z.I. (2013). A Concept for Expert System Based Accident Prediction Technique for Ship Maneuvering, *Proceedings of the 5<sup>th</sup> International Conference on Design for Safety (IDFS)*, 25-27 November 2013, Shanghai, China, In USB Drive.
- Heinrich, H.W. (1931). *Industrial accident prevention*. New York: McGraw-Hill.
- Heinrich, H.W. (1931). *Industrial accident prevention: a scientific approach*. McGraw-Hill.
- Heinrich, H.W. 1929. Relation of accident statistics to industrial accident prevention. *Proceedings of the Casualty Actuarial Society*, Vol. XVI, No. 33, pp. 170-174.

- Hendrick, K. and Benner, L. (1987). *Investigating Accidents with STEP*, Marcel Dekker: New York.
- Hollnagel, E. (1998). *Cognitive reliability and error analysis method (CREAM)*. Elsevier.
- Hollnagel, E. (2001). Anticipating failures: what should predictions be about? *Linköping university (Sweden) graduate school for human-machine interaction*.
- Hollnagel, E. (2008). 'Risk + barriers = safety?', *Safety Science*, Vol. 46, No. 2, pp. 221–229. doi: 10.1016/j.ssci.2007.06.028.
- Hollnagel, E. (2013a). *An Application of the Functional Resonance Analysis Method (FRAM) to Risk Assessment of Organizational Change*. Swedish Radiation Safety Authority, Report number 9.
- Hollnagel, E. (2013b). Safety-I and Safety-II: A new perspective on patient safety, In Osaka University. Available at: [http://www.hosp.med.osaka-u.ac.jp/home/hp-cqm/ingai/seminar/pdf/2013/012-1\\_hollnagel\\_SouthernDenmark.pdf](http://www.hosp.med.osaka-u.ac.jp/home/hp-cqm/ingai/seminar/pdf/2013/012-1_hollnagel_SouthernDenmark.pdf). Accessed: 22 June 2016.
- Hollnagel, E., & Goteman, O. (2004). The functional resonance accident model. *Proceedings of cognitive system engineering in process plant*, pp. 155-161.
- Hollnagel, E., Woods, D.D., Leveson, N.G. (Eds.), 2006. *Resilience Engineering: Concepts and Precepts*. Ashgate Publishing Limited, Aldershot, UK.
- Hossain, M.T., Awal, Z.I. and Das, S. (2010). Reconstruction of capsized type of accidents by fault tree analysis, *Proceedings of the International Conference on Marine Technology (MARTEC), 11-12 December, 2010 Dhaka, Bangladesh*, pp. 445-452.

- Hossain, M.T., Awal, Z.I. and Das, S. (2014). A Study on the Accidents of Inland Water Transport in Bangladesh: The Transportation System and Contact Type Accidents, *Journal of transport system engineering*, Vol. 1, No. 1, pp. 23-32.
- Johnson, C. and Holloway, C. M. (2003). A survey of logic formalisms to support mishap analysis. *Reliability Engineering & System Safety*, Vol. 80, Issue 3, pp. 271-291.
- Johnson, C. W. (2000). Proving properties of accidents. *Reliability Engineering & System Safety*, Vol. 67, Issue 2, pp. 175-191.
- Johnson, C.W. (1997). The epistemics of accidents. *International Journal of Human-Computer Studies*, Vol. 47, Issue 5, pp. 659-688.
- Johnson, C.W., McCarthy, J.C. and Wright, P.C. (1995). Using a formal language to support natural language in accident reports. *Ergonomics*, Vol. 38, Issue 6, pp. 1264-1282.
- Journée, J.M.J. and Pinkster, J. (2002). *Introduction in Ship Hydrodynamics*.
- Keefe, P. (2014). Disasters at sea & their impact on shipping regulation. In Marine Link. Available at: <http://www.marinelink.com/news/disasters-shipping-impact371542.aspx>. Accessed: 29 June 2016.
- Khanzode, V. V., Maiti, J., and Ray, P. K. (2012). Occupational injury and accident research: A comprehensive review. *Safety Science*, Vol. 50, No. 5, pp. 1355-1367.
- Kjellen, U. (1984a). The deviation concept in occupational accident control - I. *Accident analysis and prevention*, Vol. 16, No. 4, pp. 289-306.
- Kjellen, U. (1984b). The deviation concept in occupational accident control - II. *Accident analysis and prevention*, Vol. 16, No. 4, pp. 307-323.

- Kjellen, U. and Larsson, T.J. (1981). Investigating accidents and reducing risks - A dynamic approach. *Journal of occupational accidents*, Vol. 3, Issue 2, pp. 129-140.
- Koester, T. (2002). Human factors and everyday routine in the maritime work domain. *In: de Waard, D., Brookhuis, K. A., Moraal, J. & Toffetti, A. (eds.). Human Factors in Transportation, Communication, Health, and the Workplace*. Shaker Publishing, pp. 361-375.
- Kossoris, M.D. (1939). A statistical approach to accident prevention. *Journal of the American statistical association*, Vol. 34, No. 207, pp. 524-532.
- Laflamme, L., (1990). A better understanding of occupational accident genesis to improve safety in the workplace. *Journal of occupational accidents*, Vol. 12, Issues 1-3, pp. 155-165.
- Leveson, N. (2004). A new accident model for engineering safer systems. *Safety Science*, Vol. 42, No. 4, pp. 237-270.
- Leveson, N. (2015). A systems approach to risk management through leading safety indicators. *Reliability Engineering & System Safety*, Vol. 136, pp. 17-34.
- Li, S., Meng, Q. and Qu, X. (2012). An overview of maritime waterway quantitative risk assessment models. *Risk Analysis*, Vol. 32, No. 3, pp. 496-512.
- Lieto, A. D. (2012). *Costa Concordia Anatomy of an Organizational Accident*. Available from: <https://maddenmaritime.files.wordpress.com/2012/07/costaconcordiaanatomyofanorganisationalaccident.pdf>. Accessed: 10<sup>th</sup> June 2016.



- Loimer, H., Driur, M. and Guarnieri, M. (1996). Accidents and Acts of God: A History of the Terms. *American journal of public health*, Vol. 86, No. 1, pp. 101-107.
- Marine Casualties Investigative Body, Italy. (2013). *Cruise Ship COSTA CONCORDIA Marine casualty on January 13, 2012 - Report on the safety technical investigation*, Italy: Ministry of Infrastructures and Transports.
- Mastin, L. (2008). *Logic - by branch / doctrine - the basics of philosophy*. Available at: [http://www.philosophybasics.com/branch\\_logic.html](http://www.philosophybasics.com/branch_logic.html). Accessed: 27<sup>th</sup> June 2016.
- McFalls, J.A. (2003). *Population: A lively introduction (4<sup>th</sup> Edition)*, Vol. 58, No. 4, Washington, DC: Population Reference Bureau.
- NASA Safety Center, (2010). Brace for Impact, *System Failure Case Studies, National Aeronautics and Space Administration (NASA)*, Volume 4, Issue 10, pp. 1-4.
- National Transportation Safety Board (NTSB), (1998). *Marine Accident Report*, PB98-916401, NTSB/MAR-98/01.
- Newbold, E.M. (1927). Practical applications of the statistics of repeated events' particularly to industrial accidents. *Journal of the royal statistical society*, Vol. 80, No. 3, pp. 487-547.
- Nomoto, K., Taguchi, K., Honda, K. and Hirano, S. (1957). T, *International Shipbuilding Progress*, Vol. 4, pp. 354-370.
- Oil spills*. (2013). Available at: <http://oilspillswiki.pbworks.com/w/page/64031584/Oil%20Spills>. Accessed: 22<sup>nd</sup> June 2016.

Perrow, C., 1984. *Normal Accidents - Living with high risk technologies*, Basic Books: New York.

*Piper Alpha*. (2016). In Wikipedia. Available at:  
[https://en.wikipedia.org/wiki/Piper\\_Alpha#Timeline\\_of\\_the\\_incident](https://en.wikipedia.org/wiki/Piper_Alpha#Timeline_of_the_incident).  
Accessed: 22<sup>nd</sup> June 2016.

Qureshi, Z.H. (2007). A review of accident modelling approaches for complex socio-technical systems. In *Proceedings of the twelfth Australian workshop on Safety critical systems and software and safety-related programmable systems*, Vo. 86, pp. 47-59.

Rasmussen, J. (1997). Risk management in a dynamic society: a modelling problem. *Safety science*, Vol. 27, No. 2, pp.183-213.

Rasmussen, J., and Svedung, I. (2000). *Proactive risk management in a dynamic society*. Swedish Rescue Services Agency.

Reason, J. (1990). The contribution of latent human failures to the breakdown of complex systems. *Philosophical Transactions of the Royal Society London B*, Vol. 327, pp. 475-484.

Reason, J., (1997). *Managing the risks of organizational accidents*. Aldershot: Ashgate.

Rieder, R. and Bepperling, S.L. (2011). Heinrich Triangle for Ground Operation. *Journal of System Safety*, pp. 23 - 28.

Rodgers, M.D. and Blanchard, R.E. (1993). *Accident proneness: A research review*, FAA Civil Aeromedical Institute, Report No. DOT/FAA/AM-93/9.

- Russell, S.J., Norvig, P., Canny, J.F., Malik, J.M. and Edwards, D.D. (2010). *Artificial intelligence: a modern approach (3<sup>rd</sup> Edition)*. Upper Saddle River: Prentice hall.
- Sammarco, J.J., (2005). Operationalizing normal accident theory for safety-related computer systems. *Safety Science*, Vol. 43, No. 9, pp. 697-714.
- Schröder-Hinrichs, J. U., Hollnagel, E., and Baldauf, M. (2012). From Titanic to Costa Concordia-a century of lessons not learned. *WMU Journal of Maritime Affairs*, Vol. 11, No. 2, pp. 151–167. doi.org/10.1007/s13437-012-0032-3.
- Sklet, S. (2002). *Methods for accident investigation*. Trondheim: Gnist Tapir.
- Sklet, S. (2004). Comparison of some selected methods for accident investigation, *Journal of hazardous materials*, Vol. 111, No. 1, pp. 29-37.
- Spivey, M. (1996). *An introduction to logic programming through Prolog*. London: Prentice-Hall.
- SS Morro castle (1930). (2016). In Wikipedia. Available at: [https://en.wikipedia.org/wiki/SS\\_Morro\\_Castle\\_\(1930\)](https://en.wikipedia.org/wiki/SS_Morro_Castle_(1930)). Accessed: 22<sup>nd</sup> June 2016.
- Suchman, E.A. (1970). Accidents and social deviance. *Journal of health and social behavior*. Vol. 11, No. 1, pp. 4-15.
- Tuominen, R. and Saari, J. (1982). A model for analysis of accidents and its application. *Journal of occupational accidents*, Vol. 4, Issues 2-4, pp. 263-273.
- Tzeng, C.W. and Chen, J.F. (1999). Fundamental Properties of Linear Ship Steering Dynamic Models, *Journal of marine science and technology (JMST)*, Vol. 7, No. 2, pp. 79-88.

- Umeda, N., Usada, S., Mizumoto, K. and Matsuda, A. (2016). Broaching probability for a ship in irregular stern-quartering waves: theoretical prediction and experimental validation. *Journal of Marine Science and Technology*, Vol. 21, No. 1, pp. 23-37.
- Underwood, P., and Waterson, P. (2013). *Accident analysis models and methods: guidance for safety professionals*. Loughborough University, England.
- Underwood, P., and Waterson, P., (2014). Systems thinking, the Swiss Cheese Model and accident analysis: a comparative systemic analysis of the Grayrigg train derailment using the ATSB, AcciMap and STAMP models. *Accident analysis & prevention*, Vol. 68, pp. 75-94.
- Vernez, D., Buchs, D. and Pierrehumbert, G. (2003). Perspectives in the use of colored Petri nets for risk analysis and accident modelling. *Safety Science*, Vol. 41, No. 5, pp. 445-463.
- Vernon, H.M. (1918). *An investigation of the factors concerned in the causation of industrial accident*. Memorandum No. 21, Ministry of Munitions, UK.
- Vicente, K.J. and Christoffersen, K. (2006). The Walkerton E. coli outbreak: a test of Rasmussen's framework for risk management in a dynamic society. *Theoretical Issues in Ergonomics Science*, Vol. 7, No. 2, pp. 93-112.
- Ward, R.B. (2012). Revisiting Heinrich's law. *Chemeca 2012: Quality of life through chemical engineering, 23-26 September 2012, Wellington, New Zealand*, pp. 1179-1187.
- Watson, H.A. (1961). *Launch Control Safety Study*, Bell Labs, Murray Hill, NJ.

- Westrum, R. (2006). A typology of resilience situations. *In: Hollnagel, E., Woods, D.D., Leveson, N.G. (Eds.), Resilience Engineering: Concepts and Precepts.* Ashgate Publishing Limited, Aldershot, UK, pp. 55-66.
- Woods, D.D., Johannesen, L.J., Cook, R.I. and Starter, N.B. (1994). *Behind human error: Cognitive systems, computers and hindsight.* Report Number CSERIAC SOAR 94-01, University of Dayton Research Institute, Ohio.

## Appendix A: Prolog Code of Monkey Banana Problem

```
% =====
% The monkey-banana problem
% Zobair Ibn Awal, D3, Hasegawa Laboratory
% 12 July 2016
% =====

% These are the valid positions of the monkey
monkey_position(position_1, down).      % On the ground
monkey_position(position_2, down).      % On the ground
monkey_position(position_3, down).      % On the ground
monkey_position(position_1, up).         % On the tool
monkey_position(position_2, up).         % On the tool
monkey_position(position_3, up).         % On the tool

% These are the valid moves of the monkey
monkey_move(position_3, position_2, down). % Along the ground
monkey_move(position_2, position_1, down). % Along the ground
monkey_move(position_1, position_2, down). % Along the ground
monkey_move(position_2, position_3, down). % Along the ground

% These are the valid positions of the tool
tool_position(position_1).
tool_position(position_2).
tool_position(position_3).

% These are the valid positions of the banana
banana_position(position_1, down).
banana_position(position_2, down).
banana_position(position_3, down).
banana_position(position_1, up).
banana_position(position_2, up).
banana_position(position_3, up).

% The monkey can grab the banana
trygrab:-
    append('Results.txt'),
    nl,
    write('The monkey attempts to grab ... '),
    told,

    see('monkey_position.txt'),
    read(MonkeyX),
    read(MonkeyY),
    seen,

    see('banana_position.txt'),
    read(BananaX),
    read(BananaY),
    seen,

    monkey_position(MonkeyX, MonkeyY),
    banana_position(BananaX, BananaY),

    MonkeyX == BananaX,
    MonkeyY == BananaY,

    append('Results.txt'),
    nl,
```

```

write('++++ The monkey succeeds to grab the banana !!! ++++'),
nl,
write(' -> The monkey is in '),
write(MonkeyX),
write(', '),
write(MonkeyY),
nl,

write(' -> The banana is in '),
write(BananaX),
write(', '),
write(BananaY),
told.

% The monkey can make a move
make_a_move:-
    see('monkey_position.txt'),
    read(FromX),
    read(FromY),
    seen,

    FromY == down,
    monkey_move(FromX, ToX, down),

    append('Results.txt'),
    nl,
    write('The monkey attempts to make a move ... '),
    told,

    tell('monkey_position.txt'),
    write(ToX),
    write('.'),
    nl,
    write('down.'),
    told,

    append('Results.txt'),
    nl,
    write(' -> Success: The monkey moves from '),
    write(FromX),
    write(' to '),
    write(ToX),
    told.

% The monkey can make a push
make_a_push:-
    append('Results.txt'),
    nl,
    write('The monkey attempts to make a push ... '),
    told,

    see('monkey_position.txt'),
    read(FromX),
    read(FromY),
    seen,

    FromY == down,
    monkey_position(FromX, down),

    see('tool_position.txt'),
    read(ToolX),
    seen,

```

```

    tool_position(ToolX),

    FromX == ToolX,

    monkey_move(FromX, ToX, down),
    tool_position(ToX),

    tell('monkey_position.txt'),
    write(ToX),
    write('.'),
    nl,
    write('down.'),
    told,

    tell('tool_position.txt'),
    write(ToX),
    write('.'),
    told,

    append('Results.txt'),
    nl,
    write(' -> Success: The monkey pushes the tool from '),
    write(FromX),
    write(' to '),
    write(ToX),
    told.

% The monkey can climb up
climb_up:-
    append('Results.txt'),nl,write('The monkey attempts to climb up ... '),told,

    see('monkey_position.txt'),
    read(MonkeyX),
    read(MonkeyY),
    seen,

    see('tool_position.txt'),
    read(ToolX),
    seen,

    MonkeyX == ToolX,

    tell('monkey_position.txt'),
    write(MonkeyX),
    write('.'),
    nl,
    write('up.'),
    told,

    append('Results.txt'),
    nl,
    write(' -> Success: The monkey ascends from '),
    write(MonkeyY),
    write(' to '),
    write('up'),
    told.

% The monkey can climb down
climb_down:-
    append('Results.txt'),nl,write('The monkey attempts to climb down ... '),told,

```



```

see('monkey_position.txt'),
read(MonkeyX),
read(MonkeyY),
seen,

monkey_position(MonkeyX, MonkeyY),
MonkeyY == up,

tell('monkey_position.txt'),
write(MonkeyX),
write('.'),
nl,
write('down.'),
told,

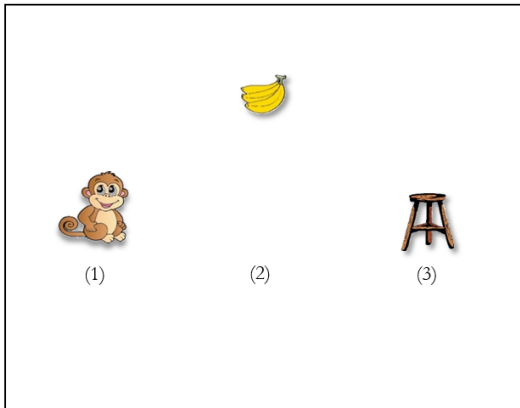
append('Results.txt'),
nl,
write('  -> Success: The monkey descends from '),
write(MonkeyY),
write(' to '),
write('down'),
told.

% Let the monkey do what it can do
gomonkey:-
    trygrab,!;
    make_a_move, trygrab, !;
    make_a_move, climb_up, trygrab, !;
    climb_down, make_a_push, climb_up, trygrab, !;
gomonkey.

```

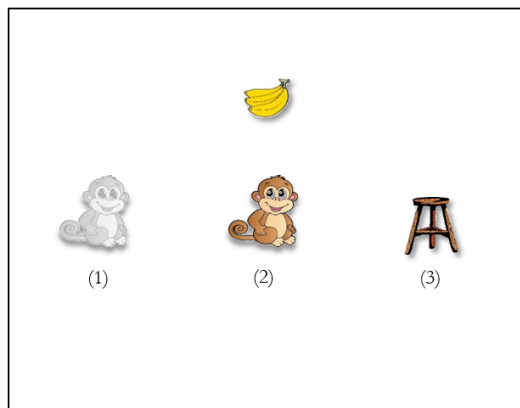
## Appendix B: Solution of Monkey Banana Problem

Search predicate in prolog	Results
<pre> 1. gomonkey:- 2.   trygrab,!, 3.   make_a_move, trygrab, !; 4.   make_a_move, climb_up, trygrab, !; 5.   climb_down, make_a_push, climb_up, trygrab, !; 6.   gomonkey. </pre>	<pre> 1. The monkey attempts to grab ... 2. The monkey attempts to make a move ... 3. -&gt; Success: The monkey moves from position_1 to position_2 4. The monkey attempts to grab ... 5. The monkey attempts to make a move ... 6. -&gt; Success: The monkey moves from position_2 to position_1 7. The monkey attempts to climb up ... 8. The monkey attempts to make a move ... 9. -&gt; Success: The monkey moves from position_2 to position_3 10. The monkey attempts to climb up ... 11. -&gt; Success: The monkey ascends from down to up 12. The monkey attempts to grab ... 13. The monkey attempts to climb down ... 14. -&gt; Success: The monkey descends from up to down 15. The monkey attempts to make a push ... 16. -&gt; Success: The monkey pushes the tool from position_3 to position_2 17. The monkey attempts to climb up ... 18. -&gt; Success: The monkey ascends from down to up 19. The monkey attempts to grab ... 20. +++ The monkey succeeds to grab the banana !!! +++ 21. -&gt; The monkey is in position_2, up 22. -&gt; The banana is in position_2, up </pre>



### Step 1

Search predicate in prolog	Results
<pre> 1. gomonkey:- 2.   trygrab,!, 3.   make_a_move, trygrab, !; 4.   make_a_move, climb_up, trygrab, !; 5.   climb_down, make_a_push, climb_up, trygrab, !; 6.   gomonkey. </pre>	<pre> 1. The monkey attempts to grab ... 2. The monkey attempts to make a move ... 3. -&gt; Success: The monkey moves from position_1 to position_2 4. The monkey attempts to grab ... 5. The monkey attempts to make a move ... 6. -&gt; Success: The monkey moves from position_2 to position_1 7. The monkey attempts to climb up ... 8. The monkey attempts to make a move ... 9. -&gt; Success: The monkey moves from position_2 to position_3 10. The monkey attempts to climb up ... 11. -&gt; Success: The monkey ascends from down to up 12. The monkey attempts to grab ... 13. The monkey attempts to climb down ... 14. -&gt; Success: The monkey descends from up to down 15. The monkey attempts to make a push ... 16. -&gt; Success: The monkey pushes the tool from position_3 to position_2 17. The monkey attempts to climb up ... 18. -&gt; Success: The monkey ascends from down to up 19. The monkey attempts to grab ... 20. +++ The monkey succeeds to grab the banana !!! +++ 21. -&gt; The monkey is in position_2, up 22. -&gt; The banana is in position_2, up </pre>



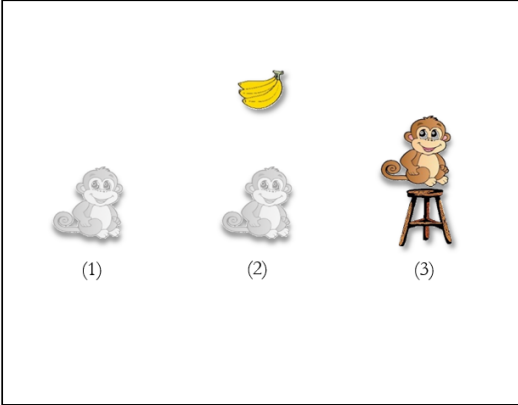
### Step 2

**Search predicate in prolog**

1. gomonkey:-
2. trygrab, !;
3. make\_a\_move, trygrab, !;
4. make\_a\_move, climb\_up, trygrab, !;
5. climb\_down, make\_a\_push, climb\_up, trygrab, !;
6. gomonkey.

**Results**

1. The monkey attempts to grab ...
2. The monkey attempts to make a move ...
3. -> Success: The monkey moves from position\_1 to position\_2
4. The monkey attempts to grab ...
5. The monkey attempts to make a move ...
6. -> Success: The monkey moves from position\_2 to position\_1
7. The monkey attempts to climb up ...
8. The monkey attempts to make a move ...
9. -> Success: The monkey moves from position\_2 to position\_3
10. The monkey attempts to climb up ...
11. -> Success: The monkey ascends from down to up
12. The monkey attempts to grab ...
13. The monkey attempts to climb down ...
14. -> Success: The monkey descends from up to down
15. The monkey attempts to make a push ...
16. -> Success: The monkey pushes the tool from position\_3 to position\_2
17. The monkey attempts to climb up ...
18. -> Success: The monkey ascends from down to up
19. The monkey attempts to grab ...
20. ++++ The monkey succeeds to grab the banana !!! ++++
21. -> The monkey is in position\_2, up
22. -> The banana is in position\_2, up



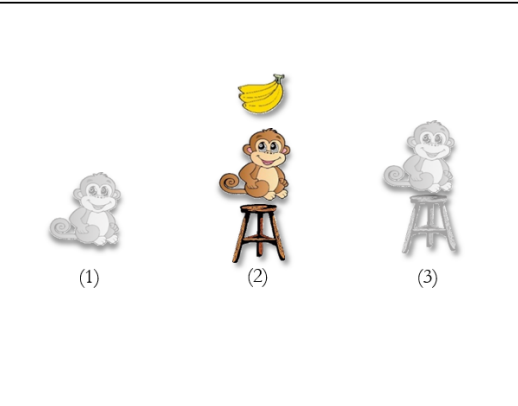
### Step 3

**Search predicate in prolog**

1. gomonkey:-
2. trygrab, !;
3. make\_a\_move, trygrab, !;
4. make\_a\_move, climb\_up, trygrab, !;
5. climb\_down, make\_a\_push, climb\_up, trygrab, !;
6. gomonkey.

**Results**

1. The monkey attempts to grab ...
2. The monkey attempts to make a move ...
3. -> Success: The monkey moves from position\_1 to position\_2
4. The monkey attempts to grab ...
5. The monkey attempts to make a move ...
6. -> Success: The monkey moves from position\_2 to position\_1
7. The monkey attempts to climb up ...
8. The monkey attempts to make a move ...
9. -> Success: The monkey moves from position\_2 to position\_3
10. The monkey attempts to climb up ...
11. -> Success: The monkey ascends from down to up
12. The monkey attempts to grab ...
13. The monkey attempts to climb down ...
14. -> Success: The monkey descends from up to down
15. The monkey attempts to make a push ...
16. -> Success: The monkey pushes the tool from position\_3 to position\_2
17. The monkey attempts to climb up ...
18. -> Success: The monkey ascends from down to up
19. The monkey attempts to grab ...
20. ++++ The monkey succeeds to grab the banana !!! ++++
21. -> The monkey is in position\_2, up
22. -> The banana is in position\_2, up



### Step 4

## Appendix C: Example 1 - Propositional Logic Based Technique

```
% =====  
% Example 1 - Propositional Logic Based Technique  
% -----  
% Zobair Ibn Awal, Doctoral Student (3rd Year), Hasegawa Laboratory  
% Department of Naval Architecture & Ocean Engineering (NAOE)  
% Division of Global Architecture, Graduate School of Engineering  
% Osaka University, Suita Campus, Osaka, Japan  
% =====  
% World is a system which consists of the following objects  
% 1. Ship A, 2. Ship B, 3. Rocks  
% =====  
fact('Ship A is sailing.').  
fact('Ship B is sailing.').  
fact('Dangerous underwater rocks are nearby.').  
% -----  
% Object 1: Ship A  
% Attributes:  
logic(Conclusion, Premise1, __, __):-  
    Premise1 = 'Captain of Ship A commands head north-east.',  
    fact('Ship A is sailing.').  
    see('ShipA_Heading.txt'),  
    read(HeadingShipA),  
    seen,  
    HeadingShipA == 'Ship A heads north.',  
    Conclusion = 'Ship A heads north-east.',  
    tell('ShipA_Heading.txt'),  
    write(''),  
    write(Conclusion),  
    write(''),  
    write('.'),  
    told,  
    nl,  
    write(Conclusion),  
    write(' Because:'),  
    nl,  
    write('      1. '),  
    write(Premise1),  
    nl;  
    Premise1 = 'Captain of Ship A commands head south-east.',  
    fact('Ship A is sailing.').  
    see('ShipA_Heading.txt'),  
    read(HeadingShipA),
```

```

        seen,
        HeadingShipA == 'Ship A heads south.',
Conclusion = 'Ship A heads south-east.',
tell('ShipA_Heading.txt'),
write(''),
write(Conclusion),
write(''),
write('.'),
told,
nl,
write(Conclusion),
write(' Because:'),
nl,
write('      1. '),
write(Premise1),
nl.

% Object 2: Ship B
% Attributes:
logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Captain of Ship B commands head north-west.',
    fact('Ship B is sailing.'),
    see('ShipB_Heading.txt'),
    read(HeadingShipB),
    seen,
    HeadingShipB == 'Ship B heads north.',
Conclusion = 'Ship B heads north-west.',
tell('ShipB_Heading.txt'),
write(''),
write(Conclusion),
write(''),
write('.'),
told,
nl,
write(Conclusion),
write(' Because:'),
nl,
write('      1. '),
write(Premise1),
nl;
Premise1 = 'Captain of Ship B commands head south-west.',
fact('Ship B is sailing.'),
see('ShipB_Heading.txt'),
read(HeadingShipB),
seen,
HeadingShipB == 'Ship B heads south.',
Conclusion = 'Ship B heads south-west.',
tell('ShipB_Heading.txt'),
write(''),
write(Conclusion),
write(''),

```

```

        write('.'),
        told,
        nl,
        write(Conclusion),
        write(' Because:'),
        nl,
        write('      1. '),
        write(Premise1),
        nl.

% -----
% Relationship among objects
% -----
logic(Conclusion, Premise1, Premise2, _):-
    Premise1 = 'Ship A heads south-east.',
    Premise2 = 'Dangerous underwater rocks are nearby.',
    fact('Ship A is sailing.'),
    fact('Dangerous underwater rocks are nearby.'),
    see('ShipA_Heading.txt'),
    read(HeadingShipA),
    seen,
    HeadingShipA == Premise1,

    Conclusion = 'Ship A is on a grounding course.',
    nl,
    write(Conclusion),
    write(' Because: '),
    nl,
    write('      1. '),
    write(Premise1),
    nl,
    write('      2. '),
    write(Premise2),
    nl.

logic(Conclusion, Premise1, Premise2, _):-
    Premise1 = 'Ship B heads south-west.',
    Premise2 = 'Dangerous underwater rocks are nearby.',
    fact('Ship B is sailing.'),
    fact('Dangerous underwater rocks are nearby.'),
    see('ShipB_Heading.txt'),
    read(HeadingShipB),
    seen,
    HeadingShipB == Premise1,
    Conclusion = 'Ship B is on a grounding course.',
    nl,
    write(Conclusion),
    write(' Because: '),
    nl,
    write('      1. '),
    write(Premise1),

```

```

        nl,
        write('      2. '),
        write(Premise2),
        nl.

logic(Conclusion, Premise1, Premise2, _):-
    Premise1 = 'Ship A heads east.',
    Premise2 = 'Ship B heads west.',
    fact('Ship A is sailing.'),
    fact('Ship B is sailing.'),
    see('ShipA_Heading.txt'),
    read(HeadingShipA),
    seen,
    HeadingShipA == Premise1,
    see('ShipB_Heading.txt'),
    read(HeadingShipB),
    seen,
    HeadingShipB == Premise2,
    Conclusion = 'Ship A and Ship B are in a collision course.',
    nl,
    write(Conclusion),
    write(' Because: '),
    nl,
    write('      1. '),
    write(Premise1),
    nl,
    write('      2. '),
    write(Premise2),
    nl.

% -----
% Defining accidents
% -----

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Ship A is on a grounding course.',
    fact('Ship A is sailing.'),
    fact('Dangerous underwater rocks are nearby.'),
    see('ShipA_Heading.txt'),
    read(HeadingShipA),
    seen,
    HeadingShipA == 'Ship A heads south-east.',
    Conclusion = 'There will be an accident.',
    nl,
    write(Conclusion),
    write(' This is because: '),
    nl,
    write('      1. '),
    write(Premise1),
    nl.

logic(Conclusion, Premise1, _, _):-

```

```

Premise1 = 'Ship B is on a grounding course.',
fact('Ship B is sailing.'),
fact('Dangerous underwater rocks are nearby.'),
see('ShipB_Heading.txt'),
read(HeadingShipB),
seen,
HeadingShipB == 'Ship B heads south-west.',
Conclusion = 'There will be an accident.',
nl,
write(Conclusion),
write(' This is because: '),
nl,
write('      1. '),
write(Premise1),
nl.

logic(Conclusion, Premise1, _, _):-
Premise1 = 'Ship A and Ship B are in a collision course.',
fact('Ship A is sailing.'),
fact('Ship B is sailing.'),
see('ShipA_Heading.txt'),
read(HeadingShipA),
seen,
HeadingShipA == 'Ship A heads east.',
see('ShipB_Heading.txt'),
read(HeadingShipB),
seen,
HeadingShipB == 'Ship B heads west.',
Conclusion = 'There will be an accident.',
nl,
write(Conclusion),
write(' This is because: '),
nl,
write('      1. '),
write(Premise1),
nl.

how:-
logic(C, P1, P2, P3).

```



## Appendix D: Example 2 - Propositional Logic Based Technique

```
% =====  
% Example 2 - Propositional Logic Based Technique  
% -----  
% Zobair Ibn Awal, Doctoral Student (2nd Year), Hasegawa Laboratory  
% Department of Naval Architecture & Ocean Engineering (NAOE)  
% Division of Global Architecture, Graduate School of Engineering  
% Osaka University, Suita Campus, Osaka, Japan  
% =====  
fact('Ship has speed.').  
%fact('Ground is nearby.').  
%fact('Rudder is functional.').  
fact('Another ship is in collision course.').  
fact('Commanded to shutdown auxiliary generators.').  
fact('Faulty regulator.').  
  
logic(Conclusion, Premise1, Premise2, Premise3):-  
    Premise1 = 'Ground is nearby.',  
    Premise2 = 'Ship has speed.',  
    Premise3 = 'Ship is uncontrollable.',  
    Conclusion = 'Ship will hit ground.',  
  
    fact(Premise1),  
    fact(Premise2),  
  
    nl,  
    write(Conclusion), write(' This is because of the following premises: '),  
    nl,  
    write('      1. '), write(Premise1), nl,  
    write('      2. '), write(Premise2), nl,  
    write('      3. '), write(Premise3).  
  
logic(Conclusion, Premise1, Premise2, _):-  
    Premise1 = 'Engine not delivering enough power.',  
    Premise2 = 'Rudder is not functional.',  
    Conclusion = 'Ship is uncontrollable.',  
  
    fact(Premise2),  
    fact('Ground is nearby.').  
    fact('Ship has speed.').  
  
    nl,  
    write(Conclusion), write(' This is because of the following premises: '),  
    nl,  
    write('      1. '), write(Premise1), nl,  
    write('      2. '), write(Premise2), nl.
```

```

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Engine not delivering enough power.',
    Conclusion = 'Ship is uncontrollable.',

    fact('Ground is nearby.'),
    fact('Ship has speed.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1).

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Rudder is not functional.',
    Conclusion = 'Ship is uncontrollable.',

    fact(Premise1),
    fact('Ground is nearby.'),
    fact('Ship has speed.'),

    nl,
    write(Conclusion), write(' This is because of the following premises: '),
    nl,
    write('      1. '), write(Premise1), nl.

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Engine automatic shutdown.',
    Conclusion = 'Engine not delivering enough power.',

    fact('Ground is nearby.'),
    fact('Ship has speed.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Lubricating oil pressure low.',
    Conclusion = 'Engine automatic shutdown.',

    fact('Ground is nearby.'),
    fact('Ship has speed.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Lubricating oil pump fails.',
    Conclusion = 'Lubricating oil pressure low.',

    fact('Ground is nearby.'),
    fact('Ship has speed.'),

```

```

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

logic(Conclusion, Premise1, Premise2, Premise3):-
    Premise1 = 'Ship has speed.',
    Premise2 = 'Another ship is in collision course.',
    Premise3 = 'Ship is uncontrollable.',

    fact(Premise1),
    fact(Premise2),
    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    Conclusion = 'Ship will collide with another ship.',

    nl,
    write(Conclusion), write(' This is because of the following premises: '),
    nl,
    write('      1. '), write(Premise1), nl,
    write('      2. '), write(Premise2), nl,
    write('      3. '), write(Premise3).

logic(Conclusion, Premise1, Premise2, Premise3):-
    Premise1 = 'Ship has speed.',
    Premise2 = 'Engine shutdown.',
    Premise3 = 'Bow thruster shutdown.',

    fact(Premise1),

    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    Conclusion = 'Ship is uncontrollable.',

    nl,
    write(Conclusion), write(' This is because of the following premises: '),
    nl,
    write('      1. '), write(Premise1), nl,
    write('      2. '), write(Premise2), nl,
    write('      3. '), write(Premise3).

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Faulty regulator.',
    Conclusion = 'Engine shutdown.',

    fact(Premise1),
    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

```

```

logic(Conclusion, Premise1, Premise2, _):-
    Premise1 = 'Shaft generators shutdown.',
    Premise2 = 'Auxiliary generators shutdown.',

    Conclusion = 'Bow thruster shutdown.',

    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl,
    write('      2. '), write(Premise2), nl.

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Commanded to shutdown auxiliary generators.',
    Conclusion = 'Auxiliary generators shutdown.',

    fact(Premise1),

    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

logic(Conclusion, Premise1, _, _):-
    Premise1 = 'Engine shutdown.',
    Conclusion = 'Shaft generators shutdown.',

    fact('Commanded to shutdown auxiliary generators.'),
    fact('Faulty regulator.'),

    nl,
    write(Conclusion), write(' This is because of the following premise: '),
    nl,
    write('      1. '), write(Premise1), nl.

how:-
    logic(C, P1, P2, P3).

```

## Appendix E: Example 3 - Agent Based Perception-Action Technique

```
% =====
% Example 3 - Agent Based Perception-Action Technique
% -----
% Zobair Ibn Awal, Doctoral Student (2nd Year), Hasegawa Laboratory
% Department of Naval Architecture & Ocean Engineering (NAOE)
% Division of Global Architecture, Graduate School of Engineering
% Osaka University, Suita Campus, Osaka, Japan
% =====
% Started   : 13th January 2015
% Completed : 28th January 2015
% =====
% Agents: Ship, Captain, SOOW and JOOW.
% === Execution =====
% This segment runs the total model.
% -----
run:-
    reset,                % Reset necessary input variables
    bufferCreate,          % Creates buffer files for ship maneuvering
    repeat,                % If any of the following predicates fail then the
                        % search will repeat from this point
    iterationNo(Now),      % Counts the iteration number
    nl, nl,                % Inserts two new lines for output screen
    count(Now, Next),      % Counts the next iteration number
    write('Iteration No   : '),
    write(Now),

    helloCaptain(CaptainPerception, % Captain queries other
                  CaptainAction),    % crew members actions from which he
                        % perceives and/or takes action

    helloSOOW(SoowPerception, % SOOW queries other crew
               SoowAction),     % members actions from which he
                        % perceives and/or takes action

    helloJOOW(JoowPerception, % JOOW queries other crew
               JoowAction),    % members actions from which he
                        % perceives and/or takes action

    shipState(JoowAction),      % As JOOW is at the Helm, his commands
                        % will change ship state

    helloShip,                  % Simualtes ship's behaviour based on K-T Model

    result(Now, % Print the results in
            CaptainPerception, CaptainAction, % 'result.txt' file
            SoowPerception, SoowAction,
```

```

        JoowPerception, JoowAction),

    Next > 500,                                % Condition for 500 iterations
    !.

% --- Resetting the main loop -----
reset:-
    tell('0. Counter.txt'),
    write('1'),
    write('.'),
    told,

    tell('result.txt'),
    write(''),
    told,

    tell('output.txt'),
    write(''),
    told,

    tell('shipState.txt'),
    write(''),
    write('Ship is dead stop'),
    write(''),
    write('.'),
    told,

    tell('1. captainPerception.txt'),
    write(''),
    write('Need to make a sail past'),
    write(''),
    write('.'),
    told,

    tell('3. joowPerception.txt'),
    write(''),
    write('No Action'),
    write(''),
    write('.'),
    told,

    tell('2. soowPerception.txt'),
    write(''),
    write('No Action'),
    write(''),
    write('.'),
    told,

    tell('1. captainAction.txt'),
    write(''),
    write('No Action'),
    write(''),
    write('.'),
    told,

    tell('2. soowAction.txt'),

```

```

write(''),
write('No Action'),
write(''),
write('.'),
told,

tell('3. joowAction.txt'),
write(''),
write('No Action'),
write(''),
write('.'),
told.

% --- Printing the results -----
result(IterationNo,
      CaptainPerception, CaptainAction,
      SoowPerception, SoowAction,
      JoowPerception, JoowAction):-
append('result.txt'),
write(IterationNo),
write('\t'),
write(CaptainPerception),
write('\t'),
write(CaptainAction),
write('\t'),
write(SoowPerception),
write('\t'),
write(SoowAction),
write('\t'),
write(JoowPerception),
write('\t'),
write(JoowAction),
nl,
told.

% --- Iteration counting -----
iterationNo(Now):-
see('0. Counter.txt'),
read(Now),
seen.

count(Now, Next):-
Next is Now + 1,
tell('0. Counter.txt'),
write(Next), write('.'),
told.

% == Captain's Perception and Actions =====
% This segment deduces the actions of the Captain based on his perceptions,
% the actions of other crews and facts. The followings result Captain's
% perception: 1) Captain's vision, 2) His own actions, 3) SOOW's actions,
% 4) JOOW's actions, 5) His own perceptions. Based on the derived perception
% the Captain takes an action.
% -----

helloCaptain(Perception, Action):-

```

```

see('shipState.txt'),
read(ShipState),
seen,
ShipState = 'Ship sailing at full speed',
see('shipLatLong.txt'),
read(ShipX),
read(ShipY),
seen,
rockPosition(RockX, RockY),
ShipY = RockY,
Risk is RockX - ShipX,
Risk =< 1000,
Risk > 0,

Perception = 'Danger ahead',
tell('CaptainVision.txt'),
write(''), write(Perception), write(''), write('.'),
told,

Action = 'Command J00W 10 degree starboard',
tell('1. captainAction.txt'),
write(''), write(Action), write(''), write('.'),
told.

helloCaptain(Perception, _):-
    see('1. captainAction.txt'),
    read(Fact),
    seen,

    captainPerception(Fact, Perception),
    tell('1. captainPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloCaptain(Perception, _):-
    see('2. soowAction.txt'),
    read(Fact),
    seen,

    captainPerception(Fact, Perception),
    tell('1. captainPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloCaptain(Perception, _):-
    see('3. joowAction.txt'),
    read(Fact),
    seen,

    captainPerception(Fact, Perception),
    tell('1. captainPerception.txt'),
    write(''), write(Perception), write(''), write('.'),

```



```

    told,

    !.

helloCaptain(Perception1, Perception2):-
    see('1. captainPerception.txt'),
    read(Perception1),
    seen,

    captainPerception(Perception1, Perception2),

    tell('1. captainPerception.txt'),
    write(''), write(Perception2), write(''), write('.'),
    told,

    !.

helloCaptain(Perception, Action):-
    see('1. captainPerception.txt'),
    read(Perception),
    seen,

    captainAction(Perception, _, Action),
    tell('1. captainAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    tell('1. captainPerception.txt'),
    write(''), write(Action), write(' done'), write(''), write('.'),
    told,

    !.

helloCaptain(_, Action):-
    Action = 'No Action',
    tell('1. captainAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    !.

% === SOOW's Perception and Actions =====
% This segment deduces the actions of the SOOW based on his perceptions, the
% actions of other crews and facts. The followings result SOOW's perception:
% 1) Captain's actions, 2) His own actions, 3) JOOW's actions, 4) His own
% perceptions. Based on the derived perception the SOOW takes an action.
% -----

helloSOOW(Perception, _):-
    see('1. captainAction.txt'),
    read(Fact),
    seen,

    soowPerception(Fact, Perception),
    tell('2. soowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),

```

```

told,

!.

helloSOOW(Perception, _):-
    see('2. soowAction.txt'),
    read(Fact),
    seen,

    soowPerception(Fact, Perception),
    tell('2. soowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloSOOW(Perception, _):-
    see('3. joowAction.txt'),
    read(Fact),
    seen,

    soowPerception(Fact, Perception),
    tell('2. soowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloSOOW(Perception1, Perception2):-
    see('2. soowPerception.txt'),
    read(Perception1),
    seen,

    soowPerception(Perception1, Perception2),

    tell('2. soowPerception.txt'),
    write(''), write(Perception2), write(''), write('.'),
    told,

    !.

helloSOOW(Perception, Action):-
    see('2. soowPerception.txt'),
    read(Perception),
    seen,

    soowAction(Perception, _, Action),
    tell('2. soowAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    tell('2. soowPerception.txt'),
    write(''), write(Action), write(' done'), write(''), write('.'),
    told,

    !.

```

```

helloSOOW(_, Action):-
    Action = 'No Action',
    tell('2. soowAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    !.

% == JOOW's Perception and Actions =====
% This segment deduces the actions of the JOOW based on his perceptions, the
% actions of other crews and facts. The followings result SOOW's perception:
% 1) Captain's actions, 2) SOOW's actions, 3) His own actions, 4) His own
% perceptions. Based on the derived perception the JOOW takes an action.
% -----

helloJOOW(Perception, _):-
    see('1. captainAction.txt'),
    read(Fact),
    seen,

    joowPerception(Fact, Perception),
    tell('3. joowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloJOOW(Perception, _):-
    see('2. soowAction.txt'),
    read(Fact),
    seen,

    joowPerception(Fact, Perception),
    tell('3. joowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloJOOW(Perception, _):-
    see('3. joowAction.txt'),
    read(Fact),
    seen,

    joowPerception(Fact, Perception),
    tell('3. joowPerception.txt'),
    write(''), write(Perception), write(''), write('.'),
    told,

    !.

helloJOOW(Perception1, Perception2):-
    see('3. joowPerception.txt'),
    read(Perception1),
    seen,

```

```

    joowPerception(Perception1, Perception2),

    tell('3. joowPerception.txt'),
    write(''), write(Perception2), write(''), write('.'),
    told,

    !.

helloJ00W(Perception, Action):-
    see('3. joowPerception.txt'),
    read(Perception),
    seen,

    joowAction(Perception, _, Action),
    tell('3. joowAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    tell('3. joowPerception.txt'),
    write(''), write(Action), write(' done'), write(''), write('.'),
    told,

    nl, write('J00W perceives: '), write(Perception),
    nl, write('So J00W acts : '), write(Action),
    !.

helloJ00W(_, Action):-
    Action = 'No Action',
    tell('3. joowAction.txt'),
    write(''), write(Action), write(''), write('.'),
    told,

    !.

% === Knowledge of Captain Agent =====
% In this segment the knowledge of the Captain is saved in terms of
% 'Perception - Action' predicates.
% -----

% --- Captain's Perceptions -----
captainPerception(Perception1, Perception2):-
    Perception1 = 'Conduct route planning on large scale chart',
    Perception2 = 'Ship is ready for voyage'.

captainPerception(Action, Perception):-
    Action = 'Lift anchor',
    Perception = 'Anchor lifted'.

% --- Captain's Actions -----
captainAction(Perception, _, Action):-
    Perception = 'Need to make a sail past',
    Action = 'Command S00W to change voyage plan for sail past'.

captainAction(Perception, _, Action):-
    Perception = 'Ship is ready for voyage',

```

```

    Action = 'Command JOOW to lift anchor'.

captainAction(Perception, _, Action):-
    Perception = 'Anchor lifted',
    Action = 'Command JOOW - Full Ahead'.

% === Knowledge of SOOW Agent =====
% In this segment the knowledge of the SOOW is saved in terms of
% 'Perception - Action' predicates.
% -----

% --- SOOW's Perceptions -----
soowPerception(Commanded, Perception):-
    Commanded = 'Command SOOW to change voyage plan for sail past',
    Perception = 'Need to change voyage plan for sail past'.

soowPerception(Fact, Perception):-
    Fact = 'Need to change voyage plan for sail past',
    Perception = 'Need to conduct route planning'.

% --- SOOW's Actions -----
soowAction(Perception, _, Action):-
    Perception = 'Need to conduct route planning',
    Action = 'Conduct route planning on large scale chart'.

% === Knowledge of JOOW Agent =====
% In this segment the knowledge of the JOOW is saved in terms of
% 'Perception - Action' predicates.
% -----

% --- JOOW's Perceptions -----
joowPerception(Fact, Perception):-
    Fact = 'Command JOOW to lift anchor',
    Perception = 'Need to lift anchor'.

joowPerception(Fact, Perception):-
    Fact = 'Command JOOW - Full Ahead',
    Perception = 'Need to execute command - Full Ahead'.

joowPerception(Command, Perception):-
    Command = 'Command JOOW 10 degree starboard',
    Perception = 'Need to execute 10 degree starboard'.

% --- JOOW's Actions -----
joowAction(Perception, _, Action):-
    Perception = 'Need to lift anchor',
    Action = 'Lift anchor'.

joowAction(Perception, _, Action):-
    Perception = 'Need to execute command - Full Ahead',
    Action = 'Execute command - Full Ahead'.

joowAction(Perception, _, Action):-
    Perception = 'Need to execute 10 degree starboard',
    Action = 'Execute 10 degree starboard',
    execute10DegreeStarboard.

```

```

% --- Environment -----
rockPosition(X,Y):-
    X = 2500,
    Y = 0.

% --- Ship Agent -----
shipState(JoowAction):-
    JoowAction = 'Execute command - Full Ahead',
    ShipState = 'Ship sailing at full speed',

    tell('shipState.txt'),
    write(''), write(ShipState), write(''), write('.'),
    told,
    !.

shipState(JoowAction):-
    JoowAction = 'Execute command - Dead Stop',
    ShipState = 'Ship is dead stop',

    tell('shipState.txt'),
    write(''), write(ShipState), write(''), write('.'),
    told,
    !.

shipState(_):- !.

helloShip:-
    see('shipState.txt'),
    read(ShipState),
    seen,

    ShipState = 'Ship sailing at full speed',
    simulateOneStep,
    !.

helloShip:-
    see('shipState.txt'),
    read(ShipState),
    seen,

    ShipState = 'Ship is dead stop',
    !.

bufferCreate:-
    readInput(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime),
    saveBuffer(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime).

readInput(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime):-
    see('input.txt'),
    read(StartTime),
    read(X),
    read(Y),
    read(Psi),
    read(Drift),

```

```

read(Speed),
read(Rudder),
read(Ri),
read(K),
read(T),
read(H),
read(SimTime),
seen.

saveBuffer(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime):-
    tell('buffer.txt'),
    write(StartTime), writeln('.'),
    write(X), writeln('.'),
    write(Y), writeln('.'),
    write(Psi), writeln('.'),
    write(Drift), writeln('.'),
    write(Speed), writeln('.'),
    write(Rudder), writeln('.'),
    write(Ri), writeln('.'),
    write(K), writeln('.'),
    write(T), writeln('.'),
    write(H), writeln('.'),
    write(SimTime), writeln('.'),
    told.

simulateOneStep:-
    append('output.txt'),
    readBuffer(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime),
    calculate(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime,
        NextTime,Xi,Yi,Psi,Drift,Speed,Rudder,Rii,K,T,H,SimTime),
    saveBuffer(NextTime,Xi,Yi,Psi,Drift,Speed,Rudder,Rii,K,T,H,SimTime),
    nl,
    write(NextTime),
    XiRound is round(Xi),
    YiRound is round(Yi),
    PsiRound is round(Psi),
    write(' - '), write(XiRound),
    write(' - '), write(YiRound),
    write(' - '), write(PsiRound),
    %shipStatus(NextTime, XiRound, YiRound),
    told,
    tell('shipLatLong.txt'),
    write(XiRound), write('.'), nl,
    write(YiRound), write('.'),
    told,
    !.

readBuffer(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime):-
    see('buffer.txt'),
    read(StartTime),
    read(X),
    read(Y),
    read(Psi),
    read(Drift),
    read(Speed),
    read(Rudder),

```

```

read(Ri),
read(K),
read(T),
read(H),
read(SimTime),
seen.

calculate(StartTime,X,Y,Psi,Drift,Speed,Rudder,Ri,K,T,H,SimTime,
NextTime,Xi,Yi,Psii,Drift,Speed,Rudder,Rii,K,T,H,SimTime):-

K1 is 1/T*K*Rudder - 1/T*Ri,
K2 is 1/T*K*Rudder - 1/T*Ri - 1/T*1/2*H*K1,
K3 is 1/T*K*Rudder - 1/T*Ri - 1/T*1/2*H*K2,
K4 is 1/T*K*Rudder - 1/T*Ri - 1/T*H*K3,

Rii is Ri + H/6*K1 + H/3*K2 + H/3*K3 + H/6*K4,
Psii is Psi + Rii*H,

Xi is X + Speed*cos(pi/180*(Psii-Drift)),
Yi is Y + Speed*sin(pi/180*(Psii-Drift)),

NextTime is StartTime + H.

execute10DegreeStarboard:-
readBuffer(StartTime,X,Y,Psi,Drift,Speed,_,Ri,K,T,H,SimTime),
RudderNew is 10,
saveBuffer(StartTime,X,Y,Psi,Drift,Speed,RudderNew,Ri,K,T,H,SimTime).

```



## Appendix F: List of Publications

List of publication/presentations published/accepted during this research work

No.	Description	Type
1	Awal, Z.I. and Hasegawa, K. (2015). Analysis of Marine Accidents Due to Engine Failure – Application of Logic Programming Technique (LPT), <i>Technical Information at the Journal of the Japan Institute of Marine Engineering (JIME)</i> , Vol. 50, No., 6, pp. 39 – 46.	Journal/Transactions
2	Awal, Z.I. and Hasegawa, K., (In Press). A New Approach to Accident Analysis: Multiple Agent Perception-Action, <i>Accepted for publication in the Transactions of the Society of Naval Architects and Marine Engineer (SNAME)</i> .	
3	Hasegawa, K. and Awal, Z.I. (2013). A Concept for Expert System Based Accident Prediction Technique for Ship Maneuvering, <i>Proceedings of the 5<sup>th</sup> International Conference on Design for Safety (IDFS)</i> , 25-27 November 2013, Shanghai, China, In USB Drive.	Conference Proceedings
4	Awal, Z.I. and Hasegawa, K. (2014). Analysis of Marine Accidents by Logic Programming Technique, <i>Proceedings of the 10th International Symposium on Marine Engineering (ISME)</i> , 15-19 September 2014, Harbin, China, In USB Drive, Paper No. ISME127.	
5	Awal, Z.I. and Hasegawa, K. (2014). Application of Logic Programming Technique on Maritime Accident Analysis, <i>Proceedings of the International Conference on Ship and Offshore Technology (ICSOT 2014)</i> , 4-5 November 2014, Makassar, Indonesia, pp. 59-66.	
6	Awal, Z.I. and Hasegawa, K. (2015). Accident analysis by logic programming technique, <i>Safety and Reliability of Complex Engineered Systems: ESREL 2015</i> , pp. 13-21.	
7	Awal, Z.I. and Hasegawa, K., (2015). A new approach to accident analysis: Multiple agent perception-action, <i>Proceedings of the 5th World Maritime Technology Conference (WMTC)</i> , 3-7 November 2015, Rhode Island, USA, In CD-ROM.	
8	Awal, Z.I. and Hasegawa, K., A Study on Accident Theories and Their Application in Costa Concordia Accident, <i>Proceedings of the 16<sup>th</sup> Academic Exchange Seminar between Osaka University and Shanghai Jiao Tong University</i> , pp. 33, 22-24 October 2013, Osaka, Japan.	Seminar/Workshop
9	Awal, Z.I. and Hasegawa, K., Addressing Accident Problems Using Logic Programming Technique (LPT), <i>Presented at the Joint Workshop between Seoul National University (SNU) and Osaka University (OU)</i> , Osaka University, 31 <sup>st</sup> August 2015.	