The Assurance of Bayesian Networks for Mission Critical Systems

Mark Douthwaite

Doctor of Philosophy

University of York

Computer Science

September 2018

Dedication

To Becky and to my parents, for their unending support.

Abstract

A prerequisite for the assurance of any mission-critical system is a comprehensive understanding of a system's properties and behaviours. This is a challenging proposition for many AI-based Systems (AISs). Their functionality is often dictated by factors that are often outside the scope of the assurance concerns typical of conventional software systems. These distinctions have implications for all phases of the design, development, deployment and operation of AISs. They pose serious problems for existing software assurance standards, guidelines and techniques: the application of existing practices to an AIS will fail to expose or mitigate numerous system aspects that can contribute to hazardous system behaviours.

This thesis introduces a number of techniques that aim to support the resolution of these problems for Bayesian Network-based Systems (BNSs). This class of system has been deployed in many applications, ranging from medical diagnostic systems to navigational controls aboard autonomous systems. To date, there is no published literature on the deployment of these systems in directly safety-critical roles. This thesis introduces approaches aimed at addressing three particular challenges. Firstly, it proposes a framework for conceptualising and communicating the distinctions between BNSs and conventional software systems and uses this framework to generate and refine a set of BNS verification and validation objectives. Secondly, it introduces an assurance-focussed BNS analysis technique that can provide targeted information on mission-critical aspects of a BNS. Finally, it outlines an approach for describing how BNS-specific safety evidence relates to BNS aspects, and how the evidence can be used to derive sufficient confidence in a mission-critical BNS.

These contributions are then evaluated in the context of a case study that indicates the utility of the proposed techniques, and how these can be used to comprehensively structure and target the unconventional assurance concerns associated with the development of a mission-critical BNS.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank Tim Kelly, my supervisor over the course of this research project. His mentoring and advice have been invaluable, and I have learnt a great deal in my time working with him. He has been an exceptional supervisor.

I would also like to thank the rest of the HISE team for the warm welcome and interesting discussions over the course of my time with the Department of Computer Science. I'd particularly like to thank Katrina Attwood for all the great conversations (and the proof reading and input on this thesis too!). I'd also like to thank Rob Alexander and Jane Fenn for their support and input over the course of this research project.

This work has been performed in collaboration with BAE SYSTEMS and EPSRC as part of an Industrial CASE studentship. I would therefore like to thank all those at BAE that have provided feedback and discussion over the last few years.

Over my time at York I met a lot of wonderful people, many of whom directly or indirectly shaped this work, and I'd like to thank them all. However, there have been a handful who have been there through all the ups and downs of PhD life, particularly Matt Dale, Joe Branson and Thoryn Haylett, and I'd like to thank them for their friendship and support. I'd also like to thank Alex Roberts for helping me keep perspective over the years despite the distance.

Finally, I'd like to thank my family, my parents and grandparents who have supported and encouraged me throughout my academic life. Without them none of this would have been possible. Last but far from least, I'd like to thank Becky for her support, care and encouragement – not to mention patience – over the course of the PhD. This thesis exists because of her.

Declaration

I declare that the research described in this thesis is original work, which I undertook at the University of York during 2014 - 2018. Except where stated, all of the work contained within this thesis represents the original contribution of the author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Some parts of this thesis have been published in conference proceedings; where items were published jointly with collaborators, the author of this thesis is responsible for the material presented here. For each published item the primary author is the first listed author.

- Mark Douthwaite, Tim Kelly, Safety-Critical Software and Safety-Critical Artificial Intelligence: Integrating New Practices and New Safety Concerns, *Safety Critical Systems Symposium (SSS) '18, SCSC*

- Mark Douthwaite, Tim Kelly, Establishing Verification and Validation Objectives for Safety-Critical Bayesian Networks, *International Symposium on Software Reliability Engineering (ISSRE) '17 - Special Session on AI (Workshop on Software Certification), IEEE*

Thus to have a retentive memory, and to proceed 'by the book,' are points commonly regarded as the sum total of good playing. But it is in matters beyond the limits of mere rule that the skill of the analyst is evinced. He makes, in silence, a host of observations and inferences. So, perhaps, do his companions; and the difference in the extent of the information obtained, lies not so much in the validity of the inference as in the quality of the observation. The necessary knowledge is of what to observe.

Edgar Allan Poe, 1841

The actual science of logic is conversant at present only with things either certain, impossible or entirely doubtful ... Therefore, the true logic of this world is the calculus of probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind.

James Clerk Maxwell, 1850

The power to become habituated to his surroundings is a marked characteristic of mankind ... We assume some of the most peculiar and temporary of our late advantages as natural, permanent and to be depended on, and we lay our plans accordingly.

John Maynard Keynes, 1919

Chapter 1

Introduction

1.1 Bayesian Networks

Bayesian Networks (BNs) have been used in a variety of state-of-the-art Artificial Intelligence (AI) applications over the course of their existence.[1] One of the earliest success stories in the history of AI was the development of so-called Expert Systems (ESs) throughout the 1980's. These systems were initially hailed as a major breakthrough in the field, with significant commercial and research funding focussing on the development of increasingly advanced ESs that could model the reasoning processes of human experts [1, 2].

However, after a number of early breakthroughs, it quickly became clear that the development of human-like ESs demanded the introduction of techniques that provided a more nuanced reflection of the world. For example, a doctor's diagnosis is rarely the *only* possible explanation for a patient's symptoms: it is, however, generally the most *probable* explanation given the available information and the doctor's expert judgement. If this doctor were to receive additional information, they may *revise* their *belief* in the diagnosis - either positively or negatively - and take action if required. Such forms of reasoning were not generally feasible in early ESs [3].

The difficulties of these 'traditional' ESs in handling complex, uncertain scenarios prompted the development of a number of belief modelling approaches. The BN framework would ultimately emerge from this research. As a modelling approach, BNs provide a flexible approach that both explicitly captures the uncertainty inherent in many domains,

[1] Bayesian Networks have been known by many names, including: Bayesian Belief Networks (BBNs), Belief Networks, Causal Networks, and Probabilistic Belief Networks (PBNs). Commonly used variants include Naive Bayes Classifiers, Kalman Filters, and Gaussian Mixture Models.

while also dynamically revising the current belief of the model in the state of the world in the presence of ever-shifting information and observations.[2]

As the framework has matured, it has integrated a number of techniques for performing inference over highly complex BNs, thereby supporting the utilisation of BNs in increasingly complex applications. It has also introduced several new representational concepts, each supported by an array of tailored structure and parameter learning techniques. The modern BN modelling framework is part of a larger Probabilistic Graphical Modelling (PGM) framework. This more general framework has deep connections to many other widely-used AI and Machine Learning (ML) approaches. This includes Markov Networks (MNs), Hidden Markov Models (HMMs) and Kalman Filters (KFs), Naive Bayes Networks and Gaussian Mixture Models (GMMs) to name a few [2].

The ability of BNs to capture highly complex, nuanced domains and to provide robust, effective reasoning in the presence of significant uncertainty has made them popular in a number of academic and industrial fields. In particular, the approach has experienced enduring popularity in the fields of medicine, aerospace, robotics, natural language processing and computer vision [4, 5, 6, 7]. However, at the time of writing, BNs have not been utilised to any significant extent in the core functionality of safety-critical systems - they have been used almost exclusively as either decision support tools, or in other highly constrained applications where they demonstrate well-defined technical behaviours [8].

There is therefore no existing guidance on the direct utilisation of a BN-based System (BNS) in mission- and safety-critical contexts. However, the properties of BNS make them an attractive prospect for the fulfilment of a number of safety-critical roles aboard autonomous systems. In particular, they offer powerful capabilities of use in various mission management and system monitoring roles. In such roles, a BNS will likely constitute a significant proportion of the control logic of the autonomous system: BN-based software components will directly support mission planning (in autonomous vehicles this may include aspects of route planning, the system's behavioural profile, system availability and endurance etc.), as well as other advanced fault diagnostic and prognostic capabilities. Fur-

[2]The term 'domain', or alternatively 'target domain' is used here – and throughout this thesis – to refer to the knowledge, system or process a BN model may be designed to capture. For example, the domain of a BN model developed for use in medical applications may be the diagnostic knowledge of medical practitioners, the dynamics of a hospital ward, or the current state of a patient. In the case of the latter, the BN may represent a mathematical model of a patient's vital signs or other key statistics.

thermore, BNs may be used to provide a coherent framework for the integration of many other AI modelling approaches, including - but not limited to - Artificial Neural Networks (ANNs), Random Forests (RFs), and Support Vector Machines (SVMs) [9, 10, 11]. Some current state-of-the-art AI-based Systems use combinations of AI approaches in precisely this way [12, 13].[3]

1.2 Motivation

The research conducted for this thesis was motivated by the proposed utilisation of a Prognostic Health Monitoring (PHM) system in an on-line decision-making role that supports other mission planning capabilities of an Unmanned Aerial System (UAS). In this role, the outputs of the BN-based PHM would be used to directly inform the functional behaviour of the UAS platform. In this context, such a PHM system would constitute a mission-critical application of a BNS.

This application raises several assurance challenges for which there is no existing safety-focussed guidance. Current safety-critical software used in the aviation domain is often developed in compliance with well-established software safety standards and practices such as ARP4754, ARP4761 and DO-178C. These standards are used across the civil and military aerospace domains and are generally well regarded by system safety practitioners. For the purposes of this thesis, the following definitions for the terms 'safety-critical' and 'mission-critical' systems are adopted [14]:

Mission-critical - A mission-critical system is one where a hazard can degrade or prevent the successful completion of an intended operation. Causing property damage certainly fits this description; exposing or damaging a company's reputation would align with the definition as well.

Safety-critical - A safety-critical system is a system that poses a physical hazard to human life. If a failure occurs and exposes a hazard, it might cause physical harm to users, patients, practitioners, doctors, nurses, bystanders, and even people in the proximity of an accident.

Currently, however, there is limited research into the application (and relevance) of

[3] An AI-based System is defined as any system that relies on any number of AI techniques for the provision of core system functionality.

these standards and practices to the development and assurance of modern AISs generally – and specifically in the case of BNSs. Moreover, there is a general lack of research in this area for many other widely-used system safety and software safety standards and guidelines. In particular, there is an absence of clarity on potential weaknesses in existing guidance for this class of system, and very limited guidance on how to approach and mitigate these weaknesses. The core motivations for the research presented in this thesis can therefore be summarised as follows:

- To understand and articulate the general weaknesses of existing assurance practices in the context of BNSs and, by extension, to identify the assurance challenges associated with their development.
- To establish a comprehensive conceptual framework for the description and assurance of mission-critical BNSs that can be used to address the identified assurance challenges.
- To develop analysis techniques that mitigate the weaknesses of existing assurance practices and explicitly address the identified assurance concerns associated with this class of system.

1.3 Thesis Hypothesis

With these considerations in mind, the hypothesis of this thesis has been defined as follows:

> *It is possible to provide targeted assurance for Bayesian Network-based Systems through the analysis and evaluation of underlying probabilistic models, data artefacts and computational techniques that have the potential to affect confidence in the safety of a system.*

The following terms are defined for clarity:

Probabilistic Models - At the core of any BNS is one or more BN models. These models are *declarative representations* of a target domain. This is a purely abstract mathematical model that can be considered independent of any given software implementation and of any algorithms applied to it. Consequently, the properties and behaviours associated with this model are intrinsic to it [2, 3].

Data Artefacts - Modern BNSs typically rely on some form of learning to define a given BN model. A BN model is said to be trained on a data set. This data set may be composed of one or more data artefacts. These artefacts may range from quantitative, empirically derived component reliability metrics, to relatively unstructured, qualitative data in the form of maintenance logs or expert statements.

Computational Techniques - The ability to process large quantities of data and to extract information from BN models is facilitated through the development and utilisation of learning and inference algorithms. Learning algorithms enable the construction of BN models from data, thereby generating a model that to some degree reflects the properties of that data, and the limitations of the learning algorithm. Inference algorithms support the efficient querying of BN models, enabling BNs to be used to support complex reasoning and decision making tasks.

This hypothesis is principally concerned with four considerations. First, that assurance efforts must explicitly target those aspects of a BNS that make this class of system distinct from conventional software-intensive systems. A prerequisite for targeting these aspects is a clear and comprehensive enumeration of the 'unconventional' properties of BNSs, and a rigorous understanding of how these properties may affect the safety of such a system. This implies that common assumptions about the development, implementation and operation of software-intensive systems may need to be revised in the context of BNSs.

Next, the behaviour of a BNS is heavily influenced by the properties and behaviours of the BN models that may be utilised in the provision of safety-critical functionality. As indicated above, the properties and behaviours of a BN model is intrinsic to it. This is analogous to the manner in which the predictions of Newton's Laws of Motion are intrinsic to the mathematical model they define: the conclusions are the same irrespective of how they are computed (e.g. their software implementation). Similarly, errors in these predictions are also intrinsic to the law itself. For example, Newton's Laws begin to break down when a physical body begins to move very quickly or is in a strong gravitational field. This occurs due to the omission of a number of crucial terms in Newton's original formulation of these Laws. Likewise, a failure to account for (or accurately represent) certain mathematical behaviours within a BN model will produce errors that are intrinsic to it. This has important implications for the effectiveness of current assurance techniques and software analysis approaches in relation to their ability to identify certain forms of hazardous behaviour.

Thirdly, the data-driven nature of modern BNSs may introduce several atypical assurance considerations. Unlike many conventional software systems, data-driven BNSs are often more directly exposed to the operational environment they are to be deployed into - potentially both during development and operation. The data used to train a BNS inevitably influences what a BNS can 'know' about the world around it; the data acts as the raw material from which a BN model is generated. In this sense, aspects of a BNS's control logic may be considered to be 'open' to the domain being modelled.

Finally, the hypothesis aims to capture the need to address the role of novel computational techniques in the development and operation of this class of system, with particular reference to the implications of the use of learning and inference algorithms upon the behaviours of BNSs. In particular, both the use of learning algorithms in the development of a BN and the utilisation of adaptive learning algorithms in an operational context may introduce important assurance considerations. These three aspects were identified based on an initial review of the primary concerns of BN developers and the limited available guidance on AISs more generally. In essence, large sections of a BNS's control logic are likely to be automatically generated according to a confluence of behaviours defined by each of these three system aspects. Indeed, some within the field of AI have suggested that these types of systems may constitute a conceptually distinct form of software. As Karpathy notes in the context of Deep Learning (DL) [15]:

> *"Software 2.0 is written in [model parameters]. No human is involved in writing this code ... we specify some constraints on the behavior of a desirable program ... and use the computational resources at our disposal to search the program space for a program that satisfies the constraints."*

The intention of this hypothesis is therefore to understand the implications of these system aspects in the context of BNSs and ensure that they are adequately targeted by assurance practitioners.

1.4 Thesis Structure

The thesis is structured as follows:

Chapter 2 - This chapter introduces the key concepts underpinning the BN framework, reviews the development lifecycles of BNSs and outlines a number of existing use-cases

of BN models in both academic and commercial applications. An overview of current approaches to software and system safety is then provided, followed by a critique of the applicability of these techniques in the context of BNSs. Finally, a review of existing work on the assurance of AISs is provided, and a number of recurrent themes are identified and discussed.

Chapter 3 - The focus of this chapter is on the definition of a general framework for the comprehensive description and modelling of a BNS, and the use of this framework in the generation of a set of verification and validation objectives for BNSs. The chapter aims to provide a conceptual overview of assurance considerations for BNSs, to provide a mechanism for shared understanding of a BNS between BN developers and assurance practitioners, and to introduce a flexible set of objectives to support the assurance of a mission-critical BNS.

Chapter 4 - This chapter introduces a technique for establishing the criticality of all BN models and BN model components in a BNS. The intention of this technique is to introduce an approach that supports the safety-focussed analysis of BN models, and to partially address the challenges of maintaining traceability of the behaviours of a BNS, at the point in which a BN model becomes the principal determinant of a BNS's functionality.

Chapter 5 - The aim of this chapter is to outline an approach to establishing the sufficiency of assurance activities in the context of BNSs. An evidence-classification approach is provided to support this goal. This approach extends existing assurance techniques to BNSs and integrates closely with the RM-BNS framework defined in Chapter 3.

Chapter 6 - This chapter provides an evaluation of the concepts, frameworks and techniques introduced in this thesis in the context of the thesis hypothesis. The evaluation also addresses the practical applicability of this work to 'real-world' use cases. This is addressed partly using a case study to demonstrate the application of the ideas proposed in this thesis to an example BNS. The contributions of the thesis are discussed and evaluated in the context of the broader field.

Chapter 7 - Finally, this chapter provides concluding remarks on the contributions of the thesis. It provides an overview of potential future research directions that could be taken on the basis of the work presented in the thesis, as well as potential research gaps

identified during the course of research.

Chapter 2

Literature Review

2.1 Introduction

While there is extensive published research on the development of Bayesian Network-based Systems (BNSs) and on the development and assurance of conventional software systems, there is a near-complete absence of existing research into the assurance of mission-critical BNSs. However, there is a small but relatively established body of research into the development of Artificial Intelligence-based Systems (AISs) for safety-critical applications. Furthermore, many of these applications utilise Artificial Intelligence (AI) approaches that overlap conceptually with aspects of BNSs and can therefore be utilised to provide an insight into potential assurance concerns for BNSs.

With these considerations in mind, this chapter first reviews published work on the use of the BN representational framework within the field of AI, and then outlines key BN concepts upon which much of this thesis will be based. This review is then extended to explore BNS-specific development practices. Following this, a review of existing practices in the development of safety-critical systems is provided. This focusses on existing assurance techniques and concepts, typical assurance lifecycles and the application of existing safety standards to conventional software systems. Finally, a survey of relevant research into the development of safety-critical AISs is introduced. This focusses on those AISs that share conceptual or contextual similarities with BNSs or with the motivating example introduced in Chapter 1. Where appropriate, observations and experiences in this research that are relevant to BNSs will be discussed and criticised.

2.2 Bayesian Networks

The term 'Bayesian Networks' and the modern formalism of the BN framework was introduced by Pearl in the late 1980s [16,17]. This work was motivated by the need to develop a computational model that reflected the ability of humans to integrate data taken from multiple sources and to reason effectively over this information. Pearl recognised the need to accommodate the subjective, uncertain and incomplete state of knowledge of the world that frequently characterises human decision making [17,18]. This early research identified the failure of many then-current AI techniques in accurately replicating and capturing the uncertainties and reasoning processes of human experts as an inherent limitation of the approaches then available to AI developers [3]. The work of Pearl and others ultimately contributed to the formation of a field within AI research often referred to as 'Probabilistic AI'. This sub-field develops approaches to AI that utilise the laws of probability and the tools of statistics and mathematics to support the development of 'artificially intelligent' systems. The BN framework forms one part of this expansive and highly active area of research.

2.2.1 Uncertainty and Probabilistic Artificial Intelligence

Uncertainty is near-ubiquitous in the real-world: many aspects of the world appear to exhibit non-deterministic properties and the ability of observers to measure these properties is often constrained by the noise inherent in their measurements, or of the limitations of tools available to them. This form of uncertainty is often referred to as statistical uncertainty, and can be defined as:

Statistical Uncertainty – The uncertainty that arises from limitations in the precision of measurement devices, or in statistical fluctuations arising from random background processes. For example, uncertainty of this kind may arise when measuring the temperature of angular resolution of a star in the night sky: each independent measurement may vary due to random fluctuations as light passes through the Earth's atmosphere. Importantly, this uncertainty can – in principle – be mitigated through the use of alternative equipment or more measurements [19, 20].[1]

This form of uncertainty characterises many aspects of a range of scientific and en-

[1] Statistical uncertainty is also referred to as aleatoric or ontological uncertainty.

gineering disciplines. However, there are other forms of uncertainty. A second form, commonly defined alongside statistical uncertainty, is the uncertainty arising as a consequence of the restrictions on the observability of the world: only a subset of aspects of the world are observed at any given point. This form of uncertainty is frequently referred to as systemic uncertainty, and can be defined as:

Systemic Uncertainty – The uncertainty arising from limitations in the ability of observers to access or capture information about the world. For example, this may occur as a consequence of neglecting certain variables within a mathematical model (such as in the case of the Schiaparelli lander, where an aerodynamic model utilised during development did not account for some atmospheric effects arising from its supersonic entry into Mars' atmosphere), or because of unknown biases in data [20]. [2]

These forms of uncertainty play an important role in the decision making of human experts. For example, medical practitioners frequently work with partially complete data – they can observe only a subset of a patient's physical attributes at any one time, and do not have access to a truly exhaustive history of their patient's health. Furthermore, there are very few disease-symptom mappings that are entirely deterministic: not all patients with a disease will manifest an identical set of symptoms, nor may the causes of a disease be the same in all patients. In this case, both forms of uncertainty are therefore present. In practice, medical practitioners focus instead on reducing the state-space of a problem and on establishing what is *most probable* as opposed to what is certain.

Attempting to model many aspects of the world using approaches that do not – or cannot – capture this uncertainty can produce systems that are severely limited in their capacity to reason about the world around them. Such systems may utilise models that mask the uncertainty and complexity of the environment they operate in. This can lead directly to an inferior quality of reasoning and decision making by a given system and may ultimately restrict the utility of these systems [21].

Indeed, it was experience of precisely these limitations in early rule-based AI approaches that prompted the development of many of the modern approaches to AI [3, 22]. These experiences motivated the development of a number of schools of thought within the AI domain, each proposing various philosophical, theoretical or practical concepts

[2] This form of uncertainty is also referred to as epistemic or systematic uncertainty.

and approaches for the development and deployment of AISs that provide mechanisms for overcoming the challenges inherent in earlier approaches.

The sub-field of 'Probabilistic AI' utilises approaches that are built upon probability theory and utilise probabilistic methods for representation, learning and reasoning. Probabilistic methods naturally support the representation of a domain in terms of states that are *probable* or *improbable* as opposed to those that are certain or impossible: they are designed to enable the integration of multiple observations alongside additional uncertain items of information in order to provide an assessment of the system's belief in the state of the world [2, 22].

The development and adoption of probabilistic approaches has met with broad success in a number of domains. Many state-of-the-art AI applications are either based directly upon principles developed within the probabilistic AI domain, or can be interpreted within a probabilistic framework [23]. These approaches can avoid the (often intractable) problem of enumerating all possible states within a domain, and the associated developmental and computational challenges that may accompany the development of such large state-spaces. Within the field of statistics and artificial intelligence, the state-space of a model is defined as:

State-space – The set of all possible states of a system; each state of the system corresponds to a unique point in the state-space. The state or the measurement can be either continuous or discrete. In this context, the term originates in the field of control engineering [24].

A probabilistic framework can be used to capture and reason over the most probable states in a given model. This can produce models that maintain a degree of interpretability. Consequently, probabilistic approaches can sometimes provide more faithful (or more nuanced) representations of the state of a given domain – and thereby facilitate the development of a more effective AIS [2, 23, 25].

2.2.2 Probabilistic Graphical Models

There are many approaches to utilising probabilistic approaches in AISs. One of the most common approaches is to represent (and reason over) domains using Probabilistic Graphical Models (PGMs). In general, the complexity and nature of many domains is such that 'traditional' probabilistic or statistical methods become impractical or intractable for com-

plex problems [1]. In particular, the state-space that must be modelled may be extremely large and may prohibit the effective use of alternative methods [3, 26]. PGMs build upon concepts from discrete mathematics and graph theory in order to provide a representational approach that can support the development of systems that can manage this complexity: they facilitate a compact graphical representation of high-dimensional probability distributions that could not be practically captured using traditional approaches [2, 3].

Models developed within the PGM framework can be categorised as belonging to one of three distinct groups of models: undirected models, mixed models and directed models. The first group, undirected models, are commonly referred to as Markov Networks (MNs). Examples of this group of graphical models include Markov Chains (a one-dimensional MN) and Markov Random Fields (a two-dimensional MN). These modelling approaches have been used extensively in many AI applications, including decision making, process modelling, computer vision, natural language processing, as well as in some state-of-the-art reinforcement learning techniques.[3] They can provide a flexible, intuitive approach to representing complex relationships within a domain, and for building up associative relationships between observed features in a target domain. This latter property has made them particularly valuable within the computer vision domain as an efficient means of providing basic computer vision functionality, such as image segmentation and image synthesis [29, 30, 31].

The second group of models, mixed models, are those models that contain both directed and undirected edges; they are generalisations of directed and undirected probabilistic models [32].[4] These models can be used to represent more complex relationships between aspects of domains and have been used to boost the interpretability of models [33]. This group of PGM models is generally the least widely used (and discussed) of the three groups. The final class of PGMs are the family of probabilistic Directed Acyclic Graphs (DAGs). The most prominent of these is the widely used Bayesian Networks (BNs) framework.

2.2.3 The Bayesian Network Framework

Early research on BNs was motivated by the recognition of the need to develop computational models of human inferential reasoning that could support the integration of

[3] A discussion of reinforcement learning is beyond the scope of this chapter. A good introduction to the topic can be found in [27] and here [28].

[4] These models are also referred to in some literature as Chain Graph Models [33, 34].

information from multiple sources and of varying types into a single, coherent description of the world [22]. A primary motivator for this research was the need to develop a system that supported 'bi-directional reasoning': the ability to integrate top-down and bottom-up reasoning within a model.[5] The resulting probabilistic modelling framework became known as Bayesian Networks.[6]

The development of the BN framework in the late 1980's led to the emergence of BNs as a state-of-the-art solution to many applications previously dominated by rule-based AI approaches. In current research, BNs continue to provide state-of-the-art performance in many AI applications, including in their 'traditional' role in probabilistic expert systems, as well as in advanced medical and fault diagnosis, data fusion, drug discovery, robot navigation and natural language processing applications [35, 36, 37, 38, 39, 40]. The BN framework represents a generalised version of many other commonly used machine learning techniques. For example, Hidden Markov Models (HMMs) and Kalman Filters (KFs) fall into this category [2, 34].[7] A further example of a specific version of a BN – and perhaps the most widely used BN variant – are those used in many document classification and 'spam' filters: the so-called Naïve Bayes models [41].

Bayesian Networks derive their name from the mathematical basis upon which the framework is based: they facilitate the generalised application of Bayes' Theorem to arbitrarily large problems. Equation 2.1 shows this theorem. This theorem provides a formal basis for revising the probability (belief) of an event in the presence of new evidence (i.e. the current state of the world). Bayes' Theorem consists of four components. The first $P(A|B)$ is a conditional distribution capturing the probability of an event A given an event B.[8] This component represents the *posterior distribution*: the degree of belief in the occurrence of event A given all available information about event B. The component $P(A)$ then represents the *prior marginal* distribution: the degree of belief in the occurrence of event

[5] This refers to the ability to reason based upon knowledge of the world and observations (or perceptions) respectively.

[6] Bayesian Networks are referred to using a number of different names. Somewhat archaic names include Probabilistic Belief Nets and Bayesian Belief Networks [2, 3]. Other names include Causal Networks (in specific contexts).

[7] Note that in this context the 'Markov' in the term Markov Models refers to the dynamics of the process being modelled, not the modelling framework itself.

[8] For example, the probability of a coin coming up heads *given* the coin is known to be loaded. The expectation of the outcome of a coin flip would vary based on whether or not it is known that the coin is loaded: Bayes' Theorem explicitly captures this problem.

A in the absence of information on the status of event B. This reflects the background rate of occurrence of an event independent of any additional information. Similarly, $P(B)$ represents the marginal probability of event B. Finally, $P(B|A)$ represents the conditional probability of the occurrence of event B given knowledge of the state of event A. This is sometimes referred to as the *likelihood* component of Bayes' Theorem [2].

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.1}$$

Formally, events A and B are described as being Random Variables (RVs).[9] In statistics, RVs can be described as being a variable in a model whose value is the product of a stochastic process. RVs are described as being either continuous or discrete. For example, a discrete RV may be used to model the probability of a coin toss, or the probability of a student receiving a particular grade. In these cases, there are a finite number of possible outcomes. For example, for an RV used to model a coin toss, the RV may be modelled as using a Bernoulli distribution with two outcomes: heads or tails.[10] In contrast, a continuous RV may be used to represent a velocity of a robot, or the heart rate of a hospital patient. In either case, the RV may be modelled using any appropriate continuous probability distribution.

A RV also has a more general definition: it describes a mapping that associates each outcome with the probability of the occurrence of that outcome: RVs within a BN model may represent *any* mathematical function that captures a valid probabilistic mapping of states. In general, a RV can be defined as:

Random Variable – A function that assigns numerical values to the outcomes of a random phenomenon [42].

A RV is parameterised. In the most basic cases, this parameterisation may correspond to discrete probabilities: a coin-flip RV may be defined by a single parameter (i.e. 0.5).

[9]The term 'Random Variable' and 'Variable' are often used interchangeably in the context of statistics and PGMs.

[10]Probability distributions are a central concept to BNSs, and arguably the Bernoulli distribution is the most important single distribution to understand in this context. However, an in-depth technical review of the statistical concepts including this distribution is beyond the scope of this chapter. An excellent introduction to statistical concepts can be found in Barlow's textbook, and Kendall's series on advanced statistics [19].

These mappings are sometimes referred to as the *local structures* in a BN model. The most widely used local structure for RVs is the table representation: these are commonly referred to as Conditional Probability Tables (CPTs). Local structures are also referred to as Conditional Probability Distributions (CPDs): this makes no assumptions about the structure of the RV and is a generalisation of a 'CPT'.

	Coin = Heads	**Coin = Tails**
State = Loaded	0.7	0.3
State = Fair	0.5	0.5

(a) The joint conditional distribution for $P(Coin|State)$.

Coin = Heads	**Coin = Tails**
0.6	0.4

(b) The marginal distribution $P(Coin|State = Unknown)$.

Coin = Heads	**Coin = Tails**
0.5	0.5

(c) The marginal distribution $P(Coin|State = Fair)$.

Figure 2.1: A comparison of a table-structured Conditional Probability Distributions (CPDs) before and after marginalisation.

Figure 2.1a shows an example table-structured CPD (CPT) for an experiment in which a coin may – or may not – be biased, and the associated *conditional* probabilities of the outcomes (i.e. whether the coin turns up heads or tails) given the state of the coin. As will be discussed later, the mapping of variables within this table and the parameters that express relationships between them are commonly *learned* from data. The parameters in this example refer to the probability of the occurrence of a given state. Intuitively, if the coin-flipper knows the coin is biased, then the probability of the coin turning up heads is known (in this case it would be 0.7). However, a key feature of this approach

can be seen when there is uncertainty in whether or not the coin is biased. By computing the *marginal* distribution for this RV (i.e. the distribution with respect only to whether the coin is heads or tails) we can express a belief in the coin turning up heads *given* the nature of the coin is unknown. The marginal CPT is shown in Figure 2.1b. If the nature of the coin becomes known, or additional data becomes available, this distribution can be *revised*. This is shown in Figure 2.1c. In this case, the marginal distribution is revised to reflect a fair coin.

Table-structured CPDs are the most common local structure in BN models. However, they are not the only local structures that are used in BNs. As the RV definition suggests, a number of alternative local structures can be used. These include several Artificial Neural Network (ANN), Random Forest (RF) and Support Vector Machine (SVM) variants, as well as other BN models.[11] This enables the development of hierarchical and object-based BN (OOBN) models for high-complexity domains [4, 43]. It is possible to 'embed' some ANNs, SVMs or indeed any other approach that meets the RV definition within a single BN model [9, 10, 11, 44].

Taken together, these properties make the BN framework an ideal *mathematical* tool for integrating information from many distinct AI approaches into a single coherent model. This model is a representation of a domain that can be used to effectively reason over all available information – and may be revised when new information becomes available. Bayesian Networks therefore offer a representational framework that directly model the world, as Pearl notes [18]:

> *Perhaps the most important aspect of Bayesian Networks is that they are direct representations of the world, not the reasoning process.*

As will be discussed later, the implications of this fact strongly influence the interpretation of individual BN models and the development process for complex BNSs.

2.2.3.1 Representation

As a framework for capturing uncertainty in the 'real world', BN models share many conceptual similarities with mathematical models developed in the natural sciences – particularly in the domain of physics. A fundamental aim of a BN model is to provide a

[11] The local structure of a variable within a BN may itself be a complete BN model in its own right. See [2] for a review of this form of BN model.

compact representation of complex problems in a computationally (and developmentally) tractable mathematical construct. Formally, they are used to represent high-dimensional Joint Probability Distributions (JPDs) in a *factorised* format. Practically, this means that the BN framework allows developers to mathematically *decompose* highly complex probabilistic models into (theoretically) less complex mathematical components, and to use this factorised representation to perform efficient computations over the model. Without this factorised representation, the state-space of complex probabilistic models can become too large for computations utilising the model to remain tractable.

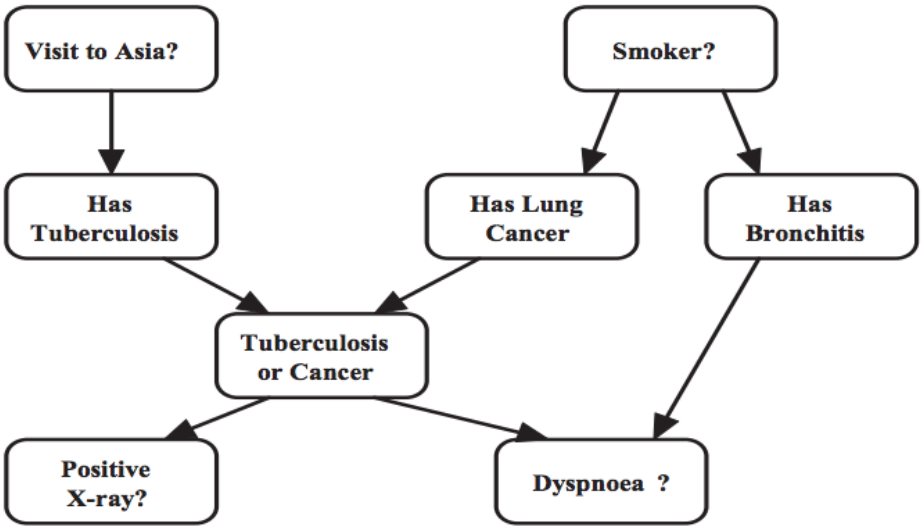


Figure 2.2: A visualisation of the structure of the 'Asia Model'.

For example, consider the model shown in Figure 2.2. This figure shows a simple (fictitious) medical diagnostic model for diagnosing respiratory diseases in patients returning from a trip to Asia. This model was introduced by Lauritzen and Spiegelhalter in their early work on BNs and remains a common example within the field of Probabilistic AI and is typically referred to as the 'Asia Model'. In this case, the model's JPD encodes the joint distribution over the eight discrete RVs within the model. Each of these variables is represented with a standard table-structured CPD, and each is defined as having only two states (i.e. as a binary discrete RV). Each node within the BN represents the CPD for that variable conditioned on its parents (i.e. ancestor nodes). For example, the Dyspnoea variable is entirely defined by the CPD shown in Figure 2.3 – only the conditional dependencies of the Dyspnoea variable and its parent variables (Either Disease and Bronchitis)

are encoded in the local structure. The Asia Model represents one of the simplest possible BN architectures.

	D = Y	**D = N**
B = Y, E = Y	0.9	0.1
B = N, E = Y	0.7	0.3
B = Y, E = N	0.8	0.2
B = N, E = N	0.1	0.9

Figure 2.3: The tabular representation of the $P(Dyspnoea|Bronchitis, Either)$ CPD, where $D = Dyspnoea$, $B = Bronchitis$, and $E = Either$.

Despite this simplicity, the Asia Model highlights an important property of the BN framework. The full JPD of this model contains 256 individual parameters. By defining the model according to only local structures and therefore parameterising the CPDs individually, the decomposed representation facilitated by the BN framework can be captured using less than 36 individual parameters. This reduces the modelling challenge associated with developing a probabilistic model of this diagnostic problem. From a practical perspective, this compartmentalises the development of the model: each variable within the BN can – in principle – be modelled with consideration *only* of its parent variables. In contrast, specifying the full JPD using alternative methods may require the consideration of each variable in the context of each other variable within the model – thereby creating a combinatorial explosion in the complexity of the model's state-space.[12]

[12]From a software perspective, the exponential growth in the size of a model's state-space given the introduction of each new variables can quickly produce models in which an explicit (i.e. unfactorized) representation of the model's JPD may exhaust all computational resources that could possibly be dedicated to it.

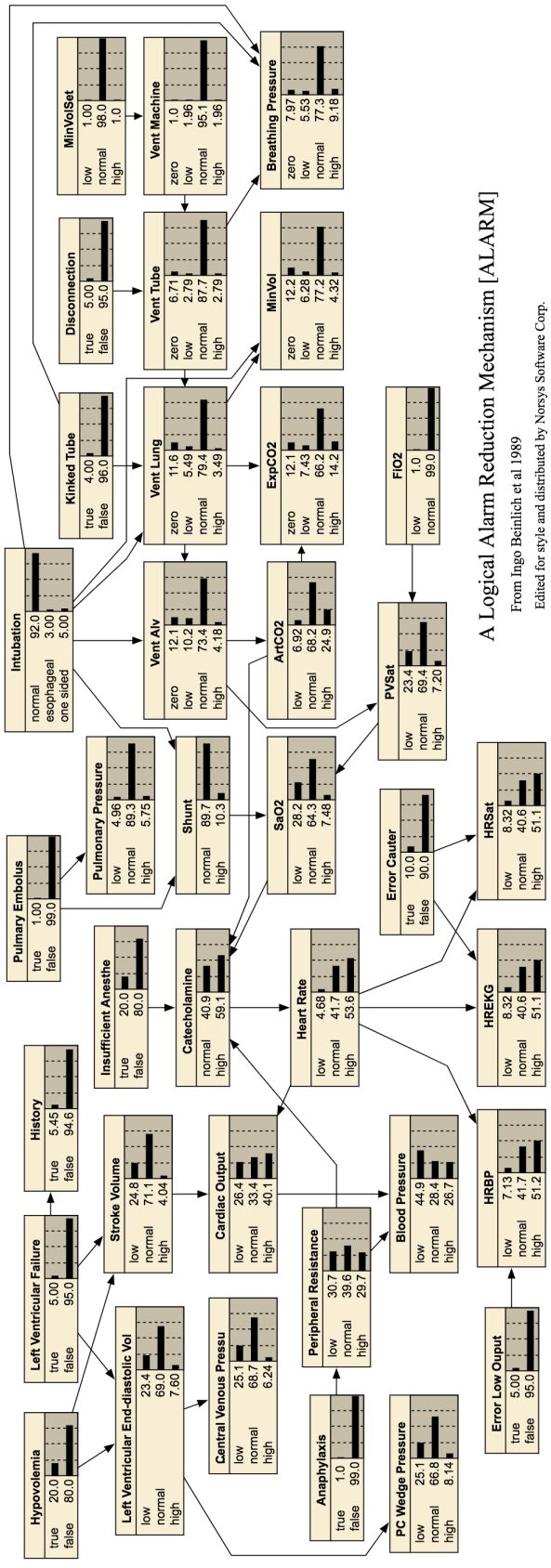


Figure 2.4: A visualisation of Beinlich's 'ICU Alarm' model [45, 46].

An example of this problem can be seen in the model shown in Figure 2.4. This model is also commonly used within the BN domain and is often referred to as Beinlich's *Alarm ICU* model. It was originally developed to reduce false positive alarms to medical practitioners on hospital Intensive Care Unit (ICU) wards. It represents the diagnostic knowledge of medical practitioners and has been integrated with the health statistics of many patients and the results of a number of medical studies. It provides a small-to-moderately sized BN model that is commonly used to benchmark analysis and inference techniques and will be returned to later in the thesis. Technically, it remains relatively simple.

However, it highlights the problem of combinatorial explosion in BN models: an explicit representation of this model's JPD would be composed of more than 1×10^{500} *unique* states. This is a large number. For scale, there are estimated to be only 1×10^{80} atoms in the universe. The BN framework reduces the number of parameters that must be specified to less than 2×10^4 independent parameters.[13] This makes the development of larger BN models technically challenging but generally practicable, especially when automated learning algorithms are utilised. Clearly, however, an exhaustive exploration of such models will be infeasible.

The *Alarm* model also reinforces Pearl's assertion about the nature of BN models as direct, *declarative* representations of the world: any conclusions derived from a BN model are *intrinsic to the model* and (in principle) reflect some state of the world according to the model. By extension, errors in this representation are intrinsic to the model too. In the case of the models shown in Figures 2.4 and 2.2, they represent expert knowledge of disease diagnosis and the health of patients. In other cases, they may represent the relationships between sensor readings and vehicle positions, the type of a document given the occurrence of a sequence of words, or the probability that a component in an aircraft may fail given its current operational context. Consequently, the properties (and errors) of these models are independent of any other underlying implementation details – whether these implementation details be related to hardware or software [1, 2].

This property has resulted in the majority of BN literature broadly neglecting any specific implementations or tools in favour of treating BN models as abstract mathematical constructs; research focusses on understanding *what* has been built (or learned) and the

[13] This can be further reduced by exploiting independence properties within the network. However, the number of independent parameters that must be specified (or learned) remains in the hundreds.

dynamics of these models. This is consistent with the concerns of BN developers: a single software implementation may support many models. In many cases, the models will be the most valuable artefacts in a given system.

2.2.3.2 Inference

The ability of BNs to factorise complex probabilistic spaces into more easily tackled mathematical problems is a key property of the framework. However, as outlined in section 2.2.3, the utility of a BN model lies in the ability to infer the state of the world according to that model and any available information about the world. This requires updating and manipulating the model's JPD. As discussed in section 2.2.3.1, an explicit representation of the model's JPD is typically prohibitively large for the purposes of tractable computation.

The aim of BN inference algorithms is therefore to update and manipulate the model's JPD without computing the joint distribution. In the BN literature, this process is typically referred to as *querying* the model.[14] This involves instantiating a model with any available information, and then computing the model's *degree of belief* in the state of the variables in the model. The inference algorithms that facilitate these operations fall into two categories: exact inference algorithms and approximate inference algorithms.

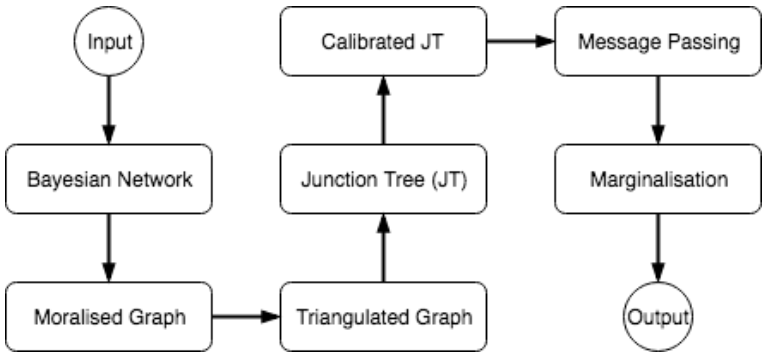


Figure 2.5: A high-level visualisation of the Junction Tree algorithm.

In general, algorithms in the former category are favoured whenever they can be feasibly used: as their name suggests they provide the exact degree of belief in an RV's state given all available information. There are several approaches to exact inference for BNs. The most ubiquitous approach relies on 'belief propagation' (BP) within a BN model. Figure 2.5 shows the general process of a BP inference algorithm. As the figure indicates,

[14]This language appears to have somewhat fallen out of favour as the use-cases for BNs have grown and the field has matured.

the algorithm begins by performing a series of transformations (*Moralisation* and *Triangulation*) over the graph structure of a BN. This produces an undirected and further compacted representation of the original model. However, the semantic value of the original model is often eroded or lost in the resulting structure. This structure is often referred to as a Junction Tree (JT).[15] The JTs produced from this process are then 'calibrated': the probabilistic information from the originating BN is used to configure the JT's structure. This ensures that the JT encodes an equivalent *probabilistic* representation to the original model – even if some of the semantic value of the originating representation is lost.

At this stage, the JT is ready for belief propagation. Belief propagation is typically performed using some variant of a message-passing algorithm. These algorithms ensure the probabilistic integrity of the JT is maintained (i.e. that the distribution encoded by the JT remains a valid probability distribution). When this message passing is complete, the distributions encoded within the JT are exactly equivalent to the state of belief in the originating BN *given* all available information. The marginal distributions for each variable within the JT can then be computed and represented within a BN model.[16] An in-depth review of belief propagation algorithms for BNs is beyond the scope of this chapter. However, detailed explorations of this area can be found in the work of Lauritzen, Koller, or in Pearl's original publications [2, 16, 17, 32].

Whilst exact inference engines are generally desirable, they cannot always be employed in BNSs. This is often because of the specific properties of the structure encoded by the BN model, or because the complexity of the model undermines the ability of exact inference approaches to function effectively. Circumstances such as this are common in applications focussed on image processing, natural language processing and some robot localisation and navigation tasks [47, 48, 49]. These applications are characterised by models that can rapidly become extremely large and complex – even for relatively simple tasks. In these cases, approximate inference methods are utilised. There are several approximate inference algorithms available to BN developers. Amongst the most prominent of these

[15]The state-space of a JT is typically defined such that the combinatorial growth typical of other representations is to some degree mitigated. However there are many open research problems relating to the optimal construction of JTs and most rely on heuristics or brute-force search methods.

[16]Note that the *Input* and *Output* nodes inf Figure 2.5 correspond to model inputs (sometimes referred to as evidence) and the marginal posterior distributions respectively.

are approaches that utilise Markov Chain Monte Carlo (MCMC) sampling to estimate a BN model's marginal distributions.[17]

Inference techniques utilising MCMC methods are based on a simple concept: that by generating a sequence of samples drawn from the underlying probability distribution encoded in the BN model, an estimate of the 'true' posterior distribution of the BN can be approximated. This approach is mathematically guaranteed to converge to the 'true' posterior of a BN model given a sufficient number of samples. However, sampling the a BN's distribution proceeds at random, and while these algorithms are guaranteed to converge to an accurate estimate, *when* they will converge can sometimes be difficult to estimate and therefore guarantee [2, 51, 52]. Furthermore, these approaches are typically sensitive to their 'initial conditions' – the initial configuration of the algorithm and the first random samples drawn from the BN model. This introduces additional uncertainty into the estimate of the posterior that is obtained. A number of mitigation strategies have been developed to tackle these problems, but ultimately, they remain an inherent consideration for this class of algorithm.

2.2.3.3 Learning

Beyond the representational and inferential capabilities of the BN framework, there is the final aspect of BNs that has driven the popularity and interest in these approaches – and other techniques in the field of AI generally. This is the ability to define and develop a BN model such that it can *learn* from data rather than being 'manually' programmed. Indeed, many AI approaches have been developed and designed to accomplish tasks that traditional, explicit, and therefore *transparent* programming methods are poor at (or incapable of) performing. These tasks are often too complex, variable or novel for conventional software development approaches to effectively tackle [53, 54].

In the case of BNs, the semi-transparent mathematical constructs that drive them may constitute a significant proportion of a system's programming. The learning algorithms that identify and define these models can be considered to (partially) take the place of a software system's programmer. The BN developer's role is then shifted to a parallel but distinct set of considerations focussed on identifying and configuring a learning (i.e. programming) method that will achieve desired performance characteristics when exe-

[17] Another widely used approximate inference approach is Variational Inference, an introduction to this can be found in the work of Blei *et al* and Jordan [50, 51].

cuted; this is as opposed to designing the software architecture and ultimately manually programming the system.[18]

From this perspective, the properties of learning algorithms and the *hyperparameters* used to configure the algorithms that are used in the development of a BNS are of central importance to the functional behaviours of the completed system. This is true even if these algorithms are not physically encoded in the final system or otherwise deployed aboard the completed system. In the context of learning algorithms, a hyperparameter can be defined as follows:

Hyperparameter – Parameters that are set *prior* to the *model* learning process. For example, these parameters can tune the rate of learning of an algorithm or the algorithm's tolerance to noise in any training data. This is in contrast to *parameters*: these are derived through the learning process [51]. There may be additional optimisation/learning phases focussed on identifying an optimal set of hyperparameters [55].

For BNSs, there are two principal aspects of a model that learning algorithms aim to optimise: a model's structure and/or a model's parameters. In BNSs, the semantic value of a model's structure has driven the development of a wide variety of structure learning algorithms for knowledge discovery; these algorithms can be used to automatically discover relationships between variables within a domain. This has been used extensively in a number of medical knowledge discovery studies [56,57]. The structure learning problem in the case of BN models is often more important in than in alternative AI approaches [1,2].

For example, learning algorithms for ANNs and SVMs are often only minimally concerned with directly learning the structure of a model – structure learning is instead dominated by more 'coarse' structure aspects such as the higher-level architecture of an ANN, or the size and variant of an SVM. In many cases these models have limited inherent semantic value. The problem of learning models in these cases is instead generally more focussed on parameter learning algorithms to identify an effective weighting configuration for the model.[19] For BN models, structure learning algorithms can identify model structures that imply conditional relationships that may in turn indicate causal relationships

[18]This is of course still an important phase of the development of a BNS, but its importance is diminished for many systems utilising modern AI approaches.

[19]There is an equivalence in parameter learning and structure learning in ANNs: a zero-weighted connection in an ANN is precisely equivalent to learning part of the model's structure.

between domain aspects [22]. This structure will define the reasoning behaviours of a BN model and therefore directly influence the behaviour of a BNS utilising this model – it will therefore also directly influence the interpretability of the model.

The most widely-used class of learning algorithm for BNSs are those that focus on parameter learning. Algorithms of this class aim to parameterise the local structures of a BN model. They often do not indicate the conditional or causal dependencies of variables in a model and therefore do not influence the interpretability or reasoning performance of a model in the manner of structure learning algorithms. Instead, parameter learning algorithms generally modify the *degree of dependence* of variables upon one another. Expressing the quantified strength of dependence of variables within a domain is often problematic for human experts, and the ability to achieve this with an automated process has proved invaluable in a number of practical contexts [58, 59, 60]. For example, the state-space of BN models is commonly so large that eliciting parameters from human experts is impracticable. The combination of the limitations in the ability of experts to quantify dependencies in a domain and the size of the state-space of a model can make the assessment of the accuracy of individual learned (and expert-elicited) parameters a challenging problem. The development of techniques that explore these learned properties is an ongoing area of research [56, 61].

While learning algorithms for BNs can be generally divided into those focussed on either *structure* or *parameter* learning, they can also be used in support of '*off-line*' and '*on-line*' learning. These cases can be defined as follows:

Off-line Learning – A learning/optimisation activity that occurs *prior* to the deployment of the system. AISs using this form of learning are sometimes referred to as 'pre-trained' or 'static' models.

On-line Learning – A learning/optimisation activity that may begin prior to the deployment of the system, but continues *after* the system is operationally deployed. AISs using this form of learning are sometimes referred to as 'adaptive' or 'active' models.

In the former case, a BN model would be a static artefact that does not vary after a BNS is deployed. This learning approach is often appropriate for diagnostic roles or other decision support roles where – to a good approximation – the domain being modelled can be captured using only off-line training techniques (i.e. the domain will not vary significantly over time). For example, a diagnostic problem may be well characterised by expert

experience and extensive empirical trials: the problem may be sufficiently constrained so as to ensure the continued validity of a static model.

However, in other cases, the target domain may be inherently dynamic and therefore constantly evolving. In these cases, a pre-trained BN model may become obsolete during deployment. In such cases, adaptive learning approaches may therefore be required to maintain the performance of the BNS. Adaptive methods typically allow a model's structure and/or parameters to be updated in the presence of new data [62, 63]. This can allow a BNS to compensate for environmental and operational changes that may not have been present during the development of the system. When effectively implemented, these approaches can theoretically improve the robustness of a system to problems associated with 'distribution shift' and shifting operational contexts.[20]

Any errors or distortions in the sensory capabilities of a system utilising a BNS – or indeed in any input to the BNS – will be used by an adaptive learning algorithm to revise the state of the world as captured by the BN models driving the systems. From an assurance perspective, this introduces the possibility of a safety and security vulnerability for this class of systems. This could be in the form of novel 'hacking' approaches that target BNSs deployed with adaptive learning capabilities (and other AISs using similar techniques). For example, an adaptive learning approach used aboard an Unmanned Aerial System (UAS) for mission planning could be manipulated by low-levels of systematic biases in inputs (i.e. through the vehicles sensory capabilities: radar, cameras, etc.). This may have the effect of producing unexpected effects – the UAS may become situationally blind or compromised (either in terms of 'physical' perception or in terms of its ability to reason about its operational capabilities). As these vulnerabilities would exploit the *learning* and *model* aspects of these systems, a platform could be compromised *without access* to its hardware or software: the exposed 'API' of the system would be the world around it.

To summarise, the learning algorithms used by a BNS play a central role in the functional behaviour of a BNS. In both the on-line and off-line cases, the effect of the precise configuration of a learning algorithm may introduce significant differences in the reasoning and performance of two BN models trained on otherwise identical data. Indeed, the manner of their use – in conjunction with the nature of the BN models they are used to

[20] Distribution shift refers to the slow, sometimes random, changes in the statistical profile of the processes being modelled by an AIS. Technically this can result in increased statistical error in a model's predictions. Practically, this could result in an AIS manifesting apparently erratic or erroneous behaviours [21].

define – is considered by some to be a new form of software. This approach has been referred to within the AI community as differentiable programming, as Yann LeCunn, a pioneer in the field of ANNs, states:

> *"... people are now building a new kind of software by assembling networks of parameterized functional blocks and by training them from examples using some form of ... optimization ... It's really very much like a regular program, except it's parameterized, automatically differentiated, and trainable/optimizable [sic]."*

The emphasis on the optimisation of this class of system is critical. The learning algorithms optimise a software system's programming for a target problem. This optimisation is a key aspect of the properties and capabilities that make many AI approaches desirable, and indeed the only currently practicable solutions to many computational problems. However, these properties make them radically different from the conventional software systems that form the focus of existing software development lifecycles and software assurance literature.

2.2.4 Development Lifecycle

The development of a BNS is often dependent on information and system aspects that are not generally known *a priori*. For example, the quantity, breadth and type of data required for the development of a BNS may not be known. Similarly, the specific learning algorithm, its configuration and mathematical properties may be unknown at the inception of a project. These aspects may remain unknown or otherwise poorly defined until comparatively late in the development of a BNS [5, 6]. Aspects such as these are refined through iterative design, analysis and evaluation. The development lifecycle of many BNSs is therefore characterised by highly iterative design and testing phases, the duration and extent of which may not be possible to specify in advance. This is sometimes cited as the major source of technological risk in the development of AISs: it may not be possible to know whether a system will be performant until much later in the lifecycle of a BNS than is typical in conventional software systems [2, 6, 58, 64]. The difficulty in pre-emptively defining and decomposing a set of requirements alongside a general set of systematic development practices is well known within the AI domain [2, 64]. It is a major ongoing research challenge within the field. This is highlighted by Koller who states:

> *"... at the moment, the design process is more the result of trial-and-error experimentation, combined with some rough intuitions that practitioners learn by experience. It would be an important achievement to turn this process from a black art into a science."*

Indeed, the development of an effective BNS can share more conceptual similarities with the empirical experimental practices common to the natural sciences than to 'top-down' design processes more common to software engineering practices: whether a new experiment will succeed or fail at providing a particular observation cannot be known in advance. Moreover, precisely how an observation will be made may not be known in anything other than the most coarse-grained terms. Typically, experimentation proceeds from a high-level goal, and the decomposition of this goal into sub-goals (requirements) is commonly a highly iterative *process of discovery*. The decomposition process may reverse or change as new information becomes available to refine the experimental process. If a BN model is considered to be an empirical, probabilistic model of a target domain, this could be considered to be an unavoidable consequence of the framework – and is shared with other AI approaches.

While a generic framework for the development of a BNS has not yet been established, several authors have published work on the development of domain- and context-specific development practices which each aim to systematise or formalise some aspects of the iterative lifecycle of BNS [6,59,65]. One approach, proposed by Przytula *et al*, was developed specifically for constructing BN models for fault diagnosis aboard complex systems. It aims to provide a systematic process for the development of BN models. However, it is high level and still somewhat driven by BN developer intuition. The process defines four steps based on developing and deploying diagnostic BNs for complex aerospace systems:

1. **System Decomposition** The system is systematically decomposed on the basis of expert guidance into a set of subsystems. The boundaries of subsystems are then defined in accordance with the scope of knowledge of individual human experts (or teams of experts focussed on a given system aspect or subsystem). These subsystems are then decomposed further if the complexity of the subsystem is assessed to be too high. The principal aim of this stage of development is stated as being to identify individual BN models or model fragments, and to ensure the objective of these

models (i.e. to diagnose subsystem faults) is *minimally* complex.[21]

2. **Subsystem Definition** With the preliminary set of subsystems identified, the enumeration and definition of all faults and observations associated with each given subsystem then begins. Once again, the process aims to establish a minimal level of granularity for a BN model by defining faults in accordance with maintenance practices and knowledge of system experts. The process advocates defining faults according to replaceable subsystem components and standard maintenance activities. The example provided is as follows: if a group of circuit boards is replaced en masse, then this group of boards should be considered a single fault (as opposed to each individual board). At this point, the complexity of the subsystem should be re-evaluated. Further decomposition may be necessary in the event the complexity of the subsystem has grown beyond a heuristically derived value – though this cannot be established until *after* this phase of development. The process should then proceed iteratively until a minimal level of granularity of the subsystem model is agreed upon by relevant stakeholders.

3. **Subsystem Modelling** Next, the process advocates the development of BN models using the outputs from the previous phase (i.e. a list of faults and observations) by beginning from a minimal fault model of the subsystem. This initially takes the form of BN model with a single fault node: *all* faults within the subsystem are represented as being mutually independent and are all regarded to be direct causal factors in the occurrence of each observation. Therefore, at this stage, the subsystem is modelled as being equivalent to a naïve Bayesian Network. This is the simplest representation possible in the BN framework. If the model is not performant, an iterative process of refinement begins whereby the faults are gradually modelled as increasingly granular faults (i.e. iteratively broken into separate nodes with their own dependencies). This process continues until the model satisfies domain experts and may require regression to the previous phase of development for continued decomposition of the system or subsystem.

4. **Subsystem Model Integration** The final stage of the development of the models

[21]The emphasis of this development methodology on establishing 'minimally complex' models can be summarised by the quote attributed to Einstein that says; 'Everything should be made as simple as possible, but no simpler'. The aim here is to ensure the modelling activities produce parsimonious models.

is the integration of subsystem BN models into larger hierarchical models. These hierarchical models are iteratively tested over the course of these integration activities to ensure that the models remain performant when integrated with other models. If there are any shared dependencies across subsystems, the integration of these subsystem models may erode the performance of the models. The guidance suggests the parallel development and testing of such interdependent subsystems to avoid these issues where possible. However, it is noted that the integration process is highly application dependent. The guidance provided for the model integration phases is therefore comparatively scarce. It also indicates the potential need for further regression to previous development phases.

With little existing guidance into the development of large, complex BNSs for 'real world' applications, this research provides insight into the development of these systems in a practical context. However, as a piece of guidance, it applies only to a subset of BN applications: it makes a number of assumptions with respect to the BN models used – particularly with regard to the structure variants and modelling objectives of the BN models and subsystems. Consequently, the guidance does not provide (nor does it attempt to provide, given the research scope) a generic framework for developing BNSs. For the class of BNS it targets, it provides a set of heuristics for identifying when a development phase is complete. Finally, the research highlights a recurrent feature of existing BN development literature and methodologies: a near-complete absence of discussion of the role of the software and hardware implementation aspects – those aspects most of concern to software safety practitioners in conventional systems. The emphasis is once again placed on the BN models and the data (and expert input) used to develop the system. It again raises the issue of the effective communication of these unconventional system aspects to other domain experts and system stakeholders.

Another more general development methodology – referred to as the Knowledge Engineering for Bayesian Network (KEBN) methodology – has also been proposed by Pollino [59]. This approach does not assume any specific application but does assume that the BNS being developed is a knowledge-based decision support tool; this therefore narrows the scope of the guidance. The methodology was proposed to support the development of BN models for ecological modelling applications. As noted by the authors of the methodology, these applications are exposed to particularly high levels of both statistical and systemic uncertainty in their target domains. While the KEBN methodology is more gen-

erally applicable to a broader range of BN applications than the previous methodology, it still makes significant assumptions about the development process. In particular, the methodology does not provide guidance on the development of BNSs that utilise model structure learning approaches. These aspects are assumed to be derived through expert elicitation rather than an automated learning approach. Furthermore, as a knowledge-driven methodology, the approach does not provide insight into the development of purely data-driven BNSs – those systems that are developed with limited expert input. A flow chart illustrating the development methodology is shown in Figure 2.6.

As with the previous development methodology, the KEBN approach relies heavily on highly iterative development activities. As can be seen from Figure 2.6, the methodology is composed of three core processes: Structural Development and Evaluation (SDE), Parameter Estimation (PE), and Quantitative Evaluation (QE). The SDE process is not explored in detail within the research, though the authors indicate this is entirely elicited from experts. The PE processes are aimed at facilitating the integration of information from both domain experts and any available quantitative data into a single BN model. This process has an internal iterative cycle of evaluation and development aimed at addressing the need to continually review parameter estimates obtained from automated (parameter) learning approaches.

The QE processes in the methodology are once again highly iterative. They address an important feature of the development of BNSs: the quantitative (generally statistical) evaluation of BN models. As indicated by the authors' experiences in developing these systems, the development of a knowledge-based BNS must accommodate many iterations between the evaluation of a BN model, and the incremental revision of a BN model's parameters or structure. This process is driven by *model-focussed* analysis and evaluation techniques as opposed to conventional software analysis techniques. As with the previous methodology, there is no explicit discussion of the software implementation and evaluation aspects from a conventional software development perspective. Again, the operational experience of the authors stresses the disproportionate importance of evaluating the model, data and knowledge of domain experts when developing a BNS.

Ultimately, both of these methodologies and other items of existing work on defining structured approaches to BN development are *highly* limited. As discussed here, research and guidance that does exist is focussed on specific applications and domains and does not provide more generic guidance that could accommodate the development of a 'modern'

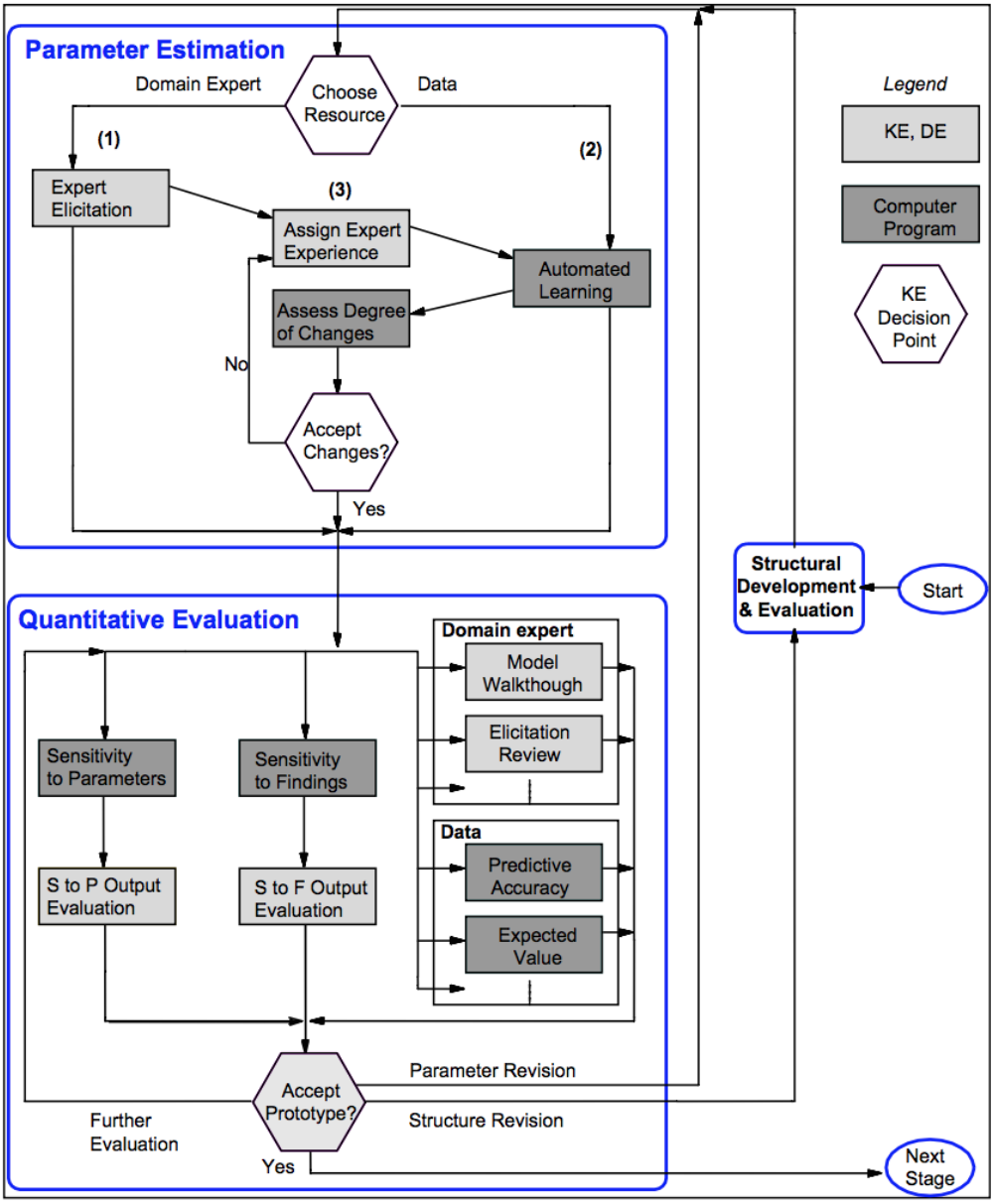


Figure 2.6: The Knowledge Engineering for Bayesian Networks (KEBN) development cycle, as proposed by Pollino *et al* [59].

data-driven BNS. The few guidance documents and development methodologies that do exist stress the necessity of accommodating highly iterative development cycles, the general constraints placed on the development of BNSs that arise from limitations in knowledge and resources that are particularly influential on the development, and finally the analysis and evaluation activities that will need to be performed by BN developers [5, 6, 59, 65, 66]. No existing methodologies or guidance for BNSs provide the rigour that would be necessary for the development of mission-critical BNSs. There is no existing guidance on how decisions on the *sufficiency* of BNS-specific analysis and evaluation techniques can be made with respect to the role of a BN model in the functional behaviour of a mission-critical BNS.

2.3 Safety Standards

The development and assurance of conventional safety-critical software systems is generally achieved through compliance with one or more safety standards. The safety-critical systems domain has produced a range of standards and guidance documents that each aim to tackle the design, implementation and deployment of this class of systems in various engineering domains. These standards espouse an assortment of development and analysis techniques that are believed to represent 'best practice' by engineers within the field a given standard is relevant to. This section gives a general overview of a subset of systems- and software-engineering safety standards and guidelines that are particularly pertinent to the motivating example of this thesis. These standards and guidelines represent a cross-section of key aerospace safety standards within the UK and provide insights into existing schools of thought within the safety-critical systems domain. Where applicable, general criticisms – and criticisms related to the applicability or relevance of aspects of these standards with respect to AIS/BNSs – are provided.

The Aerospace Recommended Practice (ARP) 4754 (Revision A) document is a widely used and generally well-regarded guidance document that addresses the development of civil aircraft and systems [67]. It has found extensive use internationally, and in particular is a *de facto* standard for aerospace systems developed within Europe and North America. As a set of guidelines, ARP 4754 is aimed at addressing the lifecycle of system-level functions, including system requirements, system verification and requirements validation. It is written to integrate closely with other guidelines focussed on addressing software and hardware aspects of an aircraft – aspects addressed by DO-178C and DO-254 respectively.

The guidelines are also designed to be used in conjunction with a further document: ARP 4761. This latter document defines the safety assessment methodologies to be used in the assurance of the aircraft. Taken together, these four documents are intended to provide comprehensive guidance on the development and certification of any aircraft or other airborne systems.

Practically, ARP 4754 provides a mechanism for assigning Development Assurance Levels (DALs) to system-level functions (in the context of this thesis: UAS-level functions). This includes the assignment of DALs to software systems deployed aboard an UAS through the application of the safety analysis methodologies outlined in ARP 4761 [68]. The software-specific development, analysis and evaluation of software systems and their associated lifecycles are handled by DO-178C [69]. For software systems, DO-178C provides sets of objectives alongside the recommended activities for satisfying these objectives. As with many other standards, the number of objectives and the analytical rigour required to meet them is typically proportional to some form of *criticality* of a given software system or subsystem to that system's safety: there are more objectives and an assumed higher level of rigour associated with the development of systems to higher DALs. By focussing on compliance with processes and pre-defined objectives, ARP 4754 and DO-178C have remained popular with industry groups and engineers thanks to their relatively concrete, well defined requirements. In the context of safety-critical systems, criticality can be defined as follows:

Criticality – A product of the *degree of contribution* of a system (or subsystem) to safety-related functional behaviours and the *severity of hazards* associated with these behaviours.

Though popular and widely used, this group of guidance documents has been criticised by safety practitioners. A frequent criticism originates in precisely the prescriptive nature of the documents that has contributed to their popularity. While they provide well-defined processes and requirements, the intent and rationale behind these processes and requirements can be lost: as Weaver states, development activities may instead focus on 'the letter of the law' as opposed to the intended 'spirit' of the guidelines [70]. From an assurance perspective this raises the possibility of the deployment of a system that has been developed in exact accordance with guidelines but may retain significant and potentially hazardous latent behaviours due to the narrowness of the verification and validation activities carried out [70, 71, 72, 73, 74].

A second related criticism arises from concerns over the completeness and relevance of the processes and requirements laid out in the guidelines [75, 76, 77]. This becomes particularly concerning in cases involving the development of new and novel systems that may integrate new development techniques, technologies or concepts [73, 78, 79]. For example, DO-178C is intended to capture best practices in software engineering for aerospace systems at the time of issue and consequently for software systems in use or in development at this time; there are assumptions implicit in the guidelines with respect to the nature and scope of these systems, and the relative invariance of these systems and development practices over time [79, 80].

One general criticism of this form revolves around the application of ARP 4754 to Autonomous Systems (AS) in the aerospace domain and the relation of these systems to their operational environment [81]. Many AS applications will adopt the reasoning and interactive behaviours of human operators. In these cases, they can become highly sensitive to environmental factors that are not typically within the scope of conventional systems. In its current form ARP 4754 provides little guidance on the role or importance of environmental modelling activities. To date, it has relied on assumptions that the relevant aspects of the operational environment of a system are well-defined, well-regulated and well-understood. As discussed in previous sections, the scope of *what* is relevant to an AIS may increase dramatically. This may be particularly true of BNSs that integrate and reason over a wide range of environmental aspects. Consequently, the current near-absence of guidance in this field will undermine the effectiveness of the guidelines in assuring this class of system.

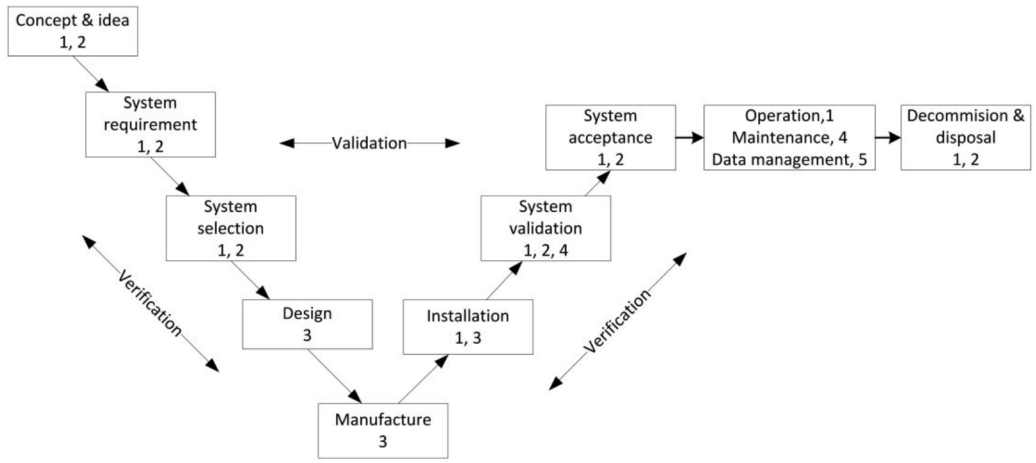


Figure 2.7: A visualisation of the 'V lifecycle' model [82].

These criticisms extend to the consideration of the generality of the structure of system lifecycles as assumed by both ARP 4754 and DO-178C. Specifically, both documents make strong assumptions about the development process of a system, the activities that can be carried out and the stages at which certain assurance information becomes available to developers. Figure 2.7 shows the so-called 'V lifecycle' commonly used to describe the development activities associated with a safety-critical software system [83]. Both ARP 4754 and DO-178C make assumptions that the development of a software system conforms closely to this general process. They assume that a conventional decomposition of requirements is possible at an early stage and that the development process proceeds in a roughly linear fashion. A number of authors have noted that for some classes of software system, this conceptual model of the lifecycle of a system may not be appropriate [81, 84, 85]. In the context of BNSs, 'traditional' software design and requirements decomposition techniques will become less relevant and insightful. This is because at some point, the functional behaviour of an AIS is often best described from a *model-centric* perspective as opposed to a software-centric one. This could be regarded as a developmental boundary for AISs – the point at which conventional function decomposition and analysis approaches can progress no further (or are otherwise uninformative) and the principal determinant of functional behaviour becomes the *models* that drive the system. This has been observed in other classes of AIS [64, 86].

These concerns are further compounded by the relatively slow pace of change in guidelines and standards in comparison to the pace of innovation in the broader technology and engineering space. In some cases, standards and guidelines have existed almost unchanged for decades despite changes to the technologies and applications they address [73, 80]. Moreover, even after modifications to guidance there is some debate as to the effectiveness of additional requirements and processes mandated by these modifications [87]. For example, it has been noted that in the case of earlier issues of DO-178C, the introduction of additional testing activities may have increased the testing burden of developers while producing no significant improvement in the effectiveness of this testing. It is assumed that the same suite of software testing techniques will be capable of exposing hazardous behaviours across all software systems addressed by the guidance [88, 89].

An example of one such technique is coverage testing using Modified Condition/Decision Coverage (MC/DC). This is mandated for the highest DALs (i.e. DAL A). This testing approach is aimed at exploring the code structures of a software system. In many

contexts, it can be effective at identifying software errors in the system and can often provide a high degree of confidence in the *logical correctness* of the system [90]. However, it has been noted that within conventional systems this form of testing may be of only limited use in identifying certain behaviours: it can be sensitive to some code structures and may be no more useful in some contexts than other, less 'expensive' analysis approaches [75, 91, 92, 93]. Moreover, in the context of BNSs, the properties of these systems may further hamper the effectiveness of MC/DC testing (and other code-focussed testing approaches) due to the importance of model and data aspects of the BNS in contributing to the functional behaviours of the system. This will be explored in more detail shortly.

A further group of standards that are particularly relevant to the motivating example of this thesis are those standards derived from the International Electrotechnical Commission (IEC) 61508 standard. This standard aims to provide a general functional safety standard [94]. This generality has made it popular in a number of fields, including the nuclear, automotive, rail and process industries. A number of standards derived directly from IEC 61508 are in active use in several domains. One such derived standard – of particular note to road vehicles – is the automotive industry's International Organisation for Standardisation (ISO) 26262 standard [95]. This explicitly states that the autonomous use of AI in these systems is not recommended. As IEC 61508 lays out a comprehensive safety lifecycle for electrical, electronic and programmable systems it also makes strong assumptions about the development process and activities. Similar criticisms have been levelled at IEC 61508 as have been levelled at ARP 4754 and DO-178C, notably that systems developed in accordance with this standard may encounter issues related to the prescriptive nature of the standard, and the lack of explicit rationale that underpins the defined requirements [70, 75, 76]. There is a general lack of clarity of how the processes and requirements relate to the system in development [76]. Again, some of the novel development activities and behaviours of many AISs have been identified as being poorly addressed by the standard [96].

While the general landscape of safety engineering is still heavily reliant on relatively prescriptive safety standards, there is increasing interest in the development of conceptually distinct, flexible standards to tackle the challenges discussed previously. One example of this form of standard is the UK Ministry of Defence (MOD) Defence Standard 00-056 (DS 00-056) [97]. This standard sets out requirements addressing the safety management of systems operated by the MOD but does *not* prescribe a set of objectives for how these

requirements can be satisfied. Instead, it relies on a goal-based approach to assurance. This is achieved through the development of a safety case that explicitly argues for the safety of a system and uses evidence to support these arguments [70, 98].

This standard is augmented by the recent re-introduction of the MOD's Defence Standard 00-055 (DS 00-055) [99]. This standard specifically addresses the development of 'Programmable Elements' for safety-critical applications – with particular focus on software aspects of safety-critical MOD programmes. It adopts a complementary stance to DS 00-056 in focussing on providing a goal-based safety framework. Again, the standard encourages assurance practitioners to explicitly expose the rationale for their assurance activities and ultimately the safety of their systems.

The MOD has adopted this approach partially for flexibility; it enables developers to use relevant civilian standards where appropriate (i.e. DO-178C for aerospace systems) [100]. However, it has also been adopted to encourage more explicit reasoning when establishing the safety of the system. Goal-based approaches have been criticised by some as producing more ambiguous safety artefacts and for their often poor integration with existing development practices [73].[22] However, they have also been credited with mitigating some of the weaknesses of standards such as ARP 4754 and IEC 61508: they can ensure the reasoning behind assurance activities and development processes is explicit and relevant to the system under development. They can also reduce the degree of disruption caused by late-stage design or development changes [73, 100].

2.3.1 Safety Analysis Techniques

The successful application of each of these standards is predicated on the *effective* utilisation of safety analysis techniques that explore the underlying properties and behaviours of a system. The safety-critical systems domain has spawned a plethora of such techniques that are both based upon techniques from the broader systems- and software-engineering domains, and that have been developed entirely within the safety domain.

Understanding the general approaches to analysing the safety of a system and the specific techniques that can be applied to these systems is a necessary prerequisite for the assurance of the safety-critical systems. This section provides a top-level overview of two of the most common safety analysis techniques currently in use: Fault Tree Analysis

[22] This issue of poor integration refers to cases in which goal-driven assurance cases are constructed retrospectively, thereby undermining their potential utility.

(FTA) and Failure Modes and Effects Analysis (FMEA). These two techniques have been selected because of their ubiquity in the safety domain, and because they frame the two logical approaches to safety analysis: *inductive* and *deductive* analysis. These two terms can be defined as follows [83]:

Inductive – Any form of analysis which proceeds from a known or hypothesised condition to possible outcomes.

Deductive – Any form of analysis which proceeds through the investigation of possible causes of a specific system condition.

The logical process of all safety analysis techniques can be categorised in terms of these two definitions. While they are presented here as a dichotomy, many widely used techniques adopt elements of both forms of analysis. Perhaps the most popular technique that consists of both deductive and inductive aspects is a Hazard and Operability study (HAZOP).[23] In the case of the two techniques selected for discussion in this section, FTA is can be described as a deductive analysis technique, while FMEA can be described as an inductive technique. Before continuing into a discussion of these two techniques, the following terms must be defined [70]:

Error – A discrepancy between a computed, observed or measured value or condition, and the true, specified or theoretically correct value or condition [101].[24]

Fault – An imperfection or deficiency in the system which may, under some operational conditions, contribute to an error.

Failure – The inability of a system or component to fulfil its operational requirements.

Failure Mode – The way in which a system or component may fail.

These definitions will be used throughout this thesis.

2.3.1.1 Failure Modes and Effects Analysis

The aim of FMEA is to provide an inductive methodology for identifying how a system may fail, and to assess the relative impact of these failures upon the system [83]. This

[23] A review of HAZOP is beyond the scope of this chapter. A good introduction can be found in Pumfrey's work [83].

[24] An error may not necessarily develop into a failure if it is within acceptable limits

theoretically enables practitioners to prioritise assurance activities in order to address those system aspects that require the highest levels of assurance. The analysis uses inductive reasoning to identify a set of potential failure modes in a system, and then works backwards from this point in order to attempt to identify possible causes of these failure modes, and the safety effects these failures may have upon a system (and if applicable, system operators) [83, 102].

In some cases, the severity of the effects of a failure are also captured by the analysis. In these cases, an FMEA is sometimes referred to as a failure modes, effects and criticality analysis (FMECA).[25] When the severity of a failure mode is assessed, it is common for assurance practitioners to define the severity categories and any associated quantitative risk profiles according to a given standard or guideline. Indeed, the specific form of an FMEA and language used in these analyses are often defined by standards and guidelines. For example, both FMEA and FTA are recommended analysis techniques in ARP 4761 [68]. Within the safety domain, FMEA methodologies are a core tool in a safety engineer's repository. They have been credited with providing a means of proactively exploring a system's failure modes rather than relying on analysis techniques that may identify potential failure modes more reactively. They are also used to guide the design process towards hazardous failure modes and the development of potential mitigation techniques for these failures. A detailed breakdown of FMEA is beyond the scope of this chapter, but a more complete review can be found in the work of Pumfrey [83].

However, while the utility of FMEA as a standard safety analysis technique is well established, there have been a number of criticisms of the technique. One of the most common criticisms is that FMEA is subjective: it is developed based upon a practitioner's *internalised model* of a system and is generally not driven by empirical observations or other forms of objective evidence. Consequently, the repeatability of FMEAs is a major source of criticism of this methodology: different safety engineers may produce divergent FMEAs [103, 104]. In the most serious cases, this may result in the complete omission of failure modes, an erroneous assessment of the effects of a failure mode, or some combination of both [104, 105]. This contributes to a further common criticism of FMEAs: they may

[25] There is a degree of ambiguity in the usage of the term – the terms FMEA and FMECA are sometimes used interchangeably by practitioners. Inclusion of severity/risk assessments are typically regarded as the demarcation between techniques. More generally, the term FMEA is used to describe a range of similar but distinct techniques, further complicating matters.

inadvertently distort a safety practitioner's estimate of risk in a system. This may be particularly true of systems with unconventional properties.

2.3.1.2 Fault Tree Analysis

In contrast to FMEA, FTA is a deductive technique developed at Bell Labs in the 1960's as a by-product of the United States' ballistic missile programme [106, 107]. The FTA process begins with the selection of a failure mode (sometimes initially identified using FMEA) and the use of this failure mode as the 'top node' of a Fault Tree (FT). The safety practitioner then proceeds deductively from this failure mode by identifying causal factors leading to this 'top node'. These are typically represented using only "AND/OR" logic gates. The intent of an FTA is to reveal *combinations of events* that may lead to the occurrence of the top-level event (i.e. the failure mode). As the definition suggests, the conclusions drawn from FTA are logically irrefutable *provided* the structure of the FT is correct and complete. Figure 2.8 shows an example of a simple FT capturing a classic fire-protection system failure [108]. A full review of the construction of a FT and the symbols used is beyond the scope of this chapter but can be found in the work of Barlow and Ericson [106, 109].

An important feature of FTA is the integration of quantitative information into the FT in the form of the assignment of event probabilities to each of the events in a given FT (in the case of the tree shown in Figure 2.8, these are captured by the rectangular symbols in the tree). This enables safety practitioners to estimate the probability of the occurrence of an outcome (represented with the circular symbol in Figure 2.8) by evaluating the probability of a sequence of events occurring over the whole FT. A by-product of this capability is the ability to evaluate which aspects of a system's design is most important to a given outcome [106].

The probabilistic properties of FTs has resulted in the development of techniques for the translation of FTs into BNs to overcome some of the limitations of the logical structure of FTs: most notably the constraints imposed by the logical AND/OR gates most commonly used in their construction [110,111]. Indeed, some research indicates that the flexibility and robustness of BNs may provide a practical alternative to 'traditional' FT-based analysis techniques.

Again, despite its ubiquity and general popularity, FTA has received a number of criticisms. For example, some studies indicate that the generation may not be as objective

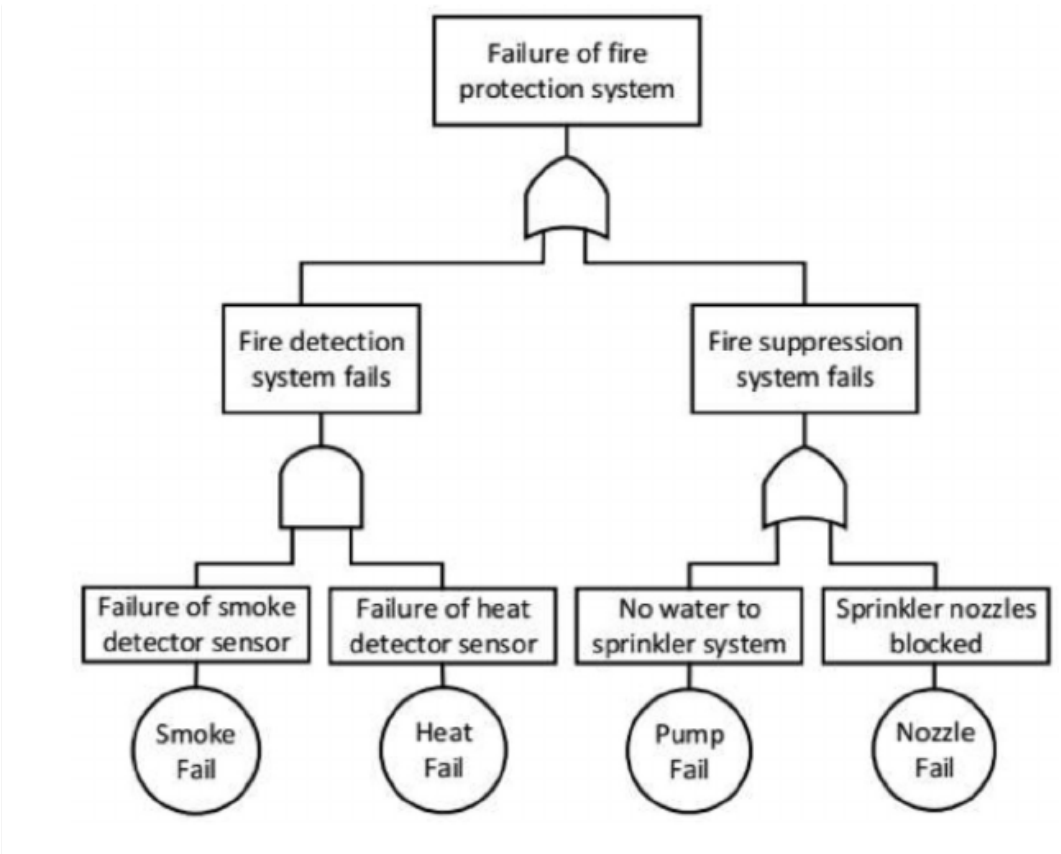


Figure 2.8: A simple Fault Tree for the classic 'fire alarm' problem. [108]

or repeatable as sometimes thought; for complex systems, independent experts can produce FTs with dramatically different estimates of failure rates or failure sequences [112]. Indeed, some practitioners point to FTAs as being potentially misleading as the quantitative nature of a FT can result in disproportionate confidence being expressed in aspects of a system.

2.3.2 Analysing Bayesian Network-based Systems

While there are many existing safety analysis techniques, it is important to explicitly consider the applicability and utility of these techniques in the context of a given class of system. This subsection provides a brief critique of existing safety analysis techniques in the context of BNSs. For example, there may be limitations in the level of detail that existing approaches can provide on the root causes of certain BNS-specific failure modes. An example of this may be in the case of FTAs: these analyses are based upon highly constrained causal structures that are often appropriate for many failure analysis problems. However, they cannot capture more complex causal interactions due to the constraints of their tree-structure.

For example, the causal sequence that produces an error mode in a BN model may be dependent on interactions between model elements that cannot be well-approximated using a FT: discrete, binary logic may provide – at best – only a coarse approximation of the root cause of BN model errors (e.g. error modes may occur as a consequence of the contributions of arbitrarily many continuous-valued variables). Indeed, the limitations of FTs in capturing certain causal properties in a system is a contributing factor to the use of BN models in state-of-the-art failure modelling (such as in the motivating example of this thesis) and diagnostic systems as opposed to FTs.

Similarly, FMEA-like analyses can provide a proactive exploration of which system aspects are of particular importance to failure modes in a BNS. As with conventional systems it can be used to guide preliminary design activities. However, in practice the methodology will be too high-level to provide detailed insight into the contribution of many aspects of a BNS to failure modes in this class of system; once again, the ability of this methodology to facilitate the useful analysis of some aspects of a BNS (particularly model and data aspects) at anything other than the most coarse-grained analysis is doubtful.

In conclusion, both FTA and FMEA approaches will provide useful insights into a subset of system aspects and behaviours of a BNS. However, they are likely to encounter serious limitations in their ability to provide sufficient detail with respect to the specific

error modes that may be encountered within a BNS. It may still be possible to categorise BNS error modes using a more general framework. However, such a framework has not yet been developed. Finally, current analysis techniques cannot provide the level of detail necessary to evaluate the *contribution* of certain important BNS error modes to system-level (i.e. UAS-level) failures: this also requires a new approach.

2.4 Safety-Critical Uses of Artificial Intelligence

For much of their existence, the fields of Artificial Intelligence and safety engineering have existed largely independently. In some ways, this is surprising. Since its conception and initial popularisation in the post-war years, AI has been touted as a solution for a plethora of commercial, military, industrial and private applications. Many of these proposed applications have clear (and potentially extremely serious) safety concerns associated with them. Despite this, the published academic literature on the safety concerns and ultimate assurance of these systems is at best scarce, and in some cases non-existent. This has arisen due to a combination of the highly application-focussed nature of much safety research and of the engineering domain more generally, and the fact that (until very recently) the prospect of large-scale deployments of complex AISs into safety-related roles has been a remote prospect.

2.4.1 Phases of Safety-Critical Artificial Intelligence Research

This situation – which has persisted for several decades – is changing rapidly. An increasing number of major commercial and industrial interests have committed to developing new products and platforms with AI at their centre. This includes many major automotive, rail and aerospace manufacturers, medical organisations, and governments. The adoption of AI in almost every sector is currently expected to accelerate. Many of these applications will take on greater degrees of autonomy than have been previously possible. The fact that these systems will require new – or heavily augmented – development and assurance practices is increasingly recognised [21, 64, 72, 84, 86, 113, 114]. To date, research into the development of safety-critical AISs remains comparatively limited, there have been at least three significant waves of research into the problem. These can be defined as follows:

2.4.1.1 Phase I - Expert Systems

Throughout the 1980s, academic research and the commercial application of so-called 'Expert Systems' (ESs) boomed [115]. During this period the first body of research into the safety of AISs began to emerge. This research followed the general trend in the field of AI and focussed almost exclusively on the assurance of ESs. Unlike modern AISs, many of this generation of AISs relied on significant human input due to the comparatively limited ability of ESs to 'learn' from data. Instead, these systems relied heavily on logical conditions (i.e. rules) encoded by human experts. These systems broadly reflected the early tendency in the AI field to believe that intelligent behaviour could be encoded with a sufficiently specific set of logical rules – and could be programmed by a human expert [3, 115]. The decline of research into ESs is generally attributed to the shortcomings of many of this class of system: they failed to capture the subtleties in many reasoning tasks and became computationally intractable in real-world applications [2, 3, 116].

During this period, a significant contribution in the field was the work of Culbert *et al* on behalf of NASA. This work looked at the development of methodologies to support verification and validation of rule-based ESs [85,117,118,119]. In one paper, Culbert identifies an assurance consideration that remains relevant in current AISs: that the enumeration and refinement of requirements for an ES does not – and frequently cannot – proceed as in conventional software systems. Culbert states [85]:

> *"Some expert systems can probably be developed by using conventional software engineering techniques to create software requirements and design specifications at the beginning of the design phase. However, the type of knowledge used in other expert systems doesn't lend itself to this approach."*

Consequently, Culbert suggests that the development cycle for an ES is perhaps best represented with some modification of the 'Boehm's Spiral' methodology, similar to that shown in Figure 2.9. Boehm's Spiral was developed to highlight and support the continued iterative development of a software system while maintaining a structured requirements documentation process [120]. A further observation Culbert makes (with relevance to modern AISs) is the following:

> *"Another difficulty in writing complete specifications is that some expert systems deal with problems where there is no correct, absolute answer, only more or less adequate answers."*

This touches on a recurrent theme of the complexity and vagueness of the tasks for which ESs were often developed to perform – and themes discussed in previous sections. The difficulty in defining exhaustive and/or meaningful requirements for many of these tasks is a recurrent challenge in AISs generally [85, 86]. This is a repeated observation in other work in ESs at the time and is still commented upon in contemporary literature [2, 21, 64, 80, 81, 113].

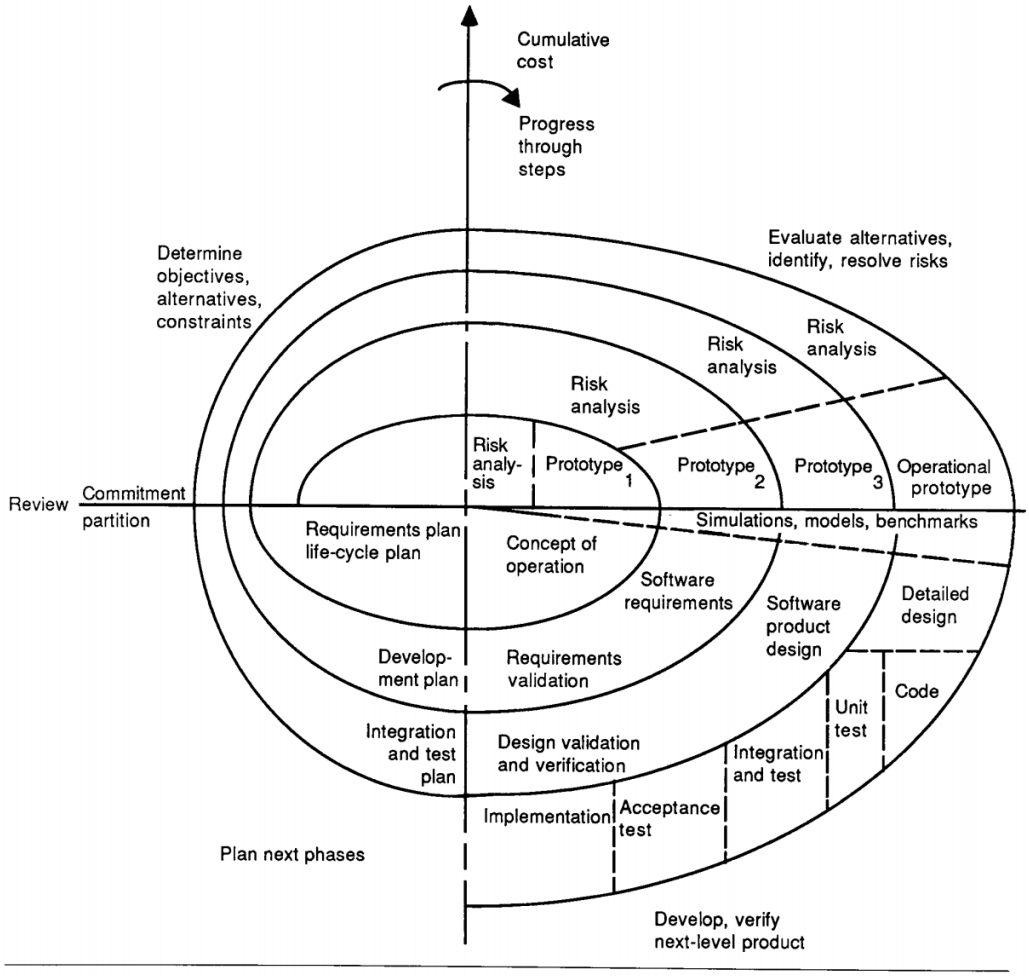


Figure 2.9: A visualisation of Boehm's proposed software development process. [120]

Beyond Culbert's work, other contributions during this time include research into the evaluation of ESs used in medical applications by Spiegelhalter *et al*, and early contributions by Rushby on the 'Validation and Testing of Knowledge-based Systems' [121]. Other contributions were made in domain- or context-specific applications for ESs, though these were generally little more than position papers [116]. With the general decline of research and commercial interest in AI in the late 1980s, research into the assurance of ESs similarly

faded. However, the applications of ESs mirror many proposed (or deployed) contemporary applications of BNSs; and so some of the insights gained from this research remain relevant.

2.4.1.2 Phase II - The Second 'AI Winter'

During the late 1980s and throughout the 1990s, AI as a field of research became relatively stagnant.[26] While advances were made, the pace of innovation slowed, and interest in the application of AI to real-world problems waned. However, during this period a number of comparatively significant advances were made in the assurance of AISs for safety critical applications. These advances broadly centred on the assurance of Artificial Neural Network (ANN) for safety-critical roles. In particular, they addressed use-cases of ANNs as adaptive controllers in safety-critical systems. Several major industrial organisations instigated programmes to develop (or attempt to develop) systems of this kind [72, 122].

Of these programmes, the largest and most insightful contribution was made as part of an effort by NASA to develop an 'Intelligent Flight Control System' (IFCS) using several ANN architectures and multiple ANN learning algorithms. The IFCS programme involved the live deployment of an ANN-based AIS aboard a NASA jet – arguably the first published, overtly safety-critical application of a complex AIS. The IFCS was envisioned as a system that could dynamically compensate for damage to the flight surfaces of an aircraft, theoretically enabling a pilot to maintain control of the aircraft in circumstances which would typically result in the loss of the aircraft and the pilot [64]. In this context, errors in the outputs of the IFCS would directly affect the control input of a pilot. Errors in the control inputs of the aircraft could lead to the complete loss of an aircraft. The IFCS was therefore identified as a safety-critical software system, and therefore subject to standard certification activities.

The IFCS programme produced a guidance document summarising the experience of the teams working on the project. This document provides a detailed summary of the considerations and practical experiences of ANN developers and safety engineers working on the project, techniques that were used or developed for the IFCS programme and

[26] The 'First AI Winter' is said to have occurred in the 1970s – the early optimism about the feasibility of the development of useful AI applications began to fade, and commercial and research funding was cut dramatically. This is often attributed to the overly grandiose claims of the AI pioneers who set extraordinary (and unattainable) expectations for the both the public and funding bodies.

key lessons resulting from the development of the system. From this perspective, the IFCS 'Methods for the Verification and Validation of Artificial Neural Networks' document is perhaps the most complete piece of guidance on the assurance of AISs currently in existence. While a full review of the document is beyond the scope of this chapter, a number of important points are worth highlighting. Firstly, the document differs from conventional software assurance guidance documents by providing comparatively little detail on the assurance of conventional software aspects. Instead, the document focusses on the *augmentation* of existing guidance (in this case IEC 12207) and advocates the introduction of new evaluation and testing techniques and procedures [64, 123].

Amongst these new techniques are a number of approaches that can be considered to be software-independent: they directly address the architectural properties of a given ANN and aim to evaluate the dynamics of the ANN model from a purely mathematical perspective. For example, this includes an analysis of all *activation functions* used within an architecture, and the *weightings* associated with each node within the ANN. The guidance goes further by advocating the use of an FMEA-like analysis of an ANN's components (i.e. nodes, edges and activation functions) to analyse how the ANN may produce error modes. In each case, the emphasis is placed on the ANN in the abstract and is largely devoid of software-centric implementation considerations. The general sentiment of much of the guidance is summarised in the following contemporaneous quotation from Lisboa[27]:

> *"In emerging computation, the complexity is often not in the software implementation, but in the interpretation and testing required to evaluate the operation of the model."*

More generally, the IFCS document advocates the need for additional appendices and processes within a standard to tackle the iterative development practices associated with the development of a system utilising an ANN. The document outlines how these processes should focus *specifically* on addressing the architectural design and evaluation of the model, and the unconventional considerations associated with the training and deployment of an ANN. At a high-level, the guidance reflects the position of Culbert and the early ES assurance researchers: the development of this class of AIS is an iterative process that relies heavily on comparatively ad-hoc design decisions and iterative refinement of requirements

[27]Lisboa did not work on the IFCS project, but has published work on the evaluation and testing of AISs in safety-related contexts.

and associated testing. Indeed, in discussing the design and development of the ANN architecture, the NASA team state:

> *"... the selection of the number of hidden layers and the number or neurons within each layer is more of an art than a science. Similarly,the V&V analysis of these choices is likewise an art form."*

This reinforces the shift – both conceptually and practically – in the development of an AIS utilising ANNs or other complex, abstract mathematical artefacts common to AISs. This is also highlighted within other independent outputs from the IFCS programme, where a failure to grasp the distinctions between conventional software systems and AIS-specific aspects is claimed to result in:

> *"' ... [producing] long development times or project delays, improper [ANN] configurations, failure to find adequate solutions, etc."*

While the by-products of the NASA IFCS programme are arguably the most significant contributions to the field in this period, several other important contributions were also made. One notable contribution (also cited in the IFCS guidance), is the work of Kurd *et al.* In their work, Kurd attempts to address a recurrent challenge with many AIS approaches: interpretability. Historically, ANN developers (and assurance practitioners) have considered the analysis of an ANN-based system to be a 'black box' problem. To compensate for this, Kurd proposes a methodology for constraining the functional properties of the ANN by injecting rules based on available knowledge into the ANN, and then extracting and refining these rules iteratively. The safety lifecycle developed as part of this work is shown in Figure 2.10.

This approach has a number of merits, chief amongst which is a coherent process for the structured, systematic development of a safety-critical ANN-based AIS. However, the process has a potential weakness: the tendency of safety engineers in software assurance generally is to impose strict constraints on the software. In conventional software, this is a well-established practice: a constraint will have well defined, deterministic effects on the behaviour of a system. However, this may not be true for AISs. The power of modern AISs comes from their ability to *learn* from data. Some architectures and training algorithms exhibit a strong 'sensitive dependence' on their initial conditions (architecture and parameterisation) and training algorithms: their behaviours can be dramatically altered by perturbations in this data or by modifications to their architectures [64, 124, 125].

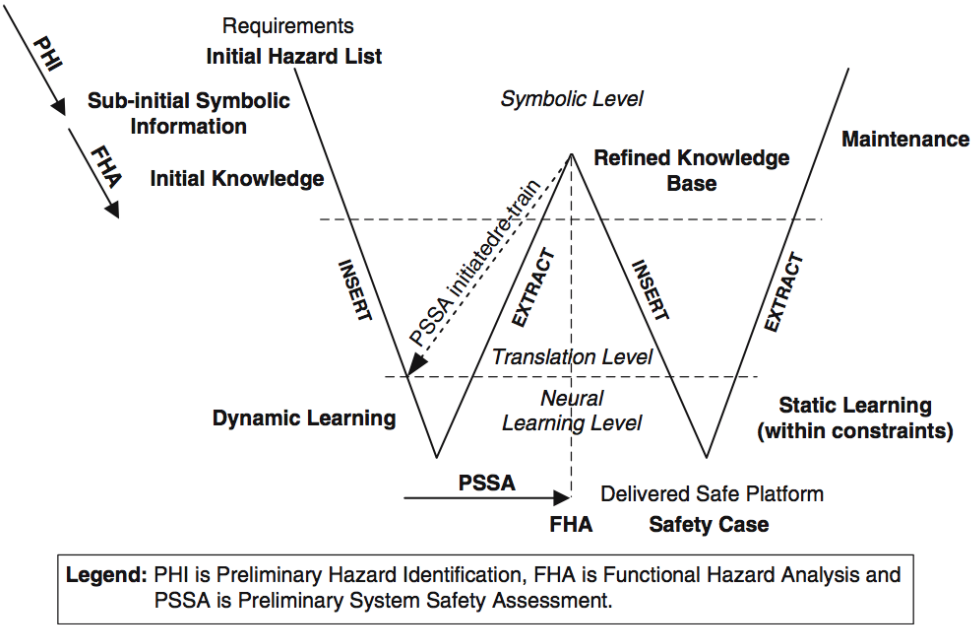


Figure 2.10: A visualisation of Kurd's proposed safety-critical ANN development lifecycle.

Without care, applying constraints will hamper the ability of the system to learn and may directly undermine the very property that makes an AIS useful in the first place. An assurance practitioner may introduce new, more pervasive, error modes into an AIS than they mitigate.

2.4.1.3 Phase III - Deep Learning

The current wave so-called 'Deep Learning' (DL) AISs that have emerged since 2012 have been used to rapidly solve (or mitigate) a range of long-standing computational problems within the field of AI [12, 28, 53, 126, 127, 128]. In general, this generation of AISs utilise ANN architectures that overcome the limitations of earlier generations of AISs through the utilisation of more powerful learning techniques and novel architectures [129,130,131,132]. Currently, these systems are extremely data-hungry. Their success has been facilitated by near-simultaneous breakthroughs in the development of learning algorithms for DL-based Systems (DLSs) and the proliferation of huge quantities of widely-available data on everything from facial recognition to natural language processing. Their success has accelerated the demand for their deployment aboard products ranging from smartphones to cars. For the first time in the history of the field of AI, these systems are on the verge of becoming both ubiquitous and truly safety-critical.

Superficially, the assurance of DLSs bears many similarities with the work carried out by previous work on the assurance of AI. However, in the intervening decade between this earlier research and current state-of-the-art DLSs, the scope and complexity of the applications of these systems has grown dramatically. For example, modern DLSs utilise models many orders of magnitude larger than those in general use when the previous wave of assurance work was introduced. Furthermore, the architectures and learning algorithms used by many DLSs rely on extracting richer representations of information from a domain with no (or limited) direction on the part of developers – this is part of the origin of the term 'deep' in DL. Consequently, some current approaches to assuring DLSs actively avoid constraining the learning of the system, and instead focus on providing increased transparency or robustness of the system's behaviours through novel analysis or training techniques [113, 133, 134].

One notable example of such a technique is the work produced by the DeepXplore team [113]. This work highlights shortcomings of traditional notions of software testing coverage in exposing potentially hazardous behaviours in DLSs. It advocates the introduction of new notions of coverage for DLSs that address model architecture aspects as opposed to code structures. The paper introduces an automated tool for exploring the parameter-space of a Convolutional Neural Network (CNN) for computer vision tasks aboard an autonomous vehicle and highlights the ability of their tool to more comprehensively explore the dynamics of their CNN models. The work bears some similarities to the earlier work of the IFCS programme in focussing on exploring model-centric aspects of the system. However, DeepXplore provides a more systematic, quantitative analysis than that proposed in the IFCS guidance, and therefore more readily accommodates the additional complexity associated with using a DLS.

Other notable techniques include the utilisation of adversarial testing approaches to produce learning algorithms that bear some resemblance to conventional fault injection techniques [133, 134]. In these cases, the aim is to improve the performance of the DLS by injecting 'faults' into the data and aiming to have the DLS learn to recognise or mitigate such faults. A parallel movement within the AI domain has been to develop training algorithms that enable a model to explain its decision making. Rather than constraining the learning algorithm as in traditional approaches, these approaches typically modify the *objective* of the DLS to include information pertaining to the reasoning of the model. In some sense, the current efforts to test and evaluate AISs are tending towards more 'holistic'

techniques that avoid the historical tendencies of assurance practitioners to attempt to exert low-level end-to-end control over system behaviours.

2.4.2 Emergent Themes

While each of the three waves of research into the development of safety-critical AISs dealt with a distinct set of AI technologies and approaches, a number of shared themes emerge across them. These themes provide some general insights into the challenges facing the assurance community in developing safety-critical AISs and are relevant to BNSs. In the case of all three, whilst the language and technical considerations vary, a consistent set of recurrent assurance concerns are evident. These can be generalised as follows:

- Existing software analysis and testing techniques may fail to identify the source of many hazardous behaviours.
- Typical approaches to the definition and decomposition of requirements are not (in general) feasible/useful.
- Conventional software development lifecycles poorly accommodate the development of safety-critical AISs.
- The complexity of the development and testing of an AIS does not lie in software implementation challenges.

In the two later waves of research, AISs have become increasingly data-driven and are powered by more complex or elaborate AI architectures. Given the similarities in the core technologies and approaches that drive the systems developed as part of these two later generations of research, a further set of shared themes emerge. These can be generalised as follows:

- Key aspects of a modern AIS's functional behaviour are – in a sense – 'offloaded' onto black- or grey-box system aspects that can be considered 'implementation independent'.
- Analysis and testing techniques that rely on exploration of code structures will not provide insight into the origin of many of the functional behaviours of an AIS's.
- Existing safety standards do not adequately address learning and adaption capabilities of AISs.

The distinction between the second and third phases of safety-critical AI research can be described in terms of the scope and power of the latest generation of learning algorithms and their underlying architectures. For example, some 'historical' facial recognition technologies would often include several pieces of pre-processing software for extracting and transforming faces in images before this was passed to the recognition system. Modern DLSs learn can be trained to extract *and* recognise faces in an image – thereby cutting out the intervening software layers. Technically, this is described as automatic feature extraction. Furthermore, the architectures of these systems are generally more elaborate, utilising many more layers of neurons than in 'traditional' ANN architectures. Indeed, in many cases, each layer of these Deep Learning ANN architectures is composed of different neuron variants.

For example, it is common practice to 'stack' convolutional layers with recurrent layers, and use these alongside 'traditional' layers common to earlier Multi-layer Perceptron-style (MLP) models. Consequently, DL models can be considered – in a very general sense – to be less *internally* homogenous in terms of architecture than previous generations of ANN-based approaches. They also significantly more *internally* complex than previous ANN architectures. Finally, they can provide end-to-end preprocessing, transformation and prediction capabilities beyond those available in earlier generations of AI techniques.

2.5 Summary of Research Problems

The survey of existing literature presented in this chapter highlights several important themes with respect to the current state of research into the assurance of BNSs for mission-critical and safety-critical applications. Fundamentally, there is a pervasive lack of concrete guidance on how to approach the development of BNSs for these applications. What guidance there is for AISs is typically targeted at addressing the assurance concerns of related, non-BNS AISs. Some of the key outstanding problems identified by this literature survey are:

1. There are no published methodologies for explicitly analysing the assurance implications of the various data, representational and algorithmic choices made during the development of a BNS. Specifically, while some published techniques may have utility in evaluating representational aspects of mission-critical BNSs, there is no existing guidance on how these techniques may be applied to a mission- or safety-critical

BNS to address the concerns of assurance practitioners.

2. There are no established techniques for communicating assurance information between all pertinent stakeholders of a mission-critical BNS. In particular, there is an absence of any concrete examples of techniques for capturing the full range of concerns that BN developers and assurance practitioners alike may have with respect to the development of a mission-critical BNS.

3. There is no standard guidance for managing the unconventional design aspects of BNSs (e.g. non-deterministic aspects of their design) across the entirety of their developmental and operational lifecycles. For example, there is no guidance on how to manage the continued validity of many representational properties and algorithmic behaviours over the course of the extended life of a system.

4. There are no published analyses of the implications of using existing software standards to certify the behaviours of mission-critical and safety-critical BNSs. There is no information on the specific shortcomings of these standards with respect to the development and operation of BNSs and any related considerations assurance practitioners may need to address in order to certify a BNS for such an application.

5. Whilst there are techniques for performing various types of analyses on BNSs, there is an absence of literature on how these techniques relate to mission-critical BNSs. There are no existing safety analysis techniques that are readily capable of accommodating the unconventional properties of BNSs. In particular, techniques that cater for aspects such as the unique role of data and abstract mathematical models in determining functional behaviour, and the non-deterministic properties of these systems are notably absent from existing literature. There is also potential for modifying BNS-specific techniques to integrate safety information directly into existing BN-focussed analyses.

6. There is no guidance and there are no established techniques for addressing the implications of the unconventional forms of safety evidence. For example, the properties of a BNS may manifest failure modes that are non-deterministic, and the corresponding evidence for the mitigation of hazards associated with these failure modes is often highly statistical in nature. There is also no guidance on how such evidence could be used to contribute to arguing that a BNs is safe. The nature and impli-

cations of evidence required to assure such properties of BNSs is often dramatically different from the evidence generated by existing software analysis techniques.

The following chapters will touch on each of these points and in particular aim to make contributions to resolving problems related to points 1, 2 and 5.

2.6 Conclusion

This chapter has presented an overview of the current state of research into BNSs and safety-critical AISs generally. The discussion throughout this chapter has focussed on those areas of BNS design and development and safety engineering that are most relevant to the motivating example of this thesis, though a number of additional concepts have been introduced to provide context for the position of BNSs within the AI field, and of the status of safety-critical AI in general. The survey of literature presented here highlights the near-complete lack of existing guidance on the development of BNSs for safety-critical roles. Some early work on the assurance of probabilistic expert systems provides perhaps the closest guidance on this problem, though the systems used vary dramatically in both technical details and applications. Whilst the field of BN development is largely devoid of safety research that directly addresses BNSs, some of the more recent work in the development of safety-critical DLSs shares strong similarities with the aspects of BNS design and development.

However, the lack of structure within the field of safety-critical AIS development is concerning. Many teams use language specific to their AI approach and, as highlighted by NASA's IFCS programme, failure to effectively communicate the properties and behaviours of an AI approach to all relevant stakeholders can have a serious impact on the effectiveness and safety of the resulting AIS. In the context of mission-critical BNSs, this highlights the need for standardised approaches to sharing BN-specific system information between BN developers and assurance practitioners. Moreover, many existing analysis techniques developed for AI development do not integrate additional contextual information.

For BNSs, this raises a second key area of research: the extension of *existing* BN analysis techniques to accommodate additional safety considerations. Such approaches are attractive as they do not undermine the development and analysis activities of BN developers, but rather add a layer of 'safety-awareness' into the training or testing of a

BNS. Finally, there remains the challenge of establishing the adequacy of activities carried out to address AIS system aspects. As repeatedly indicated in this chapter, many aspects of AIS development are – at best – relatively ad hoc. Key design decisions are frequently made on little more than the intuition of an AI developer, and the rigour of testing these systems is similarly erratic. In the case of BNSs, this indicates the need to consider how a BNS may be developed in a more rigorous fashion: one that directly establishes *why* a given technique (or set of techniques) is relevant and adequate for the assurance of a given system aspect.

Chapter 3

Establishing Verification and Validation Objectives

3.1 Introduction

The assurance of any system is predicated on a comprehensive understanding of the nature and behaviours of that system. As discussed in Chapter 2, this can be challenging in the case of Artificial Intelligence-based Systems (AISs) generally, and Bayesian Network-based Systems (BNSs) specifically. To date, literature focussed on the assurance of AISs has typically considered high-level aspects of AIS assurance. In general, this work does not provide any concrete, technical insight into how this class of systems can manifest hazardous behaviours. In contrast, the Artificial Intelligence (AI) domain has predominantly explored the practical and theoretical understanding of the approaches and technologies it has developed.

Until very recently, both domains have been able to maintain largely independent spheres of research and commercial interests. This has been possible as a consequence of the relative technological immaturity and various related limitations of many previous generations of AI-based technologies, and the (naturally) risk-averse position of the safety domain. In the former case, there have historically been very few compelling reasons for the adoption of AISs. Often conventional software systems offered comparable – and better understood – solutions. In the latter case, the safety domain has tended to adopt a highly cautious position on AISs, in some cases going as far as to proscribe AISs in safety-critical applications [94].

Consequently, the existing repertoire of vocabulary, tools and concepts familiar to as-

surance practitioners is poorly equipped for the detailed technical description and analysis of AISs. Moreover, there is a similar lack of analysis tools and techniques in the AI domain that explicitly address system safety considerations. This is a significant barrier to ensuring all relevant system stakeholders understand precisely *what* has been built. By extension, without this mutual technical understanding, the assurance of this class of systems may not be possible.

This chapter presents a framework that aims to provide a structured, comprehensive and mutually intelligible approach for describing BNSs. From an assurance perspective the development, properties and behaviours of BNs differs markedly from conventional software in a number of ways. The framework introduced here has been designed to explicitly capture the ways in which these unconventional aspects may directly or indirectly influence the functional behaviours of a system. It is intended to provide a shared conceptual basis for the description and understanding of BNSs in general, and to expose these distinctions to safety practitioners for subsequent analysis.

This framework is then built upon to develop a set of generic BNS verification and validation objectives that have been defined to provide a concrete enumeration of system properties that must be explicitly addressed by developers and assurance practitioners. A methodology for generating system-specific objectives is then introduced. The objectives aim to provide a systematic approach to developing assurance objectives for BNSs, and a first step towards a comprehensive approach to assuring this class of system. They have been defined with the intention of providing a means of comprehensively addressing system aspects that are inadequately addressed by existing software safety standards.

The chapter is structured as follows. First, an overview of how and why a BNS may fail is introduced, and a consolidation of existing work on error modes in BNSs is provided. This discussion is then used as the basis for the introduction of a set of system viewpoints for BNSs. These viewpoints are subsequently developed into a Reference Model for Bayesian Network-based Systems (RM-BNS). Next, this reference model is used to develop a set of generic verification and validation objectives. A methodology for generating system-specific objectives is then introduced.

3.2 System Error Modes

The field of AI is notable for its extraordinary diversity. There are a number of distinct schools of thought, each with a rapidly developing and ever-expanding set of techniques

and concepts. What constitutes 'AI' can vary between practitioners, and the language and approaches used reflect this internal variation. In many cases, this diversity of ideas and language is a direct result of the highly cross-disciplinary nature of the field [1, 2, 63]. Consequently, technical concepts within the domain can be challenging to grasp – in part due to the overlapping language and semantics of different fields and research areas. However, at a high-level, there are a set of general concepts and techniques that unify the domain. A clear, succinct description of the field is provided by Wilson [135]:

> *Artificial Intelligence is about algorithms, enabled by constraints, exposed by representations, that support models targeted at thinking, perception and action.*

This description is compatible with the area of AI concerned with Probabilistic Graphical Models (PGMs), and by extension, Bayesian Networks (BNs). This statement can be further condensed by stating that the field of AI (and by extension BNs) is principally concerned with the intersection of three primary areas: representation, algorithms and methodology [1, 2, 135]. Each of these areas is focussed on the development of aspects of AI that may independently introduce erroneous behaviours into a completed AIS.

This section provides a high-level overview of the concerns associated with these three areas. The aim is to provide a general conceptual understanding of how errors may emerge in BNSs and what effect these errors may have on a system's behaviours. It highlights how these errors differ conceptually from those common to conventional software systems. The section concludes by providing a summary of existing research into BN error modes and their corresponding references. A consolidated enumeration of this work has not yet been presented within literature.

3.2.1 Representation

The selection of an effective approach to representing a given computation problem is a critical first step in the development of any AIS. Within the AI domain, the area of representation is focussed on producing conceptual and (typically) mathematical mechanisms for capturing a given problem. Common representational approaches include Artificial Neural Networks (ANNs) and, of course, BNs. Research in this area is motivated by the need to produce compact, efficient solutions to notoriously complex problems. Indeed, these representations are often applied to problems that may be intractable for conven-

tional, explicit, imperative programming approaches.

In some fields, BNs and related approaches and variants are among the most popular representational frameworks currently in use for this reason. The BN representational framework lends itself well to complexity and uncertainty inherent in many diagnostic and causal reasoning tasks [22, 60]. Behaviours and knowledge can be *learned* rather than manually programmed. This has made it a popular approach for many state-of-the-art medical applications [2, 56].

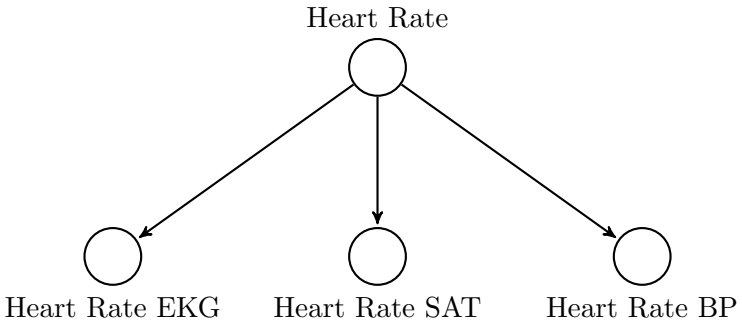


Figure 3.1: An example of a fragment of a BN model representing the conditional relationships of factors effecting a patient's heart rate [136].

Figure 3.1 shows a fragment of a toy medical BN model of a patient's heart rate that illustrates key features of *local* and *global* representation considerations.

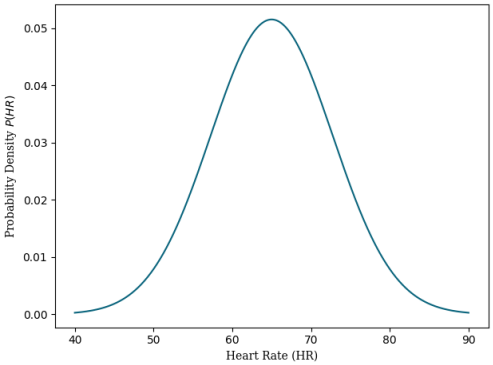
Each of the nodes in the model represents a Random Variable (RV). In this case, the Heart Rate RV is modelled as hidden: the model assumes that it cannot be *directly* observed. Instead, the patient's heart rate is modelled as a variable whose state can be *inferred* through observations. This more accurately reflects the state of the world: a medical practitioner invariably relies on a machine's *estimate* of a patient's vital signs, they do not observe it directly. The BN fragment in Figure 3.1 therefore integrates information from three distinct observations in order to provide a more accurate probabilistic estimate of a patient's current heart rate. This approach can compensate for errors in one of the inputs and dynamically weight the estimate of the patient's vital signs to those observations that are less uncertain. This can produce adaptive, precise monitoring of signals that can outperform other approaches. Indeed, for these types of applications BNs have been shown to provide an optimal estimate of hidden states (such as a patient's heart rate) – provided a number of formal mathematical properties hold.

Formally, each edge in the fragment represents the conditional dependence of these variables upon one another. The properties of BN models are such that each of these RVs

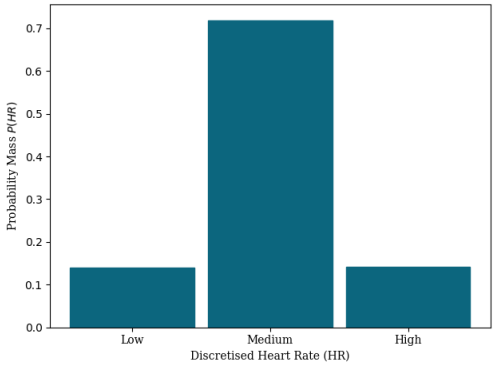
can be considered as a 'component' – or local – model of the global BN model. The local model can be considered to be the conditional relationships of an individual RV, and the *internal* structure of the RV's distribution. This internal structure may be a simple table-structured distribution, a decision tree, or an ANN. The global model refers to the BN model taken together. A discussion of the structural properties of BN models is presented in in section 2.2.3.

3.2.1.1 Local Representation

The composition and structure of these models – both locally and globally – may directly influence the functional behaviour of a BNS. For example, consider again the BN fragment shown in Figure 3.1. Here, the ECG node captures a probabilistic model of the measurements of an ECG machine. There are a number of important modelling decisions associated with capturing an ECG measurement using this local representation. Figure 3.2 shows two toy probabilistic models for illustration purposes. Figure 3.2a shows a continuous probabilistic model of a patient's heart rate measurements - the measurements of the machine are characterised as being normally distributed. This is a standard, theoretically well-founded probabilistic model for many measurement problems similar to this [20]. In contrast, Figure 3.2b shows a discretised version of this distribution, where the continuous-valued sensor measurements are binned into low, normal and high categories.



(a) A normal distribution fitted to a patient's heart rate (HR).



(b) Visualisation of weights in an ANN trained for handwriting recognition.

Figure 3.2: A *discretised* normal distribution fitted to a patient's heart rate (HR).

The selection of the representation of these local models has important consequences for the capabilities of the global model (i.e. the BN taken as a whole). Error modes may emerge in a BNS directly or indirectly due to errors associated with local models.

In the case described here (choosing between representing an ECG sensor as a normally distributed set of measurements, or a discretised categorical model) there are two principal consequences of this decision:

1. The structure of the model is constrained if a RV is represented as a normally-distributed variable; the BN representational framework does not support representing a normally distributed RV as having discrete child RVs [2, 3].[1] This limits the expressive power of the resulting model. Similar constraints include the inability of the BN framework to represent recurrent or cyclical relationships within its structure. Constraints of this kind can result in excessively strong modelling assumptions, which may in turn compromise the effectiveness of a BN model. Error modes may emerge as a consequence of these limitations in local models – constraints introduced by local structures may prevent a BN from capturing information necessary for reasoning and fulfilling its intended functionality.

2. The discretisation of a continuous-valued RV necessarily suffers information loss during the discretisation process [137]. Concretely, the discretisation of a continuous-valued RV requires the specification of a finite set of states to represent a distribution with an infinite number of states. In the example in Figure 3.2b, a developer has represented a patient's heart rate as being in either a high, medium or low category. However, from an assurance perspective, there is the question of whether or not these categories are sufficient. For example, if an additional category is added (e.g. very high, very low), a model's diagnostic performance may change. This is because access to this additional information may boost its reasoning capabilities – in the same way a doctor may refine their reasoning if presented with more detailed information. Without this additional state, the global BN model is 'blind' to any information it may convey to the broader diagnostic problem. However, an excessive number of categories in this local model may produce the same effect as having too few categories – the model may actually decrease in performance. Error modes associated with this can be considered as being manifestations of excessive *local* bias or variance

[1] There are workarounds to this problem, with a number of proposals for solving this challenge. However, they can be regarded as non-standard approaches.

for a given model.[2]

The examples shown in Figure 3.2 are toy examples. Practical applications of these systems often demand the utilisation of high dimensional probability distributions. In these contexts, intuitive visualisation and analysis of RVs is more challenging. Furthermore, even in comparatively simple cases such as that in the example given, a modern medical diagnostic BN may be comprised of thousands of such variables. A detailed review of the modelling decisions taken for each RV may be extremely labour intensive.

3.2.1.2 Global Representation

The discussion so far has focussed on some of the properties of local models in a BN. However, there are several key decisions associated with the particular variant of the BN representational framework selected. These decisions play a role in the functional behaviour of a BNS. Many diagnostic systems are developed as static models. There is an assumption that the model is time-invariant: the model does not depend on any other previous time aside from the present [2]. For medical diagnosis, for example, this may well be a sound assumption. However, it will not be the case universally.

An example of this is the use of BNs for some control and navigation tasks aboard autonomous vehicles. The state of the road and the location of the vehicle provides important information about where the vehicle *will* be and the state of the road at a future time. The BN framework supports the representation of this class of problem. Dynamic Bayesian Networks (DBNs) can be used to model temporal dependencies of this kind [2,

[2]This alludes to the so-called 'Bias/Variance Tradeoff'. The Bias/Variance Tradeoff is a key concept in applied statistics, Machine Learning (ML) and AI. It addresses the need to produce models that are sufficiently coupled to their target application that they produce useful outputs, but that are not so strongly coupled to their training data that they perform poorly when used in 'live' operational contexts. The aim of a good AIS is to provide a well generalised representation of a problem – one in which the system can perform well in contexts outside of those in its training data. This means BN developers will typically select models that manifest lower 'absolute' performance on training data in favour of a model that demonstrates better generality. For example, a medical diagnostic BN model may achieve 99% accuracy on training cases, but may achieve only 75% accuracy in test cases. A second model may achieve approximately 90% accuracy in both training and test cases. Typicaly, BN developers will select the latter model: it demonstrates more stable, generalised performance, while the former can be described as 'overfitting' to the training data. Models with high complexity are more susceptible to overfitting; adding too many states to a model may cause it to demonstrate poor generalisation performance.

49,138]. However, there are once again important limitations on what these dependencies may represent. The selection of the specific BN representational framework variant is a key step in the development of a BNS. Failure to represent temporal dependencies within a model may produce erroneous model outputs – a model cannot reason about something it does not 'know'.

A further consideration for the representation of a problem domain as a BN model is the utilisation of object-oriented (or template) models. In many applications of BNSs, aspects of the problem being modelled manifest self-similar properties that lend themselves well to object-oriented (OO) approaches. These OO approaches facilitate the re-use of structures – including sets of RVs, BN fragments or complete BN models – within a single larger BN model.

For example, consider a theoretical use-case of a BNS used to monitor the launch of a Space-X Falcon Heavy launch vehicle. As with other launch vehicles, the architecture of this system exhibits a number of self-similar properties: the system is composed of three similar booster sections and each booster is comprised of nine Merlin 1D engines [139]. From an OO modelling perspective, a BN model of such a system may represent each engine, and then each booster as objects. This would involve the definition of a single BN model object or model fragment for each engine. These objects would then be replicated nine times and integrated within a single BN model of a booster. In turn, this single booster model could be replicated three times and integrated into a model of the whole launch vehicle. This enables a more compact representation of the problem.[3]

These OO approaches minimise the modelling challenge in the case of complex applications [4,141]. They reduce the burden of specifying structures and parameters in a BN model by reusing structures and parameters across the model. This can help reduce issues arising from the bias/variance trade-off common to AISs. However, these approaches often rely on strong assumptions about the degree of similarity between the system aspects modelled. The reasoning for the justification of the use of OO approaches is sometimes not made explicit. They do not rigorously explore or demonstrate why such assumptions hold across all objects; face-validity is the most commonly used justification.

As a simple example, while the use of an OO for modelling engines may seem reasonable, a number of problems may present themselves. For example: engines on the

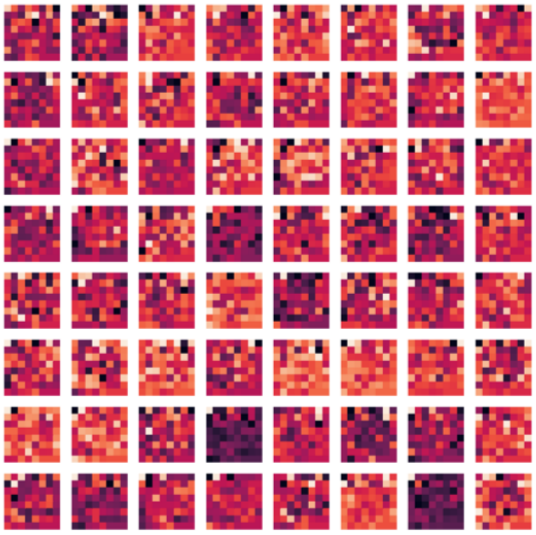
[3]For a discussion of OOBNs with examples, see [4] and [140].

central booster achieve a higher altitude before separation than the two other boosters and may therefore experience a greater range of atmospheric effects as a consequence. If a BN engine model object does not account for this potential variation, it may have unexpected effects on the system's reasoning capabilities. For example, it may assume an over-simplistic representation of the dynamics experienced by the central booster, and consequently produce erroneous outputs. This may produce a similar category of error as that experienced by the European Space Agency's (ESA) Schiaparelli lander. In this case, an excessively simple model of the atmospheric dynamics of the vehicle during its landing resulted in the complete loss of the vehicle - despite the software executing exactly as intended [142].

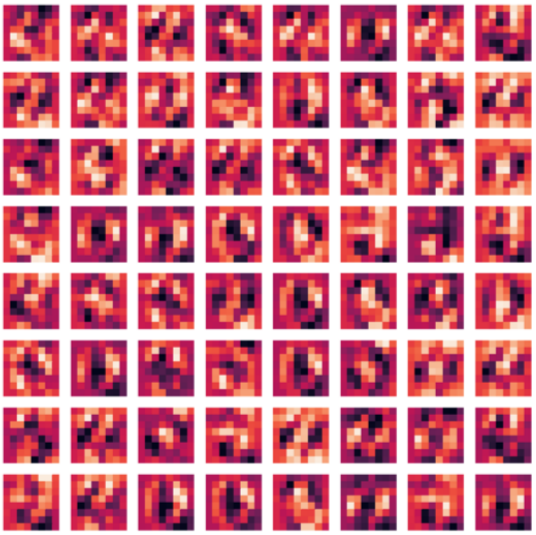
A BN model artefact, *what* it represents, and *how* it is represented, are primary determinants of the functional behaviour of BNSs. In the same way a single hardware platform may support many software systems, a single software platform may support many BN models. In BNSs, the system may not be explicitly programmed by a human developer. Instead, the BN models used by the software system act as the system's programming. This may be without any concrete changes to the underlying software itself (in a traditional sense).

This is perhaps most intuitively illustrated by considering ANNs as the general concept is transferrable to BNs. Figure 3.3 shows aspects of two ANNs with identical model architectures that have been trained to provide two distinct functional capabilities. Figure 3.3a visualises an ANN trained on a facial recognition problem, while Figure 3.3b shows an ANN trained on a handwriting recognition task. The systems run with identical code, using identical system resources. The only distinction in this case is the specified path to training data, and the ultimate parameterisation of the respective ANNs. These parameters are therefore the primary source of distinction between the two systems.

However, the parameters should not be considered independently of their parent ANNs. For example, the importance of the parameters in the ANN models are related to the exact structure of the ANN and the activation functions used by neurons within the model. Consequently, understanding the functional behaviour of these systems is fundamentally a problem of understanding the respective model's dynamics, and these dynamics are - in principle - independent of any given software implementation. As stated, these concepts also hold true for BNs.



(a) Visualisation of weights in an ANN trained for facial recognition.



(b) Visualisation of weights in an ANN trained for handwriting recognition.

Figure 3.3: Visualisations of the 'hidden' weights of two identical ANN architectures trained for two distinct tasks.

3.2.2 Algorithms

The second principal focus of the AI domain is on the development of algorithms for the efficient and effective utilisation of the abstract mathematical representations and modelling techniques for practical applications. Breakthroughs in this area have resulted in the deployment of AISs using domain representations and techniques that were erstwhile computationally intractable [126, 130]. For BNSs, these algorithms facilitate two primary capabilities: optimisation and reasoning. Algorithms aimed at optimisation are focussed on extracting estimates for the structure and/or parameters of a given model. In contrast, algorithms supporting reasoning capabilities are used to query a model. The output of these queries can be used to directly inform the behaviour of a BNS, or to otherwise inform subsequent decision-making or control systems that may be dependent upon it.

3.2.2.1 Optimisation Algorithms

Optimisation algorithms (commonly referred to as *learning* algorithms) have enabled AISs to provide solutions to technical challenges that have been beyond the ability of human developers to program manually. For example, many computer vision, autonomous system control and natural language processing tasks are not well-suited to the conventional explicit, imperative programming of traditional software. For BNs, there are two primary optimisation considerations: learning a suitable structure and learning an effective set of model parameters [2, 3]. As the properties of the BN representational framework support the compartmentalisation of modelling activities, so too does it support the compartmentalisation of certain aspects of the optimisation of the model in specific contexts. This can have important consequences for the emergence of errors in resulting BN models.

In cases in which a complete data set is available, and a BN model is not represented as having any hidden or latent variables (or any shared parameters) it is typically possible to *locally optimise* the parameters of each variable within the model separately. From the assurance perspective this has the useful property that validity of the local model representation and the local model optimisation can be largely considered as distinct activities from the representation and optimisation of the global model. However, the final validity of a global BN model should not be established by independently evaluating the validity of the model's individual local models. This requires a comprehensive understanding of the interactions within a BN.

For example, consider again the simple BN model fragment shown in Figure 3.1. In

order to achieve optimal performance over the *global* structure of the model, there may need to be local trade-offs that make aspects of the local model strictly sub-optimal in isolation. For example, in order to ensure certain properties of the 'Heart Rate' variable, it may be necessary to modify the distributions of the child nodes (i.e. Heart Rate EKG and Heart Rate SAT). Common modifications include injecting various forms of noise during the training of a local structure, or setting the distribution of a local structure to some form *prior* to training the model.[4]

In terms of assuring a BN model, the properties of all parameter optimisation algorithms selected for use in the BNS will directly determine many of the properties of the resulting BN. For example, in some contexts, the optimisation of a model's parameters may require the convergence of an optimisation algorithm towards a given solution [9, 52]. The exact convergence properties of these algorithms, and their implications for any resulting model must be carefully assessed. Errors arising from poorly converged local parameter estimates can propagate in subsequent reasoning tasks and produce errors elsewhere in a model [2, 52].

Furthermore, the properties of learning algorithms are often tuned by hyperparameters. These hyperparameters are used to modify the rate of convergence of optimisation algorithms, or the sensitivity of an algorithm to noise in data ingested by the algorithm [1, 2, 52, 65]. A parameter optimisation algorithm may produce a poor parameter estimate as a consequence of any combination of the following circumstances: the specification of erroneous hyperparameters, the inherent behaviour of the algorithm given the BN model it is applied to, and the properties of the dataset it has ingested [2, 9].

Similar considerations apply to optimising the structure of a BN model. However, optimising a model's structure - or structure learning - is further complicated by the limitations of these algorithms and through the structural properties of BNs themselves. There is a fundamental challenge in structure learning in BNs in establishing a *uniquely* optimal model structure. Many structure learning algorithms have no way of distinguishing between certain model structures - two distinct models may be an I-map of the other [2]. Practically, this means it may not be possible to distinguish (from the algorithm's perspective) between two or more distinct model structures [2]. As a consequence of these properties of structure learning algorithms, the learned model structures may indicate de-

[4]These activities are forms of 'regularisation' and can boost the generality of a model. An introduction to regularisation in BN models can be found in Koller's excellent textbook on probabilistic AI [2].

pendencies that may be interpreted as causal dependencies where none exist. A further consequence of these properties is the fact that the structure learning algorithm may produce a model with a non-parsimonious parametric space. The combined effect of these algorithm behaviours may be to introduce error modes into a BNS by learning erroneous structures, or by increasing the complexity of a BN model in such a way as to render selected inference techniques intractable. This latter case may be particularly problematic for adaptive systems.

3.2.2.2 Reasoning Algorithms

Perhaps the most important reasoning algorithm for the utilisation of BNs in practical applications has been the development of the Belief Propagation (BP) algorithm [22, 32]. This algorithm made exact inference over many BN models computationally tractable. Many variants of this algorithm have been developed, with the intent of providing various computational trade-offs that are appropriate for different applications. For example, a number of algorithms exchange accuracy for guaranteed performance properties (e.g. timeliness) of the algorithm [50, 51]. Others make common algorithmic exchanges such as exchanging memory usage for CPU usage [1, 2]. An important property of this class of reasoning algorithm is the dependence of the algorithm on certain structural properties of the BN model it is being applied to [1, 2].

For example, areas of high complexity within a BN model may result in (computational and reasoning) performance changes in the chosen algorithm [60, 143]. In some cases, this may result in exact inference becoming intractable, as discussed in the previous section. This has resulted in the development of a range of approximate reasoning algorithms. These algorithms commonly utilise sampling-based or convergence-based inference approaches to achieve an estimate of the state of the BN model. The selection of an inference algorithm must therefore be paired with the model it is to be applied to. In cases in which a model's complexity is guaranteed to remain within the bounds in which exact reasoning remains computationally tractable, exact inference will naturally be preferred by assurance practitioners - errors emerging from this class of algorithm will fall within the scope of considerations associated with 'standard' algorithm concerns (i.e. complexity, timeliness, logical correctness etc.).

In contrast, the approximate reasoning algorithms may present an additional set of challenges and a corresponding set of potential error modes. Sampling-based reasoning al-

gorithms typically rely on systematically drawing samples from the probability distribution represented by a given BN model. These approaches often provide a statistical guarantee of convergence of the sampled distribution to the 'true' posterior distribution [51]. However, this convergence is often highly sensitive to the structure of the model and to the local properties of individual RVs in the model [2,51]. Moreover, the error in the estimate of the posterior distribution behaves statistically [20, 51]. By extension, this introduces an additional source of uncertainty in the posterior distribution of the model, and by extension in a 'BNS's output.

A practical effect of the properties of statistical errors arising from these algorithms may be that a BN model converges poorly to its true posterior, and thereby produces an erroneous output. In a safety-critical context, this may produce a model that under- or over-estimates the probability of a given diagnosis, thereby producing an incorrect diagnosis which may then directly inform subsequent actions of a crew or vehicle. In contrast, variational (approximate) inference algorithms produces a deterministic estimate of the true posterior within a well-defined limit. These have proven popular in contexts in which the stochastic nature of sampling-based algorithms cannot be tolerated [50, 51]. In both cases, error modes arising from the approximate nature of the algorithms themselves are of central importance and must be addressed in the design of both the BN model and the BNS more broadly.

3.2.3 Methodology

The final primary focus of the AI domain is the development of methodologies for the construction and deployment of AISs. These methodologies focus overwhelmingly on the development of techniques to tackle AI-specific development challenges. Indeed, these methodologies often overlook conventional software implementation aspects altogether [2, 6,65]. As highlighted in section 2.2.4, this is principally because the weight of development work lies in the construction and optimisation of models, the acquisition and processing of data and the evaluation and the development of tailored optimisation and reasoning algorithms.

For BNs, the range of development methodologies is expanded thanks to the capacity of BNs to integrate subjective and objective information. Consequently, BN development methodologies integrate many data acquisition, processing and model construction concepts common across the AI domain with Knowledge Engineering (KE) activities and

frameworks that support the integration of subjective data artefacts into BN models. A focus of these KE frameworks is on the effective elicitation of data from experts. It is frequently reported that data elicited from experts contains errors, including contradictions and serious under- or over-estimates of the probability of certain outcomes. In BNSs derived from data elicited from experts therefore, the selected KE framework must be robust to these errors. Failure to identify possible errors of this kind will result in a BN model that is not representative of the target domain, and by extension highly likely to produce erroneous behaviour.

While BNSs frequently use some degree of engineer-elicited information in their construction, it is increasingly common to rely partially or entirely on empirically-derived data: data that has been obtained directly from measurements, operational and maintenance records, and manufacturer statistics. It is often assumed that data of this variety are necessarily better than that derived from subjective estimates. Indeed, these data are often more amenable to statistical analysis and evaluation [1, 20, 59]. However, extreme care must be taken in the selection, processing and utilisation of data in a BN model. Errors in these aspects of a BNS development methodology will be directly reflected in the resulting system. Systemic errors, such as biases in the underlying data may produce a BN model that is similarly biased [2]. Furthermore, care must be taken in KE and empirical data acquisition activities to ensure that the data is representative. Data acquired during a system's development and those encountered in operational deployment may be dramatically different.

For example, consider a BNS developed to provide fault diagnosis capabilities for an autonomous aerial vehicle. If the BNS is developed using failure profile data obtained during flight trials, the same system may be blind to *distributional shifts* in the failure profile of the vehicle after the vehicle becomes operational. An example of such a distributional shift may be in the change in failure profile of certain vehicle components after the system is deployed to a new environment or climate. If the development methodology of the BNS is not robust to accounting for factors such as this, the completed BNS may not be representative of the target domain despite apparently successful testing and deployment phases. In such cases, the BNS may begin to diverge from the 'true' failure characteristics of the vehicle it is modelling and produce erroneous outputs. Practically, the BNS may begin to fail erratically as the vehicle it is modelling ages or is exposed to new environmental factors [21]. This problem may not be unique to BNSs but is heightened by the

scope and complexity of these systems [144].

3.2.4 Existing Work

This section has provided a conceptual overview of how a BNS may produce erroneous outputs, and some of the key sources of error modes in this class of system. To date, a comprehensive overview of existing literature on the error modes of BNSs has not been collated. Table 3.1 provides a set of references to existing literature that explore many of the aspects discussed in this section in more detail. In particular, many of the papers listed provide an insight into analysis and mitigation techniques, as well as detailed examples of how these error modes may emerge in practice. These error modes highlight two points:

1. The error modes emerge from system aspects that can be considered to be 'software independent'. They arise from abstract mathematical interactions with the underlying data artefacts used to train the systems;

2. The error modes are associated with development activities and system aspects that are poorly addressed by existing safety guidance and safety analysis techniques.

There is therefore a need to develop approaches that effectively communicate the distinctions between BNSs (and AISs generally) and conventional software systems to all relevant system stakeholders. In particular, these approaches must *expose* and *structure* the communication between BN developers and safety engineers in such a way that safety implications of BN design and development decisions are transparent to all involved. Finally, the approach must be comprehensive: it must explicitly capture the interactions and distinction between conventional software aspects of a BNS and unconventional or BNS-specific aspects. The rest of this chapter is dedicated to addressing this challenge.

3.3 Viewpoints on Bayesian Network-based Systems

As outlined in the previous section, the properties and behaviours of BNSs produce a number of error modes as a consequence of the atypical aspects of this class of system. In order to assure these systems, a comprehensive description of these properties is necessary. The full range of concerns of all relevant stakeholders must be addressed. In particular the concerns of both BN developers and safety practitioners must be clear and explicit for both parties. To-date, the integration of ideas between these two domains has been

Table 3.1: A selection of existing work on error modes associated with developing and using BN-based systems.

Errors	**References**
CPD Discretisation	[2, 59, 137]
CPD Distributions	[2, 44, 137, 145]
CPD Parameterisation	[60, 65, 143, 146]
CPD Bias and Variance	[137, 145]
Structural Dependencies	[1, 141, 147]
Variable Omission	[2, 3, 65, 141]
Model Hyperparameters	[2, 44]
Model Complexity	[44, 146, 148]
Limited Data	[52, 59, 65, 149]
Incomplete Data	[2, 60, 65, 150, 151]
Unrepresentative Data	[9, 54, 60, 65, 152]
Data Selection	[2, 65]
Global Bias and Variance	[2, 3, 9]
Structure Optimisation	[2, 44, 60, 153]
Approximate Inference	[2, 51, 154]
Exact Inference	[3, 155]

essentially non-existent. Without such integration, the assurance of these systems will not be possible. This section introduces a set of system viewpoints that aim to provide a mechanism for this integration of concepts and concerns. The viewpoints aim to provide a structured, comprehensive view of the key assurance considerations for BNSs and explicitly address the unconventional aspects of this class of system. The following terms are defined for clarity:

Abstraction - Abstraction is a mechanism and practice to reduce and factor out details so that one can focus on a few concepts at a time. It is the process of extracting the underlying essence of a concept, removing any dependence on real-world objects with which it might originally have been connected, and generalizing it so that it has wider applications.

Object - An object is an abstract model of an entity in the real world. It contains

information, and may offer services. A system is composed of interacting objects. An object is characterized by that which makes it distinct from other objects.

Representation - A representation is (1) some way of organizing, manipulating, presenting, and storing information; and (2) a visual or tangible rendering of something.

Viewpoint - A Viewpoint is a form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system. A Viewpoint Specification defines a pattern or template from which to construct individual Views, and it establishes the rules, techniques, and methods employed in constructing a View.

View - A View is a representation of a system from the perspective of a set of *concerns*. Views are themselves modular and well formed, and each View is intended to correspond to exactly one Viewpoint. A View may include representations or correspondences to elements defined in other Viewpoints.

These definitions are drawn from [156]. A more complete definition of a Viewpoint can be found can be found in IEEE 1471-2000 and is given as follows:

> *" A viewpoint establishes the conventions by which a view is created, depicted and analyzed. In this way, a view conforms to a viewpoint. The viewpoint determines the languages (including notations, model, or product types) to be used to describe the view, and any associated modeling methods or analysis techniques to be applied to these representations of the view. These languages and techniques are used to yield results relevant to the concerns addressed by the viewpoint. An architectural description (AD) selects one or more viewpoints for use. The selection of viewpoints typically will be based on consideration of the stakeholders to whom the AD is addressed and their concerns."*

Utilising viewpoints to describe a system's architecture is a well-established approach in systems engineering. The viewpoints presented here were inspired by the work of Shames *et al* on the development of the Reference Architecture for Space Data Systems (RA-SDS) [156]. This work set out to address the unconventional aspects of engineering a complex space-borne system with complex communication networks over extreme ranges (relativistic effects, extreme latency etc.). The viewpoints presented in this chapter have been developed specifically for BNSs and reflect the design and operational experience in

existing AISs in safety critical roles, the safety domain generally, and the BN development domain. They have been designed to be broadly intelligible to all relevant stakeholders, and to reflect the concerns of these stakeholders. The viewpoints can be summarised as follows:

Data Viewpoint – The data viewpoint is concerned with all aspects of BNS design, development and deployment associated with the acquisition, processing (including data transformations of any kind) and storage of data artefacts that are to be used directly by a BNS. This viewpoint is particularly focussed at evaluating the properties of all resultant data artefacts used in the definition, training or evaluation of any BN models in a BNS.

Model Viewpoint – This viewpoint addresses all concerns related to the properties and behaviours of all BN models developed and deployed within a BNS. This viewpoint is intended to particularly address assurance considerations associated with the modelling decisions and assumption, parameterisation and structure of each individual model within the completed system, as well as the behaviour of all *models*.

Computational Viewpoint – The computational viewpoint is aimed at targeting assurance concerns associated with the properties, behaviours and potential limitations of all optimisation (learning) and reasoning algorithms specific to a BNS, as well as any unconventional data flow considerations related to the utilisation of hierarchical or distributed modelling approaches [43, 157].

Technology Viewpoint – The technology viewpoint is included to explicitly account for the reasoning behind the selection of a given modelling or development framework for a BNS. It is aimed at addressing concerns associated with the initial selection and acquisition/development of physical and technological infrastructure necessary for the development and deployment of BNS in an operational environment. It also addresses the technological risk that will accommodate the utilisation of a BNS.

Operational Viewpoint – This viewpoint aims to address the concerns related to the BNS-specific considerations associated with the deployment, evolution and maintenance of a BNS, and the associated role of the operational environment on the behaviour and performance of the BNS. This includes, where applicable, climate, population demographics, localisation (language, culture etc.) and operational load to name a subset.

Implementation Viewpoint – The final viewpoint subsumes all 'traditional' concerns that are typical of a conventional software system. This includes conventional software and hardware architecture and implementation considerations, including standard function allocation and existing verification and validation activities.

These viewpoints have been defined with the aim of capturing system aspects that – if not adequately addressed – may *independently* result in the introduction of BNS-specific error modes into a resulting system. However, while each viewpoint is defined independently, it is important to note that in practice, there will be significant overlap between the concerns and activities associated with each of the viewpoints. Interactions between system aspects addressed by individual viewpoints may in fact be the most pernicious source of errors in the development of a BNS. The interactions between a model and the range of data artefacts used to generate it, or between a model and any reasoning algorithms, or indeed a combination of aspects of all three, may produce errors that are difficult to isolate using conventional techniques [64, 121].

As a step towards explicitly addressing these interactions, the viewpoints have been decomposed into a set of distinct views. This once again follows standard practice in the systems engineering domain. The views represent a specific subset of concerns associated with each respective viewpoint and provide a reduced scope within which to perform targeted analyses of system properties. Each view has then been associated with an object model. These objects provide the basis for the Reference Model for Bayesian Network-based Systems (RM-BNS) that will be introduced shortly. The definition of each view and their associated objects is provided in the following subsections.

3.3.1 Data Viewpoint

As Chapter 2 and section 3.2.2.1 discuss, many data artefacts used in the development of a BNS take on a heightened degree of importance. The role of data in a BNS is expansive; almost all aspects of the system will be directly configured by some form of data. Moreover, any system that *learns* from a given data artefact will reflect (to some degree) the properties of the underlying data it has ingested [2, 54]. By extension, a systematically biased data set, or otherwise some data set that does not adequately reflect the target domain, will produce a biased model.

Moreover, assuming some subset of the training data is 'held back' for validation activities, these systemic errors may persist within the validation set. In cases such as this,

the system may be consistent with the test cases and consequently produce acceptable performance. However, the system may manifest poor performance upon operational deployment if these systemic errors are not present in 'live' data. To anthropomorphise: the system may learn to 'cheat' the test cases. There is therefore a need to proceed with extreme care, first in the initial data selection and *acquisition* of data artefacts, and then in subsequent *processing* and *management* activities to mitigate the chances of this occurring. It is also essential that the properties of all individual resultant data artefacts are reviewed and evaluated, with particular attention being paid to their *representativeness*, *quality* and *integrity*. This applies to all data artefacts that are used directly by a BNS.

Table 3.2: The views and objects associated with the Data Viewpoint.

Viewpoint	**ID**	**View**	**Description**	**Object/s**
Data	DV-1	Acquisition	This acquisition view defines the sources, processes and personnel used to obtain all data artefacts.	Artefact, Process
Data	DV-2	Transformation	This view describes all processing (e.g. normalisation, discretisation) applied to data artefacts.	Process
Data	DV-3	Management	The management view describes the databases and assorted management activities used to store/archive and transfer data artefacts. It addresses data-specific aspects of configuration management (e.g. training and evaluation datasets).	Artefact, Process
Data	DV-4	Data Artefact	This view describes the properties of each resulting data artefact (e.g. the quality and integrity of an artefact). It addresses sources of uncertainty in the artefact/s.	Artefact

These considerations are the basis of the motivation and reasoning behind the decomposition of the data viewpoint into the four views presented in Table 3.2. The acquisition view (DV-1) specifically targets the need to evaluate the activities involved in producing a given artefact. For example, in the context of a BNS for fault diagnosis aboard an aircraft, this may involve some form of audit of the activities of maintenance crews in logging the status of an aircraft before and after a flight. Any under- or over-reported faults may produce systemic biases that may not be detected in model evaluation phases. The transformation view (DV-2) is aimed at establishing the motivation for and the appropriateness of any transformation applied to the data. In the case of discretisation of continuous data, if the information loss associated with this process is not articulated to model developers, or is otherwise inappropriate for a given application, errors arising from this may be challenging to isolate in subsequent evaluation activities.

The penultimate view, the management view (DV-3), addresses the need to carefully evaluate all storage and transfer processes associated with an artefact. In particular, this view targets aspects beyond conventional integrity and quality considerations. Instead, it is focussed on managing data artefacts in terms of assessing and maintaining the continued validity of an artefact - in particular the continuous evaluation of assumptions underpinning all data artefacts used by a BNS.

Finally, the Artefact View (DV-4) is concerned with the properties of all artefacts directly used to train and evaluate a BNS. As these artefacts are typically a product or composite of multiple distinct artefacts, this view primarily addresses only those artefacts used directly by the BNS. In particular, it addresses considerations related to the quantity, completeness, accuracy and precision of the data used to train the model.[5] The considerations associated with underlying data artefacts used in the production of the artefacts used directly by the system are addressed by the preceding views. In the abstract, each of these views can be described generally in terms of processes and artefacts. This is the motivation for the objects outlined in Table 3.2. The notion of objects will be developed more fully in following sections.

[5]In this context, considerations relating to the 'completeness' of data included should address two cases: the complete absence of data pertaining to cases or scenarios within the data, and the absence of *elements* within the data associated with specific cases or scenarios.

3.3.2 Model Viewpoint

The role of data in modern software-intensive systems is being recognised by assurance practitioners. This has led to the development of the Data Safety Guidelines (DSG) and increasing research in this area [144]. The DSG working group note that an argument may be made that software is simply data, and that the DSG document is therefore unnecessary. However, they note:

> *"... data and software emphasise different facets of risk and they are susceptible to different mitigation approaches; this means there is also a need to adopt a data-focussed perspective ..."*

Similarly, model aspects of a BNS emphasise different facets of risk to both data and software aspects, and likewise model aspects may be tackled with alternative mitigation strategies. The Data Viewpoint aimed to address data-centric aspects of a BNS and is in some respects conceptually convergent with aspects of the DSG document [144]. In contrast the model viewpoint stresses the need to explicitly consider the models used within a BNS as standalone artefacts with properties and behaviours of their own.

Indeed, the need to develop extended documentation and analysis techniques to address the specific properties of model aspects of AISs has been highlighted in the work of Taylor *et al* on safety-critical ANNs [64]. The Model Viewpoint presented here broadly aligns with the recommendations made in this work. However, it has been designed to target BNS specific aspects that are distinct from ANN systems, and to treat BN models as artefacts central to a system's design, as opposed to a consideration that is ancillary to the 'traditional' software considerations.

The views defined in Table 3.3 reflect this approach. Each of the views has been defined in a manner that ensures the generality of the view to both local and global models. Concretely, the views have been defined to be equally applicable in terms of addressing the concerns of that view when considering a BN model either at the local RV level, or at a higher BN object model, or indeed at a fully integrated, complete BN model level. This means that any component of a BN model up to and including the BN model itself can be described in terms of these views. This property extends to the RM-BNS, as will be discussed shortly.

Table 3.3: The views and objects associated with the Model Viewpoint.

Viewpoint	**ID**	**View**	**Description**	**Object/s**
Model	MV-1	Structure	Concerned with the structure (local and/or global) of a model.	Model
Model	MV-2	Parameterisation	Concerned with the properties (e.g. confidence, quality etc.) of all parameters and hyperparameters used in a model.	Model
Model	MV-3	Definition	Concerned with aspects of the model associated with the qualitative aspects of representation, including the context of the model and the definition of model components.	Model
Model	MV-4	Dynamics	Concerned with the high-level properties and dynamics of a model.	Model

For example, the Structure View (MV-1) addresses those concerns associated with the structure of a model. In the context of RVs, this view is aimed at addressing the local structure of the RV: the properties of the distribution used to represent it. The view is also applicable to the global BN model. At this level, the view is similarly aimed at addressing the concerns associated with distribution of the model, but in this context, it is considering the higher-level interactions between all RVs within the model. The Parameterisation View (MV-2) complements MV-1 by addressing the properties of all parameters in a given model. Once again, it is applicable to local and global considerations. In the former case, the view addresses the need to establish statistical confidence and bounds on the parameterisation of a variable, while in the latter case it addresses the performance implications of a given parameterisation.

In contrast, the Definition View (MV-3) is defined to target concerns associated with establishing the *scope* of individual variables and full BN models, and to ensure precisely *what* a given BN model (or RV) represents and *when* it represents it. It is aimed at ensuring that a given model represents what it is intended to, and under what conditions this representation remains valid. Finally, the Dynamics View (MV-4) aims to address higher-level concerns associated with the evaluation and testing of BN models, and is concerned with all activities related to these aspects.

In all cases, these system aspects can be considered to be properties related to each individual BN within the system. Consequently, each of these views can be described in abstract as some particular property of a given model. The model object is therefore used to represent these aspects. As before, this will be developed more fully in subsequent sections. Finally, it is also important to note that attempting to ensure the generality of these views has been partially motivated by the need to accommodate novel BN architectures. These architectures may include RVs that are themselves decision trees, ANNs or any other model that satisfies the constraints of the BN framework. Consequently, the viewpoint and views have been defined without any inherent assumptions about the topology or variant of the BN (or RV) selected.

3.3.3 Other Viewpoints

A complete enumeration of all views is beyond the scope of this chapter. Instead, a subset of views has been selected for discussion. A full tabulated set of all viewpoints and their associated views and objects can be found in Appendix A. Table 3.4 shows the

four selected views. As before, each of these views is targeted at drawing attention to BNS-specific considerations: there may be parallel or analogous considerations in other software systems, but these are not the focus of these views.

The first view in table 3.4 is the Optimisation View (CV-1) from the Computational Viewpoint. This view is aimed at addressing concerns related to all learning capabilities a system may have, both in off-line and on-line (adaptive) contexts. It targets the properties and mathematical behaviours of all learning algorithms that may be utilised by a system. For example, this view addresses the need to consider the robustness of algorithms to noise in data, whether the algorithm is biased (statistically speaking), and what other limitations the algorithms may have in terms of providing an accurate estimate of a given BN model's structure or parameterisation. This latter consideration of a learning algorithm's limitations refers particularly to circumstances in which an algorithm may converge poorly to an estimate of a BN model's properties; this may occur if certain algorithms encounter underlying properties of a BN model or available training data.

Next, the Infrastructure View (TV-1) of the Technology Viewpoint addresses a fundamental challenge associated with the development of a BNS (and other AISs): the need to develop and maintain a tailored set of skills, tools and physical infrastructure in order to subsequently develop a BNS. For many BNS applications, this may include the development of new technology test-beds, new data gathering processes and infrastructure and training of developers, as well as operational and maintenance staff. In terms of the engineering field more generally, the development of a large-scale, complex BNS may require a similar degree of additional training and infrastructure as was necessary for the transition towards software-intensive systems that the field has experienced in the past. The Infrastructure View is therefore focussed on addressing concerns associated with establishing that the organisational and technological prerequisites are in place and are adequate for the BNS being developed. This view then underpins considerations associated with other viewpoints, particularly the Model and Data Viewpoints.

Table 3.4: A subset of views from taken from the Computational, Technology, Operational and Implementation viewpoints.

Viewpoint	**ID**	**View**	**Description**	**Object/s**
Computational	CV-1	Optimisation	This view is concerned with all aspects related to the optimisation (i.e. learning) algorithms utilised in both the development and deployment of a BNS.	Computation
Technology	TV-1	Infrastructure	This view addresses considerations related to the selection, development and use of skills, tools, processes, physical infrastructure and resources necessary for the deployment of a BNS.	Framework
Operational	OV-3	Evolution	This view targets concerns associated with any continuous changes that may be present in the target domain, both in terms of engineered changes to that domain, and ambient changes, perhaps due to operational change, natural degradation due to age, or location.	Process, Scenario
Implementation	IV-4	Integration	Concerned with those activities necessary for the transfer of a BNS from a simulated test-bed into a 'live' environment, and to ensure data passed to it is in alignment with each model's expectation.	Activity

The third view in Table 3.4, the Evolution View (OV-3) of the Operational Viewpoint is aimed at addressing activities related to ensuring the continued validity of the system in continuously changing operational environments. This includes changes arising both as a result of engineered changes (e.g. platform upgrades, configuration changes etc.), 'entropic' changes occurring as a consequence of continued use (i.e. age, or other similar performance degradations or fluctuations), and changes occurring as a result of new operational environments (e.g. new climate, new operational theatre/context etc.).

Consider the motivating example of this thesis: a BNS providing prognostic health monitoring support for the mission planning functions aboard an Unmanned Aerial System (UAS). The BNS will be designed to model the physical characteristics of the UAS - in particular the UAS's components and their statistical failure profiles. If these components or their failure profiles change for any reason, and these changes are not monitored and captured by the BNS, then the BNS's 'understanding' of the characteristics of the UAS will begin to diverge from the actual characteristics of the UAS. This would be a practical example of 'distributional drift'. The Evolution View is therefore focussed on addressing how existing change management systems and other related operational activities may be augmented to target and mitigate errors emerging from BNS-specific properties such as this. Note that it is not addressing how a *model* may be designed to ensure robustness to these changes, but rather the operational *processes* that must be developed and implemented to mitigate these behaviours.

Finally, the Integration View (IV-4) of the Implementation Viewpoint targets those concerns associated with moving a BNS from a test-bed or idealised development environment into a production system and on to an operational environment. This view is focussed on drawing attention to the need to ensure assumptions made during development are valid throughout integration and deployment of the system, and for managing the integration of all future BNS upgrades aboard target platforms. Like OV-3, this view is essentially focussed on addressing how existing techniques for managing the migration of a software system from a development environment into a deployment environment can be expanded to include considerations specific to a BNS.

Using the same UAS example, this could be ensuring that the operational environment the system is migrated to aligns exactly with the development environment. This may include considerations related to the similarity of the ambient operational environment (again, including climate, operational context etc.). It will include ensuring that the

target system is sufficiently similar to development test-beds that the model is applicable to that system. This will require a process for ensuring a BNS is calibrated for the system it is deployed onto; this will differ between vehicles and operational environments. In some respects, this view is concerned with tackling concerns related to the 'overfitting' of a BNS to any development test-beds.

3.3.4 Justification of RM-BNS Viewpoints

The primary concern when developing these viewpoints was to provide a comprehensive, holistic perspective on the important technical issues facing the development of a mission-critical BNS. Each of the viewpoints was derived from an extensive search of existing literature on the development and deployment of AI-based systems and BNSs. This search was made with a particular focus on identifying contributions in literature that were derived through direct experience of deploying and operating an AI-based system, and other complex and/or novel software-intensive systems.

Consequently, many of the concerns captured by the viewpoints have previously been discussed within the AI or safety-critical systems literature at varying levels of detail. However, at the time of writing, the concerns captured by the viewpoints have not previously been structured and unified in a manner that addresses the needs of assurance and AI practitioners alike, and in particular in a manner that directly addresses the particular concerns associated with the development of a mission-critical BNS.

Practically, the viewpoints aim to standardise and structure these BNS-specific concerns, while also integrating insights and concepts developed over the course of this research project. This includes insights gained from analysing and describing the failure modes of BNSs. For example, the Data, Model and Computation Viewpoints were defined to broadly conform with certain existing strategies for framing and communicating more general AI concepts within literature [2, 54]. However, these viewpoints were defined to specifically address safety and assurance concerns related to these concepts, and – importantly – map these concerns directly to published BNS development practices. Table 3.5 enumerates a list of publications and standards that were used to inform the definition of each of the viewpoints.

These viewpoints were then further refined through the inclusion of assurance concerns arising from the experiences of Jacklin et al that were gained through the development of neural-network based flight control systems at NASA [72, 159, 160]. Insights derived from

Table 3.5: A mapping of RM-BNS viewpoints to a selection of corresponding publications that informed the definition of a given viewpoint.

Viewpoint	**References**
Data	[2, 54, 60, 65, 137, 144]
Model	[2, 3, 16, 54, 65, 141, 158]
Computation	[2, 72, 156]
Technology	[6, 141, 159]
Operational	[72, 158, 159]
Implementation	[72, 156, 159]

this work were adapted to align with the parallel but often distinct technical considerations associated with the development of BNSs. This work also highlighted the need to consider broader system aspects related to system maintenance and the ongoing validation of unconventional AIS components in operational environments [159]. These experience-derived insights, along with work of Przytula et al and other work on the development of other techniques developed to describe and model complex, novel systems motivated the introduction of the Operational, Technology and Implementation viewpoints in particular [6, 161].

With this comprehensive survey completed, the identified concerns were then structured to maximise the orthogonality of each of the viewpoints with respect to each other viewpoint. This was driven by a desire to ensure a separation of key concerns, and to thereby minimise the coupling of these concerns across viewpoints wherever possible. To achieve this, concerns were identified and grouped according whether system aspects related to a particular concern could independently produce system-level failure modes in a BNS. Practically, this should allow AI and assurance practitioners to reason and communicate more effectively about the design of a particular BNS. Of course, no system aspects exist in true isolation, and the implications of this fact will be dealt with in more detail later in this chapter.

Ultimately, the viewpoints and their respective views represent a distillation of an extensive search of existing literature across multiple pertinent fields of research on the development of safety-critical systems; novel software-intensive systems; AI-driven systems and BNSs themselves. This distilled information was then systematically structured according to standard engineering best practices, cross-checked against published experiences

of developers producing mission-critical AISs and then presented in a format designed to be transparent and useful to all relevant BNS stakeholders.

3.3.5 Relationship to Existing Safety Standards

The software safety principles that underpin the Ministry of Defence's (MOD) Defence Standard (DEFSTAN) 00-055 highlight the particular issues the viewpoints are aimed at addressing [162].

Principle 1: Software Safety Requirements shall be defined to address the software contribution to system hazards.

Principle 2: The intent of the software safety requirements shall be maintained throughout the requirements decomposition.

Principle 3: Software safety requirements shall be satisfied.

Principle 4: Hazardous behaviour of the software shall be identified and mitigated.

Principle 4+1: The confidence established in addressing the software safety principles shall be commensurate to the contribution of the software to risk.

For the safety assurance of BNSs, these principles do not change. However, the focus of the work required to address them does. For example, in guidance such as DO-178C, there is emphasis on the development of High Level Requirements (HLRs) and the decomposition of these HLRs into Low Level Requirements (LLRs) [69]. This decomposition and refinement of requirements is captured in Principle 2. For AISs based on statistical learning approaches (such as BNSs), the refinement process may proceed in a highly iterative and uncertain manner [59, 64, 86]. Moreover, the refinement process will be coupled to the behaviours the model learns and contains. In the case of BNSs for example, the functional behaviour of the system is often dependent on factors that may be difficult or impossible to specify *a priori* [81, 121]. Examples of common difficulties associated with these factors include the requirements for the necessary quantity and quality of data, aspects of the model's structure and associated modelling choices, and by extension details pertaining to relevant optimisation and reasoning algorithms [64, 72, 163].

Importantly, the interactions between these aspects can introduce sensitive dependencies in the system. These properties contribute to the emergence of the highly iterative

and relatively ad-hoc development cycles commonly used in the development of BNSs. In some cases, they may be inherent manifestations of the aleatoric and epistemological uncertainty associated with the target problem or with using statistical learning methods.

These observations also highlight additional considerations associated with Principle 4. The functional behaviour of BNSs is often determined by a complex combination of factors that are outside the scope of considerations common to conventional software systems. In particular, the functional behaviour of the system is often strongly coupled to system aspects encompassed by the proposed Model and Data viewpoints. As discussed, the Model viewpoint aims to highlight issues surrounding what a given system has learned. The system may perform well on available test data but may exhibit dramatically different - and potentially hazardous - behaviours upon deployment. This may be because available test data was not sufficiently representative of a real-world environment. In a simplistic sense, it may also be because the trained model represents a problem that is different from the intended target problem.

A frequently cited (possibly apocryphal) example of both of these issues describes the case of a US project to develop an ANN-based system for the automatic identification of camouflaged armoured vehicles [164]. The development team achieved good performance in their test environment, but poor performance in subsequent third-party testing. Analysis revealed that rather than learn to identify armoured vehicles, their system had learned to identify the weather conditions in the photographs used to train it. This occurred because the development team had used photographs in which camouflaged vehicles happened to be taken on cloudy days, and un-camouflaged vehicles happened to be taken on clear days. Importantly, the trained ANN model can be described as being consistent with the data-set: the system had learned a model from the data consistent with the classification objective given to it, but this model was not representative of the intended problem. There is also a broader problem of establishing whether – except in the most trivial circumstances – a specific model/modelling framework can be sufficiently representative of a target domain. In the case of BNs, the specific variant and structure of a BN model will provide a hard constraint on the capabilities of the model and therefore any system/s utilising that model.

Consequently, code coverage-focussed testing such as that advocated by DO-178C is unlikely to identify hazardous behaviours associated with errors of this kind. The viewpoints aim to support the identification and mitigation of hazardous behaviours arising

from this type of error (and others specific to BNSs). They also indicate the need for additional data- and model-focussed coverage activities. Without these targeted analyses, establishing a meaningful assessment of the contribution of the system to risk may be challenging. In this way, the viewpoints also aim to address considerations related to Principle 4+1. Concretely, the viewpoints address the 4+1 principles as follows:

Principle 1: The viewpoints expose and address BNS-specific aspects that may contribute to system aspects – including those beyond the scope of conventional systems.

Principle 2: The viewpoints enable the comprehensive description of system aspects using a structured framework that enables relationships between system aspects to be explored. This will be more rigorously addressed in the following sections.

Principle 3: Software safety requirements shall be satisfied. The satisfaction of software safety requirements is predicated on the generation of requirements that can be tested. The viewpoints represent a first step towards guiding assurance practitioners towards analysis and testing techniques that can be used to satisfy BNS software safety requirements.

Principle 4: By supporting the description of BNSs and drawing attention to unconventional system aspects, the viewpoints can be used to drive assurance efforts towards more expansive analysis and testing activities that address BNS-specific system behaviours.

Principle 4+1: The viewpoints expose the full range of safety-related considerations for BNSs. They therefore provide a first step towards identifying which system aspects contribute to system risk. This is itself the first step towards establishing *to what degree* a system aspect contributes to hazardous behaviours. By extension, they can be used to provide a foundation for the discussion of the *sufficiency* of assurance efforts targeting system aspects addressed by a given viewpoint in the context of the contribution of that aspect to system risk. This will be addressed more comprehensively in subsequent chapters.

Ultimately, current safety standards and guidelines do not provide adequate mechanisms for addressing these AI-specific concerns. In the case of IEC-61508, this is because systems with AI-based aspects are actively not recommended [94]. In other cases, language and guidance is included that covers some of the aspects outlined in the viewpoints, but this guidance is invariably new and limited. DO-178C falls into this latter category

with its inclusion of guidance for Data Parameter Items (DPIs). This guidance focusses on data artefacts physically encoded in the software system. For many BNSs, the bulk of data used to train (configure) the system is not physically present in the implemented software system. Therefore, the contribution of this data to system risk may not be explicitly addressed.

More recent work has acknowledged the increasingly influential role of data in modern software systems, and the corresponding shortfalls of existing standards. An example of this is the aforementioned Data Safety Guidance (DSG) [144]. This guidance aligns conceptually with some of the aspects outlined by the proposed Data Viewpoint. However, some of the technology-specific considerations associated with BNSs are naturally beyond the scope of the document, particularly those associated with the interactions between data acquisition, processing and management choices and the learning and reasoning behaviours of implemented BNSs.

Finally, no standards currently tackle concerns associated with the modelling frameworks and activities that produce the models that drive these systems. Work by NASA on the use of ANNs for adaptive flight control systems identified the need to consider additional, independent verification and validation activities in parallel with conventional software assurance activities. This need is reflected in the AI domain generally, where it is common practice to evaluate software and model aspects independently, and is directly relevant to BNs. The Model viewpoint is therefore aimed at addressing those concerns directly associated with the development and deployment of BNSs that otherwise fall outside of the scope of existing software- and data-assurance guidance.

3.4 Reference Model for Bayesian Network-based Systems (RM-BNS)

The viewpoints and their constituent views provide a basis for the structured description of all key aspects of BNSs. They aim to capture the important conceptual distinctions specific to BNSs (and in some cases to AISs generally). However, as alluded to in previous sections, there is an overlap between the spheres of interest of each of the viewpoints, and by extension their views. The interactions between many aspects that are addressed independently by distinct viewpoints may lead to emergent behaviours – some of which may be hazardous. Consequently, while it is possible to address the concerns of each of

the viewpoints individually, it is essential that these interactions are made explicit and understood by all relevant system stakeholders. To this end, this section introduces a reference model for BNSs that aims to achieve this. The reference model builds upon the language defined by the viewpoints, their views and corresponding objects.

3.4.1 Definition of Objects

The objects associated with viewpoints were defined in accordance with the guidance provided by the RM-ODP and IEEE 1471-2000 documents. This is defined as:

> *"An object is an abstract representation of an entity in the real world. It contains information and offers services. A system is composed of interacting objects. Each viewpoint defines its own objects, their relationships and interactions."*

In accordance with this definition, in the Data Viewpoint the objects are artefacts and the processes that produce them. In the Model Viewpoint, the objects are abstract mathematical models that encapsulate both local and global aspects of a BN model. The objects associated with the Computational Viewpoint are abstract computations: a generalisation of any optimisation or inference algorithms that may be used in the system. In the Technology Viewpoint the objects are Frameworks and Infrastructure – abstractions of developmental approaches and infrastructure/tools selected for the BNS lifecycle. Objects in the Operational Viewpoint are Scenarios, Capabilities and Environments related to the deployment of a BNS. Finally, the objects associated with the Implementation Viewpoint are abstract Functions and Resources that characterise a BNS.

3.4.2 Capturing Interactions in BN-based Systems

Many modern applications of BNs utilise a number of distinct development approaches, model architectures, training data types and learning and reasoning algorithms within a single system. The interactions between design decisions and implementation details associated with these aspects can be a source of error in their own right and can occur as a consequence of errors directly associated with a given interaction or as a product of sensitive dependencies between system aspects. An approach to capturing these interactions can provide a degree of transparency to the system and can be used to support subsequent analysis of a system's architecture.

As with the modelling of any complex system, capturing the complexity of these BNSs requires an approach that maintains an acceptable level of abstraction. This must in turn facilitate a useful level of generality. Practically, a suitable modelling approach must be able to capture the diverse and complex architectural properties of BNs without itself becoming too complex. A number of reference models and reference architectures have been proposed to tackle precisely the problem outlined here - that of providing a practically useful, but sufficiently general model of complex, unconventional systems that reflect the interactions between aspects of these systems [156].

The solution proposed here is based on the concepts outlined in existing research on the development of reference models for complex systems [156]. The Reference Model for Bayesian Network-based Systems (RM-BNS) builds on these concepts and provides a structured, comprehensive approach to modelling BNSs. The building blocks of this model are the objects introduced previously. These objects directly connect the RM-BNS to the defined BNS viewpoints. Consequently, this link provides the language for the description of the concerns associated with objects in a model. A full enumeration of objects is beyond the scope of this chapter, but is included within the complete set of viewpoints and views included in Appendix A.

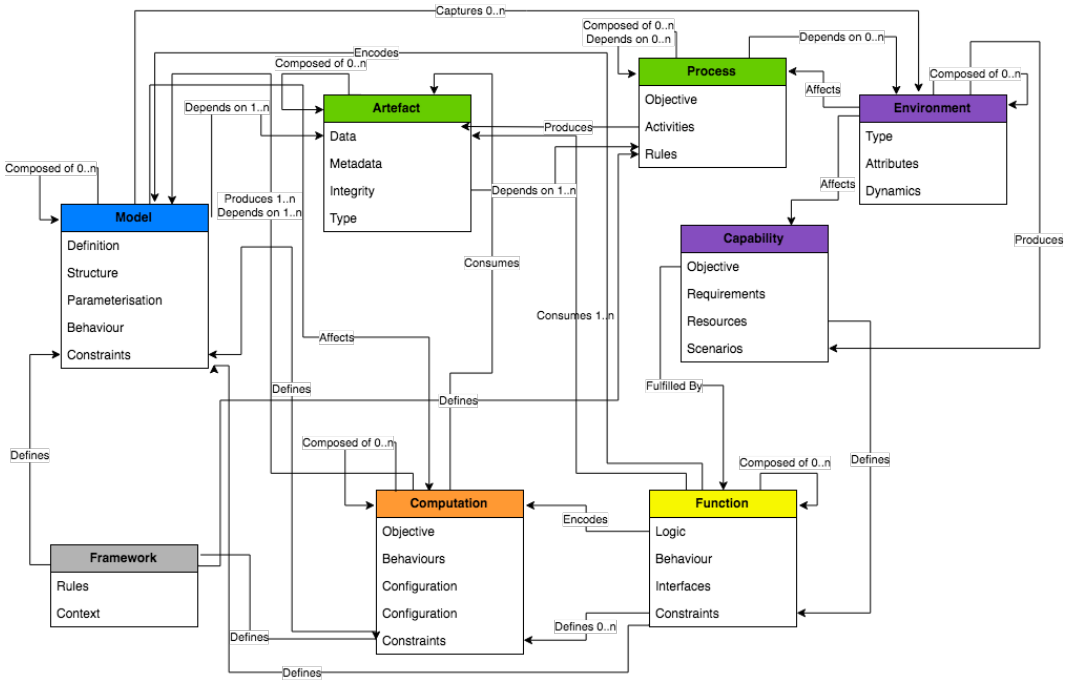


Figure 3.4: The RM-BNS Reference Model.

The proposed reference model is shown in Figure 3.4.[6] The reference model is intended to clearly and explicitly represent the key interactions between objects in the model, and consequently between views both within and across individual viewpoints.[7] For example, the reference model can be used to make explicit the interactions between learning and reasoning algorithms (represented in the abstract as the Computation object), training and validation data (represented by the Data Artefact object) and each of the BN models (represented by the Model object) within a given BNS. Similarly, the interactions between Data Artefacts and Models with conventional software aspects (captured by the Function object) are made explicit. These interactions are only implicit in the viewpoints, but are intended to be transparent and intelligible to safety practitioners and BN developers alike. The utilisation of the RM-BNS for the modelling of an example BNS will be demonstrated in the case study at the end of this chapter.

3.4.3 Role of RM-BNS

The role of the reference model is envisioned as a tool to support the development of a structured, comprehensive description of a given BNS. It is *not* intended to act as a formal model of a system, and by extension has not been designed to support formal analysis of a BNS's properties. Its primary function is to support the development of a transparent, well understood BNS architecture, and to focus attention on system aspects that may otherwise be overlooked.

Consequently, this chapter has been focussed on providing an ontology for the description of BNSs in the form of a set of viewpoints and the accompanying reference model that provide a clear, comprehensive and structured description of key system aspects either unique to, or disproportionately important to, BNSs that can be used effectively by safety practitioners and other system stakeholders with limited exposure to the technical aspects of a BNS.

Furthermore, this ontology has been designed to support extensibility. This notion of the extensibility of the ontology specifically refers to the intent to provide a reference model

[6] This image is included alongside the rest of the core components of the RM-BNS framework in Appendix A.

[7] Note that for 'undeveloped' parts of a RM-BNS instantiation, a diamond is used to represent aspects of a model that could have been developed further but were not. This will be demonstrated with examples in subsequent chapters.

that is flexible, and can be modified to address any application-specific considerations that may fall outside of the scope of the ontology as defined. This may occur in circumstances in which an AIS utilises BN-based *components* in a hybridised architecture. It may also occur in the event a BNS is developed as part of a platform using other technologies outside of the BN or AI domains that may influence a BNS' behaviour in a novel way. This could include novel sensors or other hardware technologies. Finally, it is likely that many BNS applications may not need to consider all aspects of all presented viewpoints. In these circumstances, it is expected that developers simply omit views where necessary, though this should be explicitly justified.

3.5 Generic Verification and Validation Objectives

The previous sections have outlined an approach to describing and modelling BNSs in a comprehensive, structured format. The shortcomings of existing standards have also been discussed. An approach to the assurance of a BNS must overcome these shortcomings - primarily by directly addressing the BN-specific assurance considerations discussed throughout this chapter, and by mitigating weaknesses in existing safety standards. This section therefore introduces an approach to using the viewpoints and reference model to generate a set of comprehensive verification and validations objectives for BNSs that aim to achieve this.

3.5.1 Deriving Objectives

The reference model has been designed to provide an ontology that can be used to develop a general description of all aspects of a BNS. Each of the objects within the reference model can be used to capture an abstract representation of a given system's composition or development processes. Each of these objects can be associated with design choices, and by extension these design choices can be associated with the introduction of errors into the BNS. Where appropriate, this fact can be used to generate a set of error modes for each respective object. The evaluation and enumeration of mitigation strategies for these error modes can then be used to generate a set of objectives.

When this approach is applied to the reference model shown in Figure 3.4, a set of generic verification and validation objectives for BNSs can be generated. These objectives provide a high-level insight into the composition and engineering challenges associated with

the development of a BNS, while also providing an abstract set of objectives that can be used to guide early-stage requirements engineering, or other aspects of safety management.

Furthermore, by utilising the RM-BNS as the basis of the generic objectives, it is then possible to generate system-specific objectives. This can be achieved by producing an instantiated instance of the reference model that maps onto a target BNS. The instantiated model can then be used to derive a set of objectives for the given BNS that capture the properties of the architecture and processes used to develop the BNS. The fact that the reference model provides a comprehensive basis for describing a BNS can be utilised to generate a set of objectives that provide coverage of all key system-specific properties. By extension, instantiations of the reference model - and therefore the system-specific objectives - can similarly facilitate a comprehensive set of objectives that target BN-specific assurance considerations that are directly relevant to the system's architecture and development strategies.

Furthermore, the ability of the reference model to capture interactions between objects also supports an understanding of how design decisions and verification and validation activities associated with individual objectives may overlap. The process for the derivation of system-specific objectives is explored in more detail in Chapter 6. The scope of this chapter once again prevents a full enumeration of verification and validation objectives for BNSs. Instead, an overview of a number of the objectives associated with the Data and Model Viewpoints is provided, alongside a subset of objectives sampled from the remaining viewpoints.

3.5.2 Data Viewpoint Objectives

The first objective (DV-3.3) is concerned with addressing safety-related assumptions associated with the acquisition of data for use in the BNS. A particular example of an assumption that must be made explicit is that any data gathering activities carried out by different operators, at varying geographical locations, or on distinct platforms are sufficiently comparable that data taken from these various sources can be combined effectively and transparently into a single fault database or dataset. In the context of the motivating example of this thesis, this may involve the acquisition of data from many distinct aircraft operators, or geographically and environmentally distinct sites. Each operator or site may have different internal processes or standards for the acquisition or handling of operational data prior to that data being acquired for use in a BNS. The quality, integrity

Table 3.6: A subset of generic verification and validation objectives for the Data Viewpoint.

Viewpoint	**View**	**Object**	**Objective**	**ID**
Data	Acquisition	Process	Establish and justify any assumptions made during data acquisition [Process].	DV-3.3
Data	Processing	Process	Establish and justify the necessity of the data transformation [Process].	DV-2.2
Data	Artefact	Artefact	Establish and justify the integrity of the [Data Artefact].	DV-1.4

and underlying statistical properties of data acquired from distinct operators may vary; aircraft operators in different operational theatres or in different geographical regions may require a more frequent turnaround of aircraft components (which may indicate an increase incidence of component failures), or deviate from normal procedures due to operational pressures.

In a medical diagnosis context, further assumptions may be that the location at which data is gathered - a particular hospital or district - is representative of the population that the system will be used to treat. For example, if the patient demographics at the location at which acquisition processes are performed are skewed (perhaps in terms of socio-economic status, ethnicity or age, as examples), the prevalence of different diseases and patient outcomes will be expected. As repeatedly discussed in this chapter, a BNS will learn to represent the underlying distributions in the data it is trained on. A BN can be designed to be robust to biases such as this - but only if model developers are aware of the assumptions underpinning the data that has been acquired.

The second objective is aimed at addressing the need to establish and justify all transformations applied to data artefacts. For example, it is common within the AI domain to apply a range of scaling and decomposition transformation to data prior to that data being used to train a system. Three common reasons for transforming data in this way are: to ensure acquired data conform to a target distribution, to reduce the dimensionality of the data, and to ensure a degree of homogeneity (in terms of the range of values elements within a data artefact may take) in the data artefact. Each of these motivations can improve the overall performance and robustness of the resulting model. They can also

improve the ability of optimisation algorithms to converge to locally optimal solutions. However, erroneous or inappropriate application of certain transformations may have the opposite effect. A further transformation that may be applied 'silently' is the discretisation or rounding of data. These processes may result in the loss of information in the data, and may once again alter the characteristics of the completed BNS. This objective is therefore aimed directly at ensuring that each transformation used during the development of a BNS is adequately justified and is materially necessary for the resulting BNS.

The final objective presented here addresses considerations associated with the properties of data artefacts used directly by a BNS. It is particularly interested in the evaluation of data artefacts used in training BN models (or more generally in the definition of any BN model) in a BNS. The satisfaction of this objective is dependent on the satisfaction of other objectives, both within the Data viewpoint objectives and in other viewpoints. In this case, the objective is addressing the integrity of each data artefact used to train or define a BN model. The definition of data integrity in this context aligns with the DSG definition of data integrity as [144]:

> *"The [artefact] is correct, true and unaltered."*

From a BNS perspective, satisfaction of this objective indicates that artefacts used to subsequently train BN models have not been manipulated and are representative of the target domain. The manipulation of data is a particular assurance concern for AISs: injection of synthetic or falsified data into a data artefact used to train an AIS can be used to influence the functional behaviour of the completed system. Existing examples of attacks of this kind include making small changes to images (both with editing or 'real world' changes) that can render AISs with AI-based computer vision aspects 'blind' to stop signs or certain objects [133]. From a safety perspective on the motivating example of this thesis, this is concerning for diagnostic systems as the manipulation of data either used to train the system, or ingested by the system after deployment may introduce latent errors in a model that may 'blind' the diagnostic system to some system faults. The elimination of errors arising from these considerations is the primary target of this objective.

3.5.3 Model Viewpoint Objectives

The first objective in Table 3.7 (MV-1.1) is intended to address the need to establish and justify the selected approach to representing a given model (again, either local structure

Table 3.7: A subset of generic verification and validation objectives for the Model Viewpoint.

Viewpoint	**View**	**Object**	**Objective**	**ID**
Model	Structure	Model	Establish and justify the basis for using the [structural variant] for the [Model].	MV-1.1
Model	Param.	Model	Establish and justify the accuracy of the parameterisation of [Model].	MV-2.2
Model	Dynamics	Model	Establish and justify confidence in the dynamics of the [Model].	MV-4.5

of an RV, or global structure of a BN) with a given structure. At the local structure level, this will focus on the RV variant selected to be included in the model. For example, this will require establishing which local structure best represents the target variable the structure is modelling. This will involve establishing whether the variable is best represented as a discrete distribution (e.g. Bernoulli, Poisson etc.), or as some form of continuous distribution (e.g. Gaussian, Weibull etc.), and why this structure is adequate for the task of modelling the target variable.

For example, consider an RV representing a sensor measurement. The statistical profile of the quantity being measured may be approximately normally distributed, and therefore a developer may select a conditional normal distribution as the local structure of that RV. However, the intent of this objective is to establish that this structure is optimal on the one hand (i.e. to establish the structure variant), and to justify that the structure is capable of capturing the required behaviours. This latter consideration may result in the selection of a local structure that in an absolute sense is not the optimal structure in one sense (e.g. it represents the data most closely), but instead represents the properties of the data a BN developer requires. This may involve selecting a structure that is a compromise: it may not be the best approximation to the process being modelled, but may produce the desired performance (or other property).

These local structure considerations also extend to global structure considerations. As outlined previously, the selection of local structure variants can have implications for the global structure of a model. For example, hybrid BN models with continuous and discrete

RVs are often severely constrained: a local structure captured by a continuous conditional distribution may not have discrete child RVs. This is one example where attempting to achieve absolute fidelity to the distributions in training data may produce a model with reduced performance on some tasks. Beyond these structural constraints, this objective is also aimed at addressing considerations associated with the selection of dynamic models (where appropriate), and whether the use of directed models is appropriate given the target domain being modelled. Ultimately, the variant of the model structure selected will influence what a BN model (or RV) can represent (i.e. 'know'), and when it can represent it. Failure to rigorously justify the selection of a given structure variant may result in several distinct error modes that may seriously impact functional behaviours of a BNS.

The second objective builds conceptually upon the objectives associated with the structure view. This objective is aimed at establishing the accuracy of the parameterisation of the model. The aim of this objective is to ensure a concrete, explicit measure of the statistical confidence of a given parameter (wherever possible) is provided. This information is intended to be used to support other objectives, including MV-4.5. The accuracy of the parameterisation refers primarily to the need to establish a concrete link between a model's parameters and underlying data artefacts used to estimate those parameters. Provided this link is established, the objective can build upon the objectives of the data viewpoint to establish that the parameters are indeed an accurate estimate of some 'real world' aspect.

The final objective in Table 3.7, objective MV-4.5, is defined to establish confidence in a given model holistically: the model is more than a sum of its parts; it cannot be assured by considering its definition, structure and parameterisation independently. This objective therefore aims to assess the model as a whole. It builds upon other objectives associated with this viewpoint to establish and justify confidence in a model by evaluating the overall behaviour of the model. As with the other viewpoints, this can be considered at two levels of granularity: at a local level, or at a global level. At a local level, the objective seeks to assure the behaviour of a local model across the range of inputs it may encounter - that the model as defined and trained will be robust to anomalies, for example.

Similarly, satisfying this objective at a global level will require that a model is robust to outliers. However, it is also aimed at ensuring the model is robust to some degree of errors in inputs, for example, or to ensure that the model does not enter a circumstance in which its reasoning capabilities are diminished. Ultimately, it is aimed at establishing

and justifying the adequacy of all modelling efforts associated with the objectives in this viewpoint.

3.5.4 Other Viewpoint Objectives

Table 3.8: A subset of generic verification and validation objectives for the Computational, Technology, Operational and Implementation Viewpoints.

Viewpoint	**View**	**Object**	**Objective**	**ID**
Computation	Optimisation	Comp.	Establish and justify the necessity of optimisation [Computation].	CV-1.1
Technology	Infrastructure	Framework	Establish and justify the sufficiency of available tooling [Framework].	TV-1.2
Operational	Evolution	Process	Establish and justify the [Process] for monitoring the validity of the [System].	OV-3.4
Implementation	Allocation	Function	Establish and justify the chosen [Function] allocations.	IV-3.1

The Computation Viewpoint's first optimisation objective (CV-1.1) is focussed on ensuring that every optimisation approach selected is necessary for the given system. At the highest level, this is aimed at addressing the need to ensure that optimisation approaches used within the system are appropriate for the application, and that no simpler or more transparent approaches could be used in place of the selected approach. For example, there are a range of structure learning algorithms. If a model can be represented as a tree-structured BN model, there are structure learning algorithms that have been proven to be mathematical optimal for this application. If another algorithm is used in cases such as this, the justification of the necessity of the chosen approach will rely on defining what specific properties a chosen algorithm has that means a simpler algorithm was not selected. Similar concepts are also applicable to parameter learning approaches.

The second objective in Table 3.8 (TV-1.2) addresses the need to establish the sufficiency of available software tooling that will be used to design and test BN models and the

completed BNS. For example, many commercial software packages for BN development are highly constrained with respect to the available learning and optimisation algorithms available, and the BN variants they support. These constraints may result in hard constraints on the resulting BNS - it may be discovered later in development that a learning algorithm or structure variant may be necessary that is not supported by available tools. These constraints may directly contribute to the introduction of error modes into a completed BNS.

Beyond these considerations, data processing, storage and transfer tools will also be necessary. These may include tools for BN- or AI-specific data pre-processing tasks. There will also need to be robust (software and hardware) tools in place to test a BNS. This will include the development and availability of BNS test-beds. Finally, there is the need to establish confidence in the quality and integrity of the tools themselves. This objective therefore seeks to ensure that the tooling used is of sufficient quality and is sufficiently flexible and robust to meet the demands of the BNS's lifecycle. This is particularly important as existing software safety standards do not provide adequate guidance on the development of tool qualification packs for BNS (and AIS) development. In the case of BNSs, it will be necessary to develop standardised testbeds for this class of system that share conceptual parallels with conventional software systems, but aim to address specific *model* performance characteristics.

A number of projects are in progress with the aim of developing standardised testbeds for the validation of autonomous vehicle models within the commercial domain [165]. These projects have begun to develop standardised model test suites, including 'high-fidelity' simulated operational environments based upon recorded data and constrained 'real world' testing facilities for live testing and system integration activities. Other industries will require similar industry- or application-specific testbeds. These testbeds should focus on the evaluation from at least four perspectives:

1. **Numerical simulation and analysis** - Standard metrics and evaluation techniques should be adopted that address the performance and behaviour of models at a low level. They will need to provide standard performance benchmarks and these approaches are likely to be highly framework-specific. For example, a testbed developed for BNSs is likely to be BNS-specific and would not be usable for ANNs. An example of existing tools that may be developed into more holistic testing frameworks are the DeepXplore and DeepTest tools [113, 166].

2. **Simulated testbed environment** - Standardised test suites with high-resolution simulated environments and scenarios. These are likely to be application-specific, but could be shared across multiple modelling frameworks (i.e. may not be BN-specific). Examples of existing testbeds include testbeds developed by DeepMind to evaluate the AlphaGo and GoZero AISs. Proposed examples within the aerospace and military domains include utilising recorded mission footage and telemetry to evaluate a model's performance in previous activities. These could be 'perturbed' to produce a broader set of test cases.

3. **Constrained testbed environment** - Standardised real-world environments for testing model predictions in the presence of real-world operational noise and uncertainty. A prominent example of this form of testbed is the 'Castle' testbed developed by Waymo for testing autonomous vehicles. This is an ersatz town with extensive mock-up roadways and obstacles that is designed to simulate a small village. This simulated village is used by Waymo's vehicles as 'structured testing' environment that must be 'passed' before deployment [165]. These are likely to be industry-specific. These too can be shared across modelling frameworks.

4. **Test deployment** - A final set of standardised AIS-specific 'shakedown' tests to confirm assumptions and observations made in earlier developmental phases.

Next, the objective associated with the Operational Viewpoint's Evolution view (OV-3.4) addresses the need to establish and justify all operational processes that will be used to continuously monitor the validity of a BNS after deployment. This objective is intended to ensure that attention is paid to what processes need to be developed and implemented by operators to ensure the continued validity of a BNS in whatever operational environment it is deployed - and by extension to maintain confidence in the safety of a system over its

Finally, the last objective in Table 3.8 is aimed at justifying conventional software or hardware function allocation decisions. For clarity: this is with respect to any software or hardware deployed as part of the completed BNS, and not with respect to supporting tools or development software. The objective aims to ensure that the software architecture itself is designed to mitigate potential error modes associated with the underlying models the system uses. For example, the software architecture may be designed to deprecate the use of BN model outputs for decision making in certain scenarios, or to use supporting information external to a model wherever possible. The objective essentially aligns with

conventional safety-focussed function allocation considerations, but is particularly focussed on establishing the effect of a system's architecture with respect to BN models.

3.6 Justification of the Completeness of the RM-BNS Objectives

The RM-BNS objectives have been designed to provide a complete set of verification and validation objectives for BNSs and should be of practical utility to developers working towards the assurance of a mission critical BNS. The completeness of the generic verification and validation objectives can be justified from the following perspectives:

1. By building upon the comprehensive nature of the viewpoints introduced previously and by systematically cross-checking against existing work in other areas of AI, the objectives can be regarded as being complete by construction;

2. The objectives were generated through the systematic consideration of all BNS error modes and design concerns that were identified over the course of this research, both through experimentation and from published literature.

In the case of the first perspective, the generic objectives were derived directly from the structure and content of the RM-BNS viewpoints. The RM-BNS was designed to provide a comprehensive framework for the description of BNSs, and the viewpoints, views and their associated objects and model are the product of a systematic review and the collation of all research on the development and assurance of AISs and BNSs that was found at the time of writing.

As discussed in Section 3.3.4, this search process integrated information from the full breadth of published literature from across work in software engineering, assurance of complex and novel systems and from within the AI and BN domains. As the generic objectives were systematically derived directly from this work, their completeness can be regarded as a by-product of their construction.

Beyond this, a systematic review of all published BN error modes was conducted as outlined in Section 3.2.4, and a summary of these BN-specific error modes was provided in Table 3.1. A detailed review of each of the error modes as described in that table was then conducted, and in all cases at least one of the provided generic objectives was found to address the specific error modes in the given publication.

Furthermore, a number of case studies based on published literature were developed over the course of the work for this thesis (the most comprehensive of which is presented in Chapter 6), and in all explored cases the objectives provided were found to be complete for the given case study. However, there may be cases where a given objective is not applicable. In these cases, the provided set of generic objectives could therefore be considered to be a superset of the relevant objectives for the specific application. Handling cases such as this will be discussed in more detail in subsequent chapters.

Beyond the generic objectives themselves, these case studies also indicated the utility and completeness of the system-specific objectives generated for a particular system. The structure and flexibility of the generic objectives ensured that in all encountered cases the system-specific objectives covered concerns related to the BNS aspects considered. Alternatively stated: no cases were encountered that involved concerns that could not addressed by the derived objectives.

However, while the case studies indicated the effectiveness of the framework for generating derived objectives, it should be noted that the character of the derived objectives is different from their generic counterparts. The completeness of any set of derived objectives is predicated on this distinction: the generation of a set of system-specific objectives requires the participation and input of a practitioner. As with any similar process, this practitioner will be required to exercise their best judgement throughout.

From this perspective, the process of deriving a set of system-specific objectives from the generic objectives could be considered analogous to an extremely strong variant of a guide-word-based generative process. Unlike other such approaches (e.g. HAZOP), the generic objectives stipulate specifically what aspects of a system must be addressed, and how these aspects should be captured within derived objectives [83]. An assurance practitioner will be tightly constrained to consider a BNS in terms of the language and concepts captured explicitly within the generic objectives.

As previously discussed, these constraints have been systematically defined to comprehensively address assurance concerns for BNSs. Consequently, while there is no strict guarantee that a derived set of objectives will be complete, the objectives and process introduced here and the derived objectives that result from them will ensure a level of completeness that would not have been possible through the use of an ad hoc approach. Furthermore, practitioners are encouraged to cross-check their derived objectives against the viewpoints and the concerns described in their definition. This should help ensure

that the focus of a given viewpoint or concern is maintained in any derived objectives. This should further boost the level of completeness attainable for a given set of derived objectives.

3.7 Conclusion

In conclusion, BNSs introduce a number of assurance concerns that emerge as a direct consequence of both their nature and their use-cases; in particular the high degree of dependence of certain functional behaviours on system aspects that can be considered to be software-independent artefacts, as well as the aleatoric and epistemological uncertainty commonly inherent in their use. This chapter has outlined the need to effectively capture the full scope of concerns associated with this class of system. This includes many error modes that are either unique to BNSs or conceptually distinct from analogous error modes found in conventional software systems. This has been one of the lesser contributions of this chapter: to summarise the conceptual distinctions associated with the development and deployment of BNSs for assurance practitioners, and to provide a summary of existing work into analysing and evaluating error modes in BNSs.

Beyond this, the chapter has made two primary contributions. These are:

- Providing a framework for the structured, comprehensive description and modelling of BNSs that specifically targets key assurance concerns associated with BNs. Aspects of this framework have been discussed in terms of their relationship to existing thought in the safety engineering domain, and to existing standards;
- Introducing a set of generic BNS verification and validation objectives for BNSs that aim to mitigate the inadequacies of existing safety standards in handling BNSs, as well as an approach to refining these objectives into system-specific objectives when necessary.

Taken together, the contributions of this chapter aim to provide a step towards developing a strategy for the assurance of BNSs that address the technical and conceptual disparities between this class of system and conventional software systems. The framework presented here (and its associated BNS ontology) can be used (and expanded) to analyse the composition of BNSs and to ensure assurance practitioners adequately address system aspects they may overlook.

Finally, due to the scope of this thesis and the resources available over the course of research, subsequent chapters will focus primarily on system aspects associated with the Model Viewpoint. This has been motivated by several practical factors, as well as a number of salient research considerations.

The two most important motivating factors for this decision were:

1. Of the three viewpoints defined to draw most heavily from concepts in the AI domain (i.e. the Data, Model and Computation Viewpoints), the Model Viewpoint addresses a number of considerations that are likely to be seen as the most unconventional and potentially controversial of the six viewpoints. From this perspective, it was regarded as the most interesting research direction to pursue. This was further reinforced by the fact that early in the course of this research project it became apparent that system aspects that would fall under the Data Viewpoint were increasingly recognised as serious assurance concerns in more general applications [144].

2. Over the course of this research project, the relative absence of real-world data due to both the relative scarcity of such data, and the sensitivity of the data that did exist prevented concrete conclusions and examples pertaining to certain system aspects addressed by the other five viewpoints from being explored in this thesis.

Beyond these two primary considerations, a further practical benefit of focussing on the Model Viewpoint in subsequent chapters is the availability of several well-studied BN models for research use. These provide more concrete examples than may otherwise have been possible.

Chapter 4

Model Criticality Analysis

4.1 Introduction

The previous chapter introduced a structured approach to establishing verification and validation objectives for Bayesian Network (BN) based systems. The objectives aim to improve the coverage of verification and validation activities over aspects of BN-based systems that may be poorly addressed by existing safety guidance. However, the exhaustive analysis, testing and absolute satisfaction of the proposed objectives will not be practicable in many use-cases.

Consequently, the objectives need to be prioritised with respect to the relative contribution of all aspects of a BN-based System (BNS) to overall system hazards. Moreover, the adequacy of efforts associated with the satisfaction of these objectives must be considered from this perspective. This chapter therefore focusses on establishing a means of discussing the adequacy of assurance activities related to the objectives outlined in the RM-BNS Model viewpoint. Specifically, it aims to address the need for a structured, replicable approach to defining the importance of BN models (and their constituent parts) that are used by a safety-critical system to support safety-related software functionality.

To achieve this, the chapter introduces a notion of model criticality, and a means of systematically analysing safety-related BN models. This approach aims to establish the criticality of both individual, independent models in a system, and aspects of the internal structure of a BN model. This revolves around the introduction and utilisation of a combination of safety analyses and automated numerical analysis activities to establish Model Criticality Indices (MCIs) and Variable Criticality Indices (VCIs) for BN models.

This chapter is structured as follows: an overview of modelling and assurance consid-

erations associated with the Model Viewpoint are outlined to highlight the motivation for the establishment of MCIs and VCIs for BN-based systems; a methodology for categorising the criticality of all independent BN models used in a system is introduced; the notion of model criticality is then extended to include variables within the model, with the aim of highlighting structures within a model that influence safety-related model behaviours; the analysis approaches are then applied to a case study; and finally the limitations and possible extensions to the approaches are discussed.

4.2 Motivation

The BN framework is a flexible approach to developing Artificial Intelligence-based Systems (AISs) that supports a number of distinct model architectures and modelling approaches. These model architectures can integrate other Artificial Intelligence (AI) approaches, and can be utilised by a BNS in a number of different ways. For example, it is common for a single BNS to utilise multiple independent BN models to support different software functions: a system may include a fault diagnostic BN model for each subsystem aboard a vehicle, or for each group of related diseases in a medical context. Consequently, a single model may be used to directly support one or more software functions, and each function may be utilised in one or more BN models. As these BNs may ultimately constitute a significant proportion of a BNS's control logic, understanding the relative influence of individual models, their architectures and their constituent components on safety-related software functions becomes vital.

Furthermore, the technical and practical limitations of the assurance activities undertaken in the course of satisfying the RM-BNS Model Viewpoint objectives necessitates an approach that can be used to establish the proportionality of these activities in a systematic, rigorous, targeted manner. There are three principal motivating factors for developing the Model Criticality Analysis approach outlined in this chapter; these are:

- **Traceability** - The need for a systematic approach to establish which models (and model components) in a BNS influence safety-related functions.
- **Influence** - The need to understand *the degree of influence* of BN models (or model components) on safety-related functions.
- **Proportionality** - The need to establish an approach for describing the proportionality of safety-focussed BN analysis techniques.

This section gives a brief overview of the key aspects of each of these motivations in the context of BNSs.

4.2.1 Traceability

Systems utilising AI are commonly criticised within the assurance domain for their 'black box' properties. Black boxes are concerning for assurance practitioners: establishing the traceability of a system's behaviours to individual functions and requirements is a core tenet in all systems engineering domains. While a number of techniques have been proposed within the field of AI to mitigate some of the issues associated with using black box AI approaches, these techniques use language, concepts and ideas that are beyond the scope of existing standards and assurance practices.

In some AI approaches, groups of structures within a model are responsible for individual behaviours – individual behaviours may be distributed across a model and may be (to some degree) emergent properties of the model. This differs from conventional software systems where errors can often be traced directly to logical errors in individual lines of code. As a loose analogy, the current activities of assurance practitioners in the analysis of software systems when applied to AISs are akin to attempting to assure the quality of a human driver by individually evaluating the neurons in a driver's brain for biological defects. In general, this is unlikely to provide any meaningful assurance of the driver's quality.

For AISs, typical function decomposition approaches may begin to break down at the point at which a system's behaviour may become principally determined by the models that drive them. At this 'software-model boundary', conventional approaches to maintaining traceability may become less useful or uninformative. An approach to maintaining traceability must use the language of the AI framework used in the system. In the case of BNs, this will require that functional behaviours are traced to individual models, model structures, or internal structures of Random Variables (RVs). Establishing the role of models (and again, model structures and RVs) in the functional behaviour of a system may require a bottom-up approach (as opposed to top-down function decomposition). The dynamics of a complete model must be understood before these assertions can be made.

Maintaining traceability of functional behaviours to system aspects therefore requires an approach that directly analyses the models in each BN model in a system in order to establish the assurance implications of each model's dynamics. This approach must

be systematic, rigorous and target assurance concerns. There are a number of extant BN analysis techniques that support the evaluation of BN models, but none provide a mechanism for integrating assurance considerations into the analyses. The adaption of existing techniques for the targeted assurance and analysis of a BN model's dynamics, and the use of these techniques to support the traceability of behaviours was therefore a primary motivator for this chapter.

4.2.2 Influence

While the traceability of behaviours was a primary motivation for the development of the work presented in this chapter, simply providing an approach to tracing behaviours to model aspects would not have been sufficient: ultimately, an assurance practitioner must understand the degree of contribution of a system aspect to potentially hazardous behaviours. This is essential if the confidence in a system aspect is to be proportional to the hazard contribution of that aspect. This is captured explicitly in the final principle of the 4+1 software safety principles [162]:

> *"The confidence established in addressing the software safety principles shall be commensurate to the contribution of the software to risk."*

This principle can be considered to include the BN model aspects of a BNS; understanding a BN model's (and its components) contribution to system risk is essential. This requires an understanding of how much individual BN model aspects influence the behaviour of a software system. Unlike the problem of traceability of behaviours, this is not a binary proposition: establishing influence categories for BN model aspects is therefore necessary.

4.2.3 Proportionality

Establishing the influence of a BN model or model component is then a direct precursor to establishing the proportionality of analysis and testing techniques used to assure a BNS. This was the final core motivation for establishing the criticality of BN models and model components. By developing a robust approach to establishing which models and model aspects contribute to which software functions, it becomes possible to provide a targeted, proportional response to model aspects that are of particular concern to assurance practitioners. This is as opposed to treating BN models as black boxes, or otherwise applying a

blanket approach to the rigour applied to the analysis of BN model aspects. These latter approaches may produce either intractably large analysis problems, or otherwise fail to exploit the information contained within a model.

The targeted approach to proportionality as presented here is supported by the fact that the BN models often have inherent semantic value: they directly represent a physical process or entity (e.g. a sensor measurement or vehicle component). This is in contrast to other AI approaches that may be less explicit in what a model has learned or represents. Furthermore, by directing and prioritising model assurance activities towards aspects of a model that have the most influence on safety related functionality, the method can be used to facilitate more efficient, targeted assurance activities. This will be returned to in Chapter 5, where the criticality analysis and RM-BNS are used to support the notion of the sufficiency of assurance efforts in the satisfaction of the verification and validation objectives introduced in the previous chapter.

4.3 Model Criticality

In conventional software systems, software criticality is commonly used to capture the importance of software components to safety-related functions. Software criticality metrics therefore support the prioritisation of assurance activities and ensure proportionality of assurance activities. They also support the traceability of hazardous behaviours to software components. Conceptually, therefore, the notion of criticality aligns closely with the stated requirements for an approach to establishing the traceability, influence and prioritisation of assurance activities for BNSs. However, in this work, the notion of criticality is extended to include BN models and model components. Taken together, this is defined here as 'model criticality':

Model Criticality - A product of the degree of contribution of a model to safety-related functional behaviours and the severity of hazards associated with these behaviours.

As discussed in Chapter 3, BN models can be considered in terms of both local and global properties. Analyses can target either of these properties and evaluate their respective behaviours. The notion of model criticality extends to both local and global model aspects. However, for clarity, the criticality of local models (e.g. RVs) is referred to here as *variable criticality*. A given model's criticality is therefore a combination of both its global criticality (model criticality) and local criticality (variable criticality). Establishing

a model's criticality is therefore based on the definition and assignment of two sets of criticality indices to a given model: Model Criticality Indices (MCIs) and Variable Criticality Indices (VCIs). The former set of indices capture the criticality of a BN model taken as a whole, while the latter address the criticality of the constituent parts within a BN model.

4.3.1 Justification for Proposed Structures

Before continuing, it is worth exploring the motivation for the introduction of the structures proposed in this chapter. The tables and figures in this chapter are aimed at addressing the fact that not everything within a software system's architecture is of uniform criticality (a point recognised by many standards, including IEC-61508), and that this remains true in the case of BNSs. Effectively capturing and managing the variability of criticality and the corresponding level of assurance required for a given degree of criticality across a system is a widely recognised problem within the assurance domain [74, 79, 83]. Concretely, assurance practitioners must establish replicable mechanisms for:

1. The determination and description of the criticality of aspects of a system's architecture.

2. The moderation of confidence in aspects of a system's architecture given its criticality.

Many existing standards attempt to provide such mechanisms. For example, documents such as ARP-4754A and ISO-26262 introduce the concepts of Development Assurance Levels (DALs) and Automotive Safety Integrity Levels (ASILs) respectively [67, 95]. In both cases, DALs and ASILs can be considered proxy mechanisms for capturing and moderating assurance based upon the criticality of a given system or system aspect. Other documents such as MIL-STD-882E more explicitly capture these ideas by introducing Software Control Categories (SCCs) that are used in conjunction with Software Safety Criticality Matrices (SSCMs) to determine the Level of Rigour (LOR) required for the satisfaction of a given objective [167]. Similar constructs can be found across the safety domain [83, 94].

In all cases, the producers of these guidelines and standards are attempting to balance the faithfulness of a chosen representation against the identifiability of that representation. For example, it is clear that in reality the criticality of software components would be more

accurately 'measured' on a continuous scale. A truly faithful representation would therefore dictate that practitioners utilise continuous scales when describing and moderating criticality and assurance in a system. Practically, however, such methods are difficult to define and utilise effectively: identifying measures of criticality is extremely difficult, and actioning assurance activities based upon this information would typically be unfeasible.

Existing standards therefore adopt a compromise by delineating a finite number of levels or categories that are used to render the problem tractable. In these cases, there is an implicit understanding that the given representation (e.g. integrity levels or criticality matrix) is not truly faithful to the problem at hand but that the ability to offer a practicable solution is an acceptable compromise.

These considerations are precisely the motivation and rationale for the development of the structures presented in this chapter. It is essential that a mechanism for capturing the variability of criticality in BN models is provided, as this is a prerequisite for apportioning the effort to objectives associated with the given system aspect. While the precise structures could be debated and represented in alternative ways, the aim here is to provide a practicable, transparent approach for tackling the two key considerations introduced above. As the common aphorism in statistics says: all models are wrong, but some are useful.

Finally, it should be noted that while the application of these techniques to conventional software systems is a well-established approach to describing and delineating criticality of components in a given software system, the application of these ideas to novel systems carries additional risks. In the case of the contributions presented in this chapter, the definitions, concept and relative utility of the structures has been validated against architectures and systems explored over the course of this research, and through their application to a number of case studies.

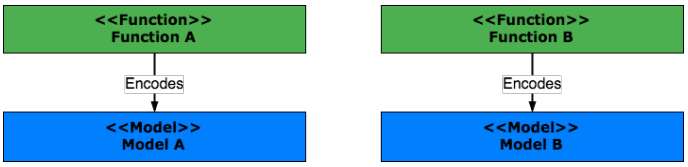
However, until such a time as rigorous testing, refinement and validation in an active development environment has been performed, these structures should be carefully considered in the context of a particular BNS application to ensure their appropriateness for the given problem. Furthermore, whilst it is possible that the exact definitions of the tables proposed here will vary over time as the proposed structures are applied and evaluated, this does not detract from the feasibility of the approach established here, namely that it is possible and potentially useful to define such structures for BNSs.

4.3.2 Authority Categories

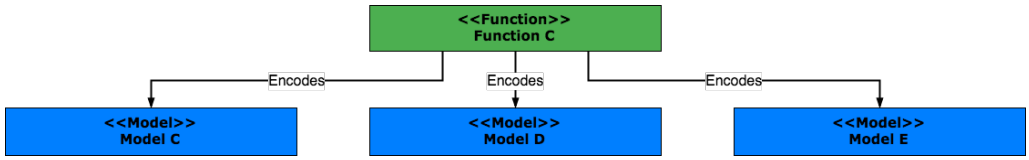
The models within a BNS can be structured and utilised in a number of ways. Common approaches include the development of multiple independent BN models to capture specific aspects of a target domain; a BN model may be used to model and reason about many different system components, for example an aircraft's engine, an autonomous vehicle's mission planning system, or for data fusion and processing for sensor telemetry. A BN model capturing all three could be deployed aboard a single platform. This approach has been popular with commercial and industrial systems [157]. For example, product troubleshooting software from Microsoft and satellite monitoring systems from NASA have both used this approach, in the case of the former by representing individual products or troubleshooting problems with separate BN models, and in the latter by modelling satellite subsystems as independent BN models [6].

As a simple illustration, Figure 4.1a shows how a single software system may use multiple BN models to support software functionality. The system architecture shown in the figure utilises a single inference engine that supports two functions (functions A and B). Each of the functions provides a distinct operational capability to the system. In the event failure to provide these functions has safety implications, each of the software functions becomes safety-related. The contribution of each of the models to the behaviour of a function is then of interest to safety practitioners. In the event the severity of the hazards associated with some failure mode of function A are greater than those associated with function B, the relative interest of assurance practitioners will focus more heavily on analysing function A. If an error in model A may directly produce a high-severity failure mode in function A, this model will then also be of more interest to assurance practitioners - and may therefore be considered to have of greater criticality than model B.

While this approach is common, there are alternative approaches to utilising BNs. One approach is to utilise multiple independently trained (or developed) BN models in place of single models. In this case, a system may reason over the outputs of each individual model - it will use some form of model averaging before the model outputs are used to provide a 'final' output. This approach is conceptually similar to the concept of redundancy in conventional systems. An example ensemble architecture is shown in Figure 4.1b. These approaches are sometimes referred to as *ensemble* modelling approaches [2,168,169]. Ensemble approaches may seem a clear choice for a BNS used in mission-critical contexts. However, the development of ensembles of BN models may not be practically feasible. The



(a) A monolithic architecture.



(b) An ensemble architecture.

Figure 4.1: RM-BNS Fragments of *Ensemble* and *Monolithic* Architectures.

development of multiple models may require large quantities of data for each individual model and may require other resources which may be prohibitively expense to obtain (i.e. time and cost). This is particularly difficult in so-called low data regimes - contexts in which limited data (and/or expert experience) is available.

There is a further consideration when considering the roles of models in BNSs. This is how the output from a model is used to support the provision of a given software function (or set of software functions). For example, in the example shown in Figure 4.1a, both of the models in the figure are used to *directly* support the provision of functions A and B, respectively. This approach would mean that errors in models A or B would directly inform the behaviour of functions A or B. In contrast, if a BNS required user input, or was otherwise used in a purely decision support role that informs operator behaviours, it may be described as *indirectly* informing safety-related system functionality. For completeness, there is also the case in which a BN model is *independent* of safety-related software functions.

To generalise, the models used within a BNS's architecture can be defined either as being used in either *monolithic* or *ensemble* configurations. An example of a monolithic model may be the development of a single BN model to represent a single vehicle, whereas an ensemble may use multiple BN models that each represent the same vehicle. The role of these models can also be defined as either being *executive*, *advisory* or *independent* models. For clarity, these terms are defined here as follows:

Monolithic Models - Models that are used independently of any other model to support the provision of a set of software functions.

Ensemble Models - Models that are used collectively to support the provision of the same set of software functions.

Executive Models - Models that are used directly in the support of the provision of one or more safety-related software functions.

Advisory Models - Models that are used indirectly in the support of the provision of one or more safety-related software functions.

Independent Models - Models that are never used to directly or indirectly support the provision of any safety-related software functions.

Finally, there is the need to consider whether or not the reasoning of a given model occurs in isolation; there may be additional information available to a BNS that can support the output of a given BN or ensemble of BNs that is not derived from a model itself. This may take the form of sensor readings, operator input, or developer-imposed thresholds, rules or other constraints. More generally, there is a need to consider whether or not there exists information external to the model or ensemble of models that can be used to corroborate or validate the model outputs. This can be referred to as a means of external validation, and can be defined as:

Means of External Validation - Any source of data or input external to the model that can be used to corroborate or otherwise validate the output of a model.

These definitions provide the basis of the Model Authority Categories (MACs) for BNSs. The MACs bear conceptual similarities with the Software Control Categories (SCCs) outlined in MIL-STD 882E [167]. In this case, the SCCs are used to categorise the degree of control software components have over safety critical functions. In contrast, a BN model's MAC assignment is intended to reflect the degree of exposure of a safety-related software function to a BN model, and by extension any errors that are *entirely* attributable to the model utilised in the provision of that function.[1] Concretely, a MAC can be defined as follows:

Model Authority Category - The degree of exposure of a safety related function to a BN model.

[1]This refers to any errors that are entirely encompassed by the concerns addressed by the Model Viewpoint of the RM-BNS: errors that can in principle be considered software and platform independent.

The definitions of each of the proposed MACs is enumerated in Table 4.1. The definitions have been designed to be exhaustive and sufficiently general to accommodate various BN architectures.

Table 4.1: Model Authority Categories (MACs)

Cat.	**Name**	**Description**
0	Full Authority	[Monolithic and Ensemble] Executive Models without means of external validation.
1	Partial Authority (I)	[Monolithic] Executive Models with means of external validation.
2	Partial Authority (II)	[Ensemble] Executive Models with means of external validation.
3	Supporting Authority	[Monolithic and Ensemble] Advisory Models with or without means of external validation.
4	No Authority	Independent Models

4.3.3 Justification for Model Authority Categories

Practically, Table 4.1 was defined to represent a comprehensive taxonomy of possible BNS architectures from the perspective of the Model Viewpoint: all identified BNS architectures can be categorised according to this table. Consequently, the table provides a standardised approach for describing the influence of BN models in BNSs on other system aspects. However, as discussed in Section 4.3.2, while designed to be comprehensive, the adoption of a constrained number of categories can obfuscate some technical subtleties.

For example, consider BNSs using BN models in an ensemble configuration. In these cases, it is common for the outputs of a set of models to be weighted in accordance with how 'correct' that model is given some task. In cases in which one model out of a set is dramatically more accurate than the others, such weightings may overwhelmingly favour the predictions of one model out of the set over all others. In circumstances with severely asymmetric weightings such as this, the categorisation would unequivocally fall under the Partial Authority II category in Table 4.1, though their degree of influence would suggest they may be best categorised as Partial Authority I or even Full Authority given the level of influence of a single model over the system.

Ambiguities around the most appropriate classification or level for a given component

are common challenges within the assurance domain [74, 79]. The resolution of these ambiguities is invariably a task delegated by existing standards to the development teams working on a system. This is again true of the contents of Table 4.1: developers should carefully consider the particular characteristics of their system in the light of the contents of the table and should adopt their best judgement in the event any discrepancies or ambiguities are discovered. Indeed, future validation efforts should focus on establishing the utility of this structure and the category definitions it contains in a real-world development environment. These efforts should also attempt to refine the proposed definitions in the light of any ambiguities that are found. For example, the example presented above is generally a sign of a poorly generalised BN architecture, and should occur highly infrequently in operational BNSs. However, if circumstances exist in which 'edge cases' such as this are identified, these should be used to improve the contents of Table 4.1.

4.3.4 Model Criticality Index

The MACs provide a first step towards supporting the maintenance of the traceability of functional behaviours in BNSs by providing a means of assessing and describing the degree of influence of each individual BN model within a system upon any safety-related functionality. However, the MACs do not provide a mechanism for addressing the proportionality of efforts beyond identifying the general role of a BN model in the system. Specifically, they do not integrate any notions of severity into their definitions. The notion of the relative severity of a failure mode is central to the concept of the criticality of system aspects. The Model Criticality Indices (MCI) provide this mechanism. They directly relate the authority of a BN model to the worst-case severity of hazards related to failure modes associated with system functions that may emerge as a consequence of errors within the given model. This can be summarised as:

Model Criticality Index - The degree of authority of a model with respect to the maximum severity of hazards arising from failure modes of functions utilising erroneous outputs from that model.

After assigning a model a MAC, the next step towards defining a model's MCI is to establish the severity of hazards associated with the failure of any functions over which the model exerts a degree of authority. This can be achieved by applying a standard Failure Modes and Effects Analysis (FMEA) to the BNS. A FMEA can provide a robust approach

to tracing failure modes to software functions within a system. The severity associated with each failure mode can then be defined as normal for each function. The definitions of the severity categories are those provided in DO-178C [69]. These are given as:

Catastrophic - Failure may cause a crash. Error or loss of critical function required to safely fly and land aircraft.

Hazardous - Failure has a large negative impact on safety or performance, or reduces the ability of the crew to operate the aircraft due to physical distress or a higher workload, or causes serious or fatal injuries among the passengers.

Major - Failure is significant, but has a lesser impact than a Hazardous failure or significantly increases crew workload.

Minor - Failure is noticeable, but has a lesser impact than a Major failure.

No Effect (Negligible) - Failure has no impact on safety, aircraft operation, or crew workload.

With the severities assigned to failure modes of individual software functions using FMEA, the MCIs for each model in a BNS can be assigned. This can be achieved by using an architectural model of the BNS defined using the RM-BNS framework. A RM-BNS instantiation will directly map individual software functions to any BN models associated with that function. The severity associated with each individual BN model is established by selecting the most severe failure mode amongst each of the functions that that model supports. With this category established, the MCI can be defined using the matrix shown in Figure 4.2.

4.3.5 Application of MACs and MCIs

The utility of this approach can be illustrated using the architecture fragments shown in Figure 4.1. Consider the case in which Function A (Figure 4.1a) and Function C (Figure 4.1b) are used to provide some diagnosis capability to a system. In this case, an FMEA could be performed on the system and may determine that both functions manifest failure modes that could be assigned a severity rating of *Hazardous*. The MACs and MCIs can then be used to distinguish between the criticality of the models used in the two architectures.

	Cat.	**Haz.**	**Maj.**	**Min.**	**Neg.**
MAC-0	MCI-0	MCI-0	MCI-1	MCI-2	MCI-4
MAC-1	MCI-0	MCI-1	MCI-1	MCI-2	MCI-4
MAC-2	MCI-1	MCI-1	MCI-2	MCI-3	MCI-4
MAC-3	MCI-2	MCI-2	MCI-3	MCI-3	MCI-4
MAC-4	MCI-4	MCI-4	MCI-4	MCI-4	MCI-4

Figure 4.2: The Model Criticality Index (MCI) Matrix

By using Table 4.1, a MAC can be assigned to each of the models shown in the figure. For example, Model A in Figure 4.1a would fall into the MAC-0 category; the system has no available method validating the output of the model, and the model is used directly in the support of Function A. Consequently, Model A would receive a MCI-0 rating. In contrast, Models C, D and E in Figure 4.1b would receive a preliminary assignment of MAC-2 due to their use as an ensemble of models. When combined with the severity category, this would result in each model receiving a MCI-1 assignment. The process of assigning MACs and MCIs will be discussed in more detail in Chapters 5 and 6. In the event either function has access to additional information, the MCIs may be reduced further.

4.4 Variable Criticality

The previous section established an approach to defining criticality indices for each BN model within a BNS. This section addresses the need to extend the further analysis of the criticality of BN models to local properties of a model. This notion of criticality is referred to as Variable Criticality and is defined as:

Variable Criticality - A product of the degree of contribution of local aspects of a model to safety-related functional behaviours, and the severity of any hazards associated with these behaviours.

In this case, 'local aspects' refers principally to the properties of individual RVs within a BN model. However, a RV may itself be a complete BN model, or some other complex local structure [2, 44, 153]. This term is therefore intended to preserve the generality of the statement to include any additional considerations beyond those of a standard RV.

A first step towards exploring the criticality of variables within a BN model is to establish which variables directly influence the behaviours of safety-related functionality. For example, consider a medical diagnosis BN model that is utilised for the provision of two distinct software functions. One function (function A) provides an indication of a patient's vital statistics, while the other (function B) automatically administers pain-relief drugs - based on the output of the model. The former function may not fall under any safety-related considerations, while the latter may have severe repercussions in the event of the failure of that function to behave as intended.

Using the MACs introduced in the previous section, an assurance practitioner may provide the model a high MCI assignment to reflect this. However, functions A and B may utilise a disjoints set of output RVs. From this perspective, RVs directly supporting the provision of function A (RV-A) may appear to have a lower criticality than those directly supporting function B (RV-B): errors in the states of RV-A may not directly affect the provision of function B and vice versa. This observation leads to the introduction of a new class of RV to capture these assurance considerations: the Safety-Related Variable (SRV). This is defined as:

Safety-Related Variable - Any random variable whose state, value or configuration is used directly in the provision of a safety-related system function.

The assignment of the SRV classification to individual variables within a BN model provides an immediate insight into which areas of that model to prioritise for evaluation and testing. However, as has been indicated in Chapters 2 and 3, the behaviours of a BN model are heavily dependent on the interactions between RVs within a model. This requires an approach that augments the straightforward classification of variables as SRVs by explicitly evaluating interactions in order to identify which other variables influence the SRVs, and most importantly, to what extent.

The basis of the analysis of the criticality of variables in BN models is therefore predicated on two steps:

- A safety focussed analysis that extends the analysis performed in the assignment of MCIs in order to identify SRVs;
- An analysis of the dynamics of a model that integrates safety information in order to categorise the degree of influence of all variables upon SRVs.

Practically, the first step can be achieved through the utilisation of the RM-BNS to decompose a 'model' object into local models (i.e. RVs). These can then be directly associated with individual software functions. This is an extrapolation of the process outlined in section 4.3.2. The second step requires the development of a systematic approach to exploring the parametric space of a BN model to characterise the model's interaction, and to integrate safety information into this analysis. The proposed methodology supports the targeted evaluation of individual variables within the model and maintains a degree of traceability down to the level of RVs. This can then be used to prioritise assurance activities associated with both the system aspects addressed by the RM-BNS Model Viewpoint, as well as many of those associated with the RM-BNS Data Viewpoint. The methodology is based upon the introduction of three key concepts: an augmented sensitivity analysis, Variable Authority Categories (VACs) and Variable Criticality Indices (VCIs). The methodology is the focus of this section.

4.4.1 Modified Sensitivity Analysis

In the context of BN models, the term Sensitivity Analysis (SA) is used to refer to a range of techniques for analysing how the dynamics of a BN model change in the presence of perturbations applied to a model's structure or parameterisation. Generally, the aim of these analyses is to quantify how the output of a BN model depends on its inputs. For example, in a medical application, practitioners may be interested in establishing the relative importance of a specific subset of symptoms upon a specific disease diagnosis. In this case, various forms of SA can be used to rank the symptoms in terms of importance, or to bound the range of values a variable's parameters may take to guarantee certain behaviours. Fundamentally, SA provides insight into the interactions between RVs in a BN model. It therefore forms an ideal basis for the development of an approach to categorise the criticality of variables based upon the influence of variables upon one another [170].

There are many forms of SA in active use. For the purposes of the analysis proposed in this chapter, one of the simplest forms is used as its basis [59]. This is sometimes referred to as a Sensitivity to Findings Analysis (SA-F). As with other approaches, this analysis seeks to monitor how the posterior distributions of RVs in a given model change in the presence of inputs. Specifically, by monitoring the behaviours of individual RVs in a model through the systematic introduction of evidence, the analysis builds up a picture of the degree of influence of individual RVs upon one another.

The analysis algorithm can be summarised as follows. First, a target variable is selected; this can be any variable in the model. A set of query variables is identified (this set may not include the target variable). Inputs are then provided to the query variables. With the inputs provided, the posterior distribution of the target variable is then computed, and the change (perturbation) in the distribution is recorded. This process is repeated systematically for each of the variables in the model. Practically, this process measures the response of individual variables in a model to inputs elsewhere in the model. Consequently, the magnitude of changes induced in a (target) variable given a change of state in another (query) variable can be used to estimate the influence of one variable upon the other.

With the general methodology of SA-F defined, there remains the problem of selecting a measure to quantify the degree of influence. Most common SA techniques utilise entropy-based metrics [59, 65, 171]. For a discrete probability distribution, the entropy can be defined as:

$$H(X) = -\sum_{i=1} P(x_i) log P(x_i) \tag{4.1}$$

Where $P(x_i)$ is the i^{th} state of the discrete RV X. The entropy provides a measure of the dispersion of a distribution: it is an expression of the degree of uncertainty in a distribution [3]. In the case of a RV modelled using a discrete distribution, a state of low entropy would correspond to a case in which a single state has the majority of the probability mass of the distribution (i.e. one state is close to 1.0). In contrast, a maximally entropic state would correspond to a distribution for which all states have equal probability mass (i.e. all states are equally likely). In the form provided in Equation 4.1, the entropy of only one distribution is computed. A common entropy-based measure for capturing the changed induced by a second variable is the Mutual Information (MI) metric. This is expressed as [59]:

$$I(X,Y) = H(X) - H(X|Y) = \sum_{i=1}\sum_{j=1} P(x_i|y_j) log \frac{P(x_i y_j)}{P(x_i)P(y_i)} \tag{4.2}$$

Where the first term is the entropy of variable X and the term $H(X|Y)$ is given separately as:

$$H(X|Y) = -\sum_{i=1} P(x_i, y_j) log P(x_i|y_j) \tag{4.3}$$

This represents the conditional entropy of the two RVs X and Y. The conditional entropy provides a measure of the amount of information required to characterise a random variable (X) *given* the state of a second random variable (Y) is observed [2, 172]. This term captures the conditional influence of one variable upon another. Using this, the MI (I) of variables X and Y provides a direct statistical measure of the dependence between two RVs: it measures the amount of information the two variables share. If the two RVs are completely independent, Equation 4.2 will evaluate to zero. In all other cases, the MI will produce a non-zero value. However, in these cases, the value of I can take a range of non-zero values. This makes it less intuitive as a measure for categorising the dependence of variables (as is the aim of the modified SA presented here). A Normalised MI (NMI) measure can be used instead. This is sometimes referred to as the uncertainty coefficient of two variables [2, 172]. This can be expressed as:

$$U(X|Y) = \frac{I(X,Y)}{H(X)} \tag{4.4}$$

This measure scales the MI score between the values of 0 and 1. This can naturally be expressed as a fraction or percentage - this is likely to be a more intuitive measure for BN developers and assurance practitioners. For clarity, in the context of a discrete RV the NMI represents the fraction of bits of variable X that can be predicted given knowledge of the state of variable Y. Importantly, the measure is asymmetric. This enables the SA-F to capture asymmetric influence within a model (i.e. cases in which variable X may have more influence on variable Y than Y has on X). This is an essential feature: asymmetric influence within a BN may have serious assurance implications. For these reasons, the NMI is used as the selected influence measure.

With both the analysis framework defined and an influence measure selected, the modified SA analysis can be introduced. There are two principal modifications to the SA-F algorithm described previously. First, at each step in the algorithm, a single query

variable is selected, and each state of that variable is iteratively instantiated. For each state instantiated a vector representing the NMI score for each other variable in the model is obtained. Each element of this vector captures the influence of the instantiated variable upon each other variable given the instantiated variable's current state. Therefore, for each query variable a set of NMI vectors is acquired. The maximum NMI score for each variable over each vector is retained.

For example, in the event a query variable has two states, one step in the analysis would produce two NMI vectors. Each element in each NMI vector represents the influence of the query variable upon the i^{th} variable in the model. In this case, the i^{th} elements of each NMI vector can be used to produce a 2-element vector capturing the NMI for each state of the query vector. The maximum NMI score from this vector is selected. More commonly, averages of the NMI scores are used as this gives a clearer overall view of the relative influence of variables within the model. However, the worst-case (i.e. strongest influence) scenario is typically of interest to assurance practitioners. Therefore, this is the motivation for using the maximum NMI score: it reflects the most influence a variable may have upon another. This is the second modification to the algorithm. Concretely, this can be expressed as retaining:

$$Score = max(U(X|Y)) \tag{4.5}$$

This process is repeated iteratively for each variable in the BN model. The result of this process will be an $N \times N$ sensitivity matrix (SM), where N is the number of individual RVs within the BN model. Each element within this matrix will then represent the maximum influence of the i^{th} variable upon the j^{th} variable. The diagonal elements of the SM (i.e. where $i = j$) will be exactly 1.0. These small modifications to the algorithm can be used to generate a 'heat-map' of influence of variables in a model. This visualisation can be used to provide an intuitive interpretation of the dynamics of a BN model to assurance practitioners.

Without further analysis - or the integration of assurance information - Figure 4.3 provides an intuitive overview of the degree of influence (between 0 and 1) that individual variables have over one another in this BN model. For example, the figure indicates that the state of the 'XRAY' variable strongly influences the 'LUNG' variable. In this context, the 'XRAY' variable represents an input derived from the results of a patient's x-ray, specifically whether the x-ray indicates any abnormalities in a patient's lungs. The

'LUNG' variable represents a diagnosis of whether a patient has lung cancer. It also shows that 'LUNG' (i.e. the presence lung cancer) is strongly influenced by the 'SMOKE' variable (i.e. whether the patient smokes).

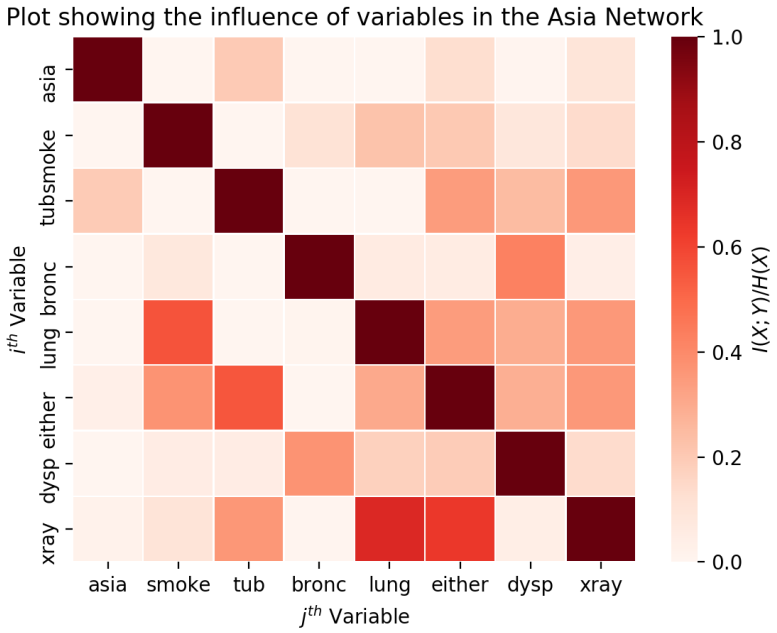


Figure 4.3: Visualisation of Variable Sensitivity in Asia Model

It may be clear to the BN model's developers that as a crucial diagnosis variable, it is particularly important to assure the behaviours of the 'LUNG' variable. In this case, it may also be clear that both whether a patient smokes and the results of a lung x-ray are strong indicators of whether or not the model indicates that a patient has lung cancer – as shown by Figure 4.3. However, in more complex models, these interactions may not be obvious and may be highly counter-intuitive [173, 174, 175]. Furthermore, this method provides a quantification of the degree of influence, and therefore provides a more concrete assessment of the interactions and relative importance of variables in the BN.

4.4.2 Justification for Utilising Sensitivity Analysis

From an assurance perspective, SA can be built upon to provide a useful mechanism for exploring the criticality of BN models and their components. As previously discussed, the aim of criticality analyses is to systematically identify and categorise components of a system that are influential to the behaviour of that system and are therefore disproportionately important to the safe functioning of the system in question.

Sensitivity analysis provides an analogous mechanism: the aim of sensitivity analysis is to characterise a BN model such that BN developers can identify which specific model components are most influential to specific model behaviours. These model behaviours could be anything from identifying which aspects of a model are most influential with respect to specific diagnoses or a given range of values of interest that an RV may take.

From this perspective, sensitivity analysis provides the ideal technical basis for extending the notion of criticality to model aspects of a BNS. By integrating additional assurance concepts into existing SA techniques and thereby mapping specific behaviours to corresponding severity categories, conventional SA approaches can be adapted to explore the dynamics of a given BN model in a more assurance-focussed manner. Concretely, SA provides a technical mechanism for addressing both of the points identified in Section 4.3.1 as key concerns of assurance practitioners. These were mechanisms that enable:

1. The determination and description of the criticality of aspects of a system's architecture.

2. The moderation of confidence in aspects of a system's architecture given its criticality.

Beyond these assurance considerations, a range of SA techniques are widely used and studied within the BN domain. This has produced a rich and varied body of literature that could be harnessed to further explore assurance-focussed properties of BN models. Furthermore, these techniques could be substituted for the technique presented in this section if the need arose. This will be discussed in more detail in Chapter 7. SA-based criticality analysis techniques are appealing as they can provide flexible, informative analysis approaches for identifying the influence of safety related variables upon one another, and is underpinned by rigorous technical, quantitative theory.

4.4.3 Variable Authority Categories

The next step in assigning Variable Criticality Indices is to develop a notion of authority categories for local structures that is analogous to the MACs introduced in section 4.3.2. Unlike the MACs, the Variable Authority Categories (VACs) are defined quantitatively. They directly utilise the results of the modified sensitivity analysis. A VAC is defined as:

- **Variable Authority Category** - The degree of exposure of one variable to the state of another variable.

For each interaction captured in the SM, a VAC is assigned. This assignment reflects the degree of exposure of the j^{th} variable to the state of the i^{th} variable. The quantitative threshold used to define the VACs are shown in Table 4.2. The thresholds reflect evenly distributed quintiles for all possible s_{max} scores. Figure 4.6a shows an augmented version of the basic heat-map shown in Figure 4.3. This visualisation shows the assigned VACs to each interaction in the SM.

Table 4.2: Variable Authority Categories (VACs)

Category	**Name**	**Score (s_{max})**
0	Very High	$S > 0.8$
1	High	$0.8 \geq S > 0.6$
2	Medium	$0.6 \geq S > 0.4$
3	Low	$0.4 \geq S > 0.2$
4	Negligible	$S < 0.2$

4.4.4 Justification for Variable Authority Categories

As discussed in Section 4.3.3, the use of the categories and scores as provided in Table 4.2 is motivated by the fact that the normalised mutual information score is fixed between the range zero and one. The value of this score can be directly interpreted as providing information on how coupled two variables are. Consider two variables A and B. Practically, if the score computed for the case P(A—B) is 0.8, this indicates that knowledge of the state of variable B increases knowledge of the state of variable A by 80%. Fundamentally, this would indicate that variable A has a strong correlation to certain states of variable B. Similarly, all combinations of pairs of variables in a model can be interpreted this way. It naturally provides the quantified degree of influence of one variable on another. Note that the method does not distinguish which state in particular is strongly coupled, just that the variable has a strongly coupled state.

As indicated, the breakdown of the Variable Authority Categories (VACs) in Table 4.2 that are based on these scores was derived by taking the quintiles of the possible normalised mutual information scores. This was been motivated by the need to provide an intelligible, constructive approach to categorising the relative influence of variables upon one another that lends itself to a fully automated methodology. However, as with any technique that attempts to apply a finite set of categories to an infinite set, information is lost. For

example, the provided approach will treat a variable with a score of 0.81 (VAC-0) as equidistant from another variable with a score of 0.61 (VAC-1) and a third variable with a score of 0.79 (VAC-1).

Despite this, the proposed approach and the given table make a complex, infinitely-valued problem tractable and intelligible, and therefore its utility should outweigh the previously discussed risks introduced as a by-product. This is another manifestation of the considerations outlined in Section 4.3.1: to provide a practicable solution to the problem of delineating the relative influence of variables upon one another, a compromise such as this is necessary. Practically, the provided table should therefore provide a useful guide to the relative safety-related importance of variables within a BN model.

Future work should look at validating the use of the quintiles as they have been defined here, and in particular should aim to identify any potential systematic weaknesses of the categories provided in the table, and of their use. This could be achieved through the evaluation of a more complex real-world system and the analysis of the potential safety effects of applying the categories to such a system. An analysis of this form should also evaluate whether evenly distributed quintiles is appropriate for the given system. Such an analysis was beyond the scope of this thesis.

4.4.5 Variable Criticality Index

Finally, a VCI can be assigned to each interaction. As with the assignment of MCIs, this requires the introduction of a notion of severity; once again, the definitions of severity categories (SCs) as defined in DO-178C are used. A severity category is assigned to a variable as follows:

- All variables that directly support the provision of safety-related functions (SRVs) are assigned a severity category that is appropriate given the most severe hazard related to any failure modes to which a given SRV may contribute.
- All remaining variables are assigned the No Effect category (4).

The establishment of the severity of failure modes associated with errors in a model output used by safety-related functions (i.e. a SRV) can be established in a manner analogous to that set out in section 4.3.4. By extending the RM-BNS model to represent variables, failure analyses can explore the contributions of individual RVs to hazardous system behaviours. The assignments of VCIs are based upon the combination of SCs and

	Cat.	**Haz.**	**Maj.**	**Min.**	**Neg.**
VAC-0	VCI-0	VCI-0	VCI-1	VCI-2	VCI-4
VAC-1	VCI-0	VCI-1	VCI-1	VCI-2	VCI-4
VAC-2	VCI-1	VCI-1	VCI-2	VCI-3	VCI-4
VAC-3	VCI-2	VCI-2	VCI-3	VCI-3	VCI-4
VAC-4	VCI-4	VCI-4	VCI-4	VCI-2	VCI-4

Figure 4.4: The Variable Criticality Index (VCI) Matrix

VACs. The VCI matrix is shown in Figure 4.4. For each interaction between the i^{th} and j^{th} variables in a model, a VCI is assigned based on the VAC of the interaction and the SC of the j^{th} variable. This process will generate multiple VCIs for each variable. The final VCI is established by selecting the *most critical* VCI assigned to any interactions associated with that variable.

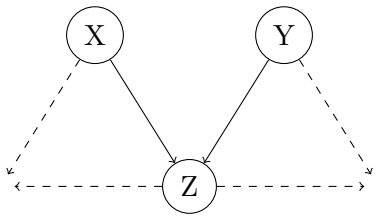


Figure 4.5: A Simple BN Model Fragment.

For example, consider the small network shown in Figure 4.5. Variables X and Y have been assigned SCs of Hazardous and Minor respectively; variable Z is not a SRV and is therefore assigned the No Effect category. Further analysis assigns a VAC of 1 to the interaction between variables Z and X (i.e. the influence of Z on X) and a VAC of 2 on the interaction between variables Z and Y. The former interaction would receive a VCI-1

assignment, while the latter interaction would receive a VCI-3 assignment. Variable Z would therefore receive a final assignment of VCI-1.

An example application of the VCI matrix to the Asia Model used previously can be seen in Figure 4.6b. For the sake of this example, the variables representing disease diagnosis variables ('LUNG', 'BRONC', 'TUB' and 'DYSP') were assumed to be SRVs, with the 'LUNG' variable being assigned a 'Hazardous' classification, 'BRONC' and 'TUB' receiving 'Major' classifications and 'DYSP' receiving a 'Minor' classification. As the figure shows, this provides an overview of the criticality of each variable with respect to each other variable. Table 4.3 shows how the VCIs can shift with inclusion of interactions.

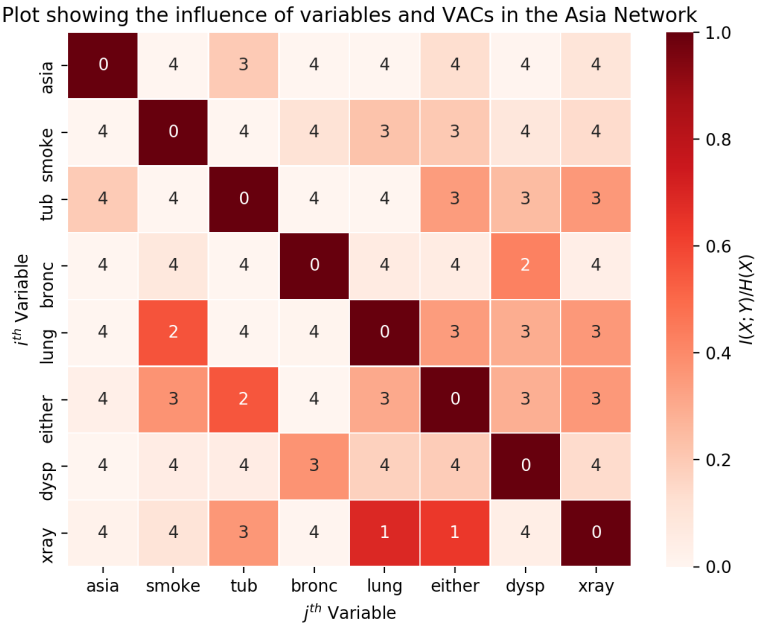
The preliminary VCI assignment in Table 4.3 reflects the criticality indices that would be assigned to each variable without analysing interactions. This in effect assumes all off-diagonal sensitivity matrix elements are set to zero, and all diagonal elements are set to one. This can of course be defined without using the modified sensitivity analysis. This highlights the additional insight assurance practitioners can gain into the safety-related behaviours of a BN by explicitly addressing the role of a model's internal dynamics on a model's outputs (and therefore SRVs).

Table 4.3: Example VCI Assignments

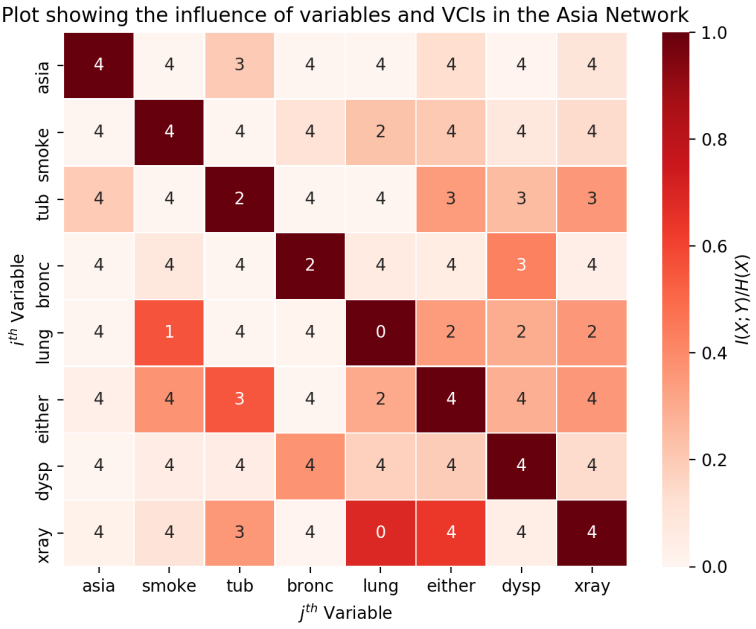
Variable	**Preliminary VCI**	**VCI**
ASIA	4	3
SMOKE	4	2
TUB	2	2
BRONC	2	2
LUNG	0	0
EITHER	4	2
DYSP	4	4
XRAY	4	0

4.4.6 Justification for the Utilisation of Criticality Matrices

The matrices shown in Figures 4.4 and 4.2 were once again adopted with the considerations presented in Section 4.3.1 in mind as a mechanism for capturing the relationship between the influence of various model aspects within a BNS and the severity of hazards associated



(a) Visualisation of VAC Assignments to Asia Model Variables.



(b) Visualisation of VCI Assignments to Asia Model Variables.

Figure 4.6: Visualisations of VAC and VCI Assignments to the Asia Model.

with those system aspects. In existing literature, matrices of this form are widely used to provide a simple, standard mechanism for analogous tasks.

As discussed, the initial concept for the tables and matrices in this chapter was inspired by the manner in which Software Control Categories (SCCs) and Software Safety Criticality Matrices (SSCMs) were defined and structured within the software standard MIL-STD 882-E. Once again, the aim of adopting structures similar to these in this chapter is to provide a useful, pragmatic solution to the problem of delineating the criticality of a system aspect and the corresponding assurance required in that aspect.

Practically, Figures 4.4 and 4.2 (and their related tables) can be considered as an illustration of how BNS development practices and concerns can be mapped into existing practices within the assurance domain, and a way of utilising a standard approach to mitigate the epistemic uncertainty associated with the development and categorisation of the criticality of aspects of any software system in the context of BNSs. Again, practitioners utilising these structures should be aware that they do not guarantee a solution to a given problem but should provide useful guidance on how to pragmatically structure and communicate discussions around the criticality of BNS-specific system aspects among system stakeholders.

Future work should once again focus on evaluating and refining Figures 4.4 and 4.2 in the light of practical experiences of their application to the development of an operational mission-critical BNS. In particular, evaluation of the progression of MCIs and VCIs within these figures should be reviewed to ensure they are of utility to practitioners, and that no violations or important edge cases are identified. Until such a point, practitioners are advised to consider these structures as guidance and assurance heuristics, and to carefully consider their use during the development of a BNS.

4.4.7 Application of VACs and VCIs

The process for the assignment of VACs and VCIs to local aspects of a BN model has been designed to be agnostic to the specific form of sensitivity analysis used, provided the analysis technique selected satisfies two requirements:

1. The technique provides quantified measures of all *pairwise* interactions between variables in a BN model.

2. The technique utilises any asymmetric, normalised measure of the degree of interac-

tion between variables, and that this measure can be mapped directly onto the VAC categories shown in Table 4.2.

The modified sensitivity analysis algorithm introduced in this chapter is therefore intended to provide a minimally viable solution to the analysis of interactions within a BN model. It was selected as it provides a relatively intuitive, computationally cheap approach to evaluating interactions within a given BN model; it is an ideal 'baseline' algorithm.

However, it has a number of important limitations. Of these limitations, the most important arises from the fact the algorithm falls into the 'one-at-a-time' (OAT) class of SA techniques. This class of SA does not exhaustively explore the parametric space of BN models. By analysing pairwise interactions between variables by changing the state of a *single* variable at a time, they do not evaluate the pairwise interactions of two variables in the presence of *additional* inputs.

For example, OAT analyses can not evaluate circumstances in which the presence of two inputs results in the creation of new *active paths* in the model. These active paths can enable the exchange of information between two or more variables that were previously d-separated.[2] This means that OAT analyses may indicate a reduced degree of pairwise interactions between two variables, and this may not accurately represent the 'true' degree of interaction between the variables.

From this perspective, OAT analyses (and by extension the modified SA technique introduced here) can be considered as providing a 'first-order approximation' to the interactions within a BN model.

This algorithm can therefore be considered as something of a 'first-order approximation' to the interactions within a BN model. More elaborate analysis methods have been proposed and implemented in literature. Many of these approaches could be used in place of the algorithm presented here. The only requirement of any replacement algorithms is that they quantify the interactions in terms of the NMI score introduced in this section.

[2]See section 2.2.3.1 for more information on the role of BN model structures in the transmission of information a network.

4.5 Methodology

Taken together, the techniques introduced here form the basis of a proposed Model Criticality Analysis (MCA) methodology. This methodology is aimed at providing assurance practitioners with an intuitive grasp of which model-centric aspects of a BNS are most important from an assurance perspective. To summarise, the MCA methodology introduced here is composed of the following seven steps:

1. Create an RM-BNS architecture model for a BNS.
2. Carry out an FMEA-like failure analysis on functional aspects of the system.
3. Assign MACs to each model within the BNS.
4. Apply the MCI matrix to each model within the BNS.
5. Perform a modified sensitivity analysis on each model in the BNS.
6. Assign VACs to each variable within each model in the BNS.
7. Apply the VCI matrix to each model in the BNS.
8. Extract final VCI assignments for each variable in the BNS.

The primary utility of this methodology is intended to be the indication of which models and aspects of models are the most critical; this information can then be used to support the subsequent *targeted* analysis of model-centric aspects of a BNS.

Moreover, by facilitating these analyses, the approach is intended to provide a means of addressing assurance considerations associated with other viewpoints outlined in the RM-BNS. In particular, it can be used to support the assurance of some Data Viewpoint aspects. For example, in many applications, individual variables within a BN model can be trained independently. Consequently, individual variables may be directly associated with individual data artefacts. Therefore, the MCA approach can be used to identify individual data artefacts (or elements of data artefacts) that are particularly important to the reasoning capability of a BN model.

4.6 Conclusion

This chapter has introduced a systematic approach to establishing the criticality of each model used within a BN-based system, and of all variables that comprise each model.

The motivation for this work was threefold. First, there is a need to ensure that the traceability of behaviours is maintained in BNSs in cases in which traditional software-centric approaches may begin to break down. Second, it is essential that unconventional aspects of BNSs can be directly targeted and prioritised based on their relative importance to the safety-related functional behaviours of a system. Finally, there must be a mechanism for ensuring the proportionality of analysis and testing techniques used in the verification and validation of BNSs.

The Model Criticality Analysis (MCA) approach presented here has provides a solution to each of these problems for those system aspects addressed by the RM-BNS Model Viewpoint. The approach provides a targeted means of analysing model-centric aspects of BNSs to establish the criticality of these aspects to the behaviour of the completed system. It supports the attribution (though not in a binary sense) of individual model components to a system's functional behaviours. By providing a means of prioritising individual model components, the proportionality of the efforts carried out in the assurance of those components can then be established.

The primary contributions of this chapter are therefore:

- The introduction and definition of Model Authority Categories (MACs) to systematically characterise the relative importance of models.
- The introduction and definition of Model Criticality Indices (MCIs) to capture the criticality of independent models.
- The introduction and definition of Variable Authority Categories (VACs) to standardise the description of the influence of variables upon one another.
- A modified sensitivity analysis algorithm for deriving VACs.
- The introduction, definition and methodology for assigning Variable Criticality Indices (VCIs) to capture and standardise the description of the relative importance of variables upon safety-related outputs (SROs).

The following chapter looks at how the RM-BNS and verification and validation objectives of the previous chapter can be used alongside MCA and other existing evaluation and analysis techniques to develop a framework for describing safety evidence generated during the development of a BNS, and how the sufficiency of this evidence may be established.

Chapter 5

From Objectives to Evidence

5.1 Introduction

So far, the work presented within this thesis has focussed upon making two principal contributions: first it has aimed to ensure the full scope of assurance considerations are understood and accounted for by system developers (Chapter 3); and second, it has provided an targeted safety analysis technique to help explore the assurance implications of BN model aspects (Chapter 4). This chapter aims to provide guidance on addressing the outstanding problem of establishing the sufficiency of the assurance efforts undertaken during the development of a BNS.

In particular, this chapter focusses primarily on the sufficiency of assurance efforts related to the RM-BNS Model Viewpoint. This viewpoint has been selected as it represents the most conceptually divergent of the RM-BNS viewpoints with respect to conventional software systems: the other RM-BNS Viewpoints have novel BNS-specific concerns associated with them, but these viewpoints typically have more conceptual overlaps with conventional software development practices. While the focus of the discussion will be on system aspects associated with the Model Viewpoint, the concepts introduced in this chapter are relatively general and should therefore be extensible to system aspects associated with the other RM-BNS viewpoints. The chapter is structured as follows:

- The challenges associated with the development of a BNS are re-introduced to provide context for subsequent sections. This provides a brief review of relevant concepts introduced in Chapters 2 and 3.

- The role of evidence in existing safety-critical systems is then reviewed. This pro-

vides additional context and highlights potential shortcomings in existing standards and practices. A framework for describing and classifying safety evidence is then introduced and applied to BNSs.

- Next, the evidence classifications are combined with the RM-BNS verification and validation objectives to provide an overview of how evidence may be used to satisfy the objectives, and how evidence may be refined alongside these objectives. This section provides a mapping of objectives to the techniques that may be used to satisfy them, and their associated items of evidence. It also builds upon the contributions of Chapter 4 to indicate how criticality metrics may be mapped onto the RM-BNS objectives and evidence to further clarify the role and relative importance of specific items of evidence in a given BNS.

- Finally, the chapter provides a discussion on how the sufficiency of evidence may be established during the development of a BNS. This section reviews two potential approaches and evaluates the relative merits of each.

5.2 Assurance Considerations

As discussed in Chapter 2, existing safety standards commonly adopt a prescriptive approach to safety: a standard will stipulate a set of mandatory development processes that must be performed if a system is to comply with that standard. These approaches frequently provide either an enumeration of techniques that must be carried out by a development team, or a set of activities (or specific items of evidence) that must be performed (or provided) in order to satisfy a given set of safety objectives [69]. These techniques typically make strong assumptions about the nature of the system being developed and of the utility of the practices they mandate [70].

Many such standards have earned enduring popularity within the various engineering domains; the well-defined activities and development steps lend themselves to large projects and organisations – and to the understandably risk-averse mentality of many such projects and organisations. Indeed, a recurrent criticism of prescriptive approaches to safety is that these approaches may encourage an uncritical 'check list' mentality to assurance [76, 100]. This criticism is rooted in the fact that the rationale for the activities and evidence mandated by a given standard is commonly only implicit. Many authors have noted that the underlying assumptions of these standards may not be valid in practice:

the generality and flexibility of various analysis techniques and best practices is frequently debated within the assurance domain [74, 75, 162].

In previous chapters, a number of important conceptual distinctions between BNSs and more conventional software-intensive systems were introduced. As outlined in Chapter 3, these conceptual distinctions may erode the validity of some of the assumptions associated with modern software safety standards [64]. The shift in the importance of aspects of the system's design away from traditional software engineering concerns will be particularly problematic. In Chapter 2 the results of the DeepXplore study into the analysis of Deep Learning (DL) systems for autonomous vehicles were outlined. In this work, the researchers highlighted that conventional software testing strategies were inadequate for the task of assuring the functional behaviour of a DL system. This research indicated that despite achieving complete code coverage of the DL system, less than 10% of the system's model space was explored: there remained nearly 90% of the model's behaviours that were unexplored by conventional code-structure focussed testing [113]. As discussed in Chapter 3, in many cases it is the *models* that drive these systems that should be considered to be the primary determinant of their functional behaviour.

Moreover, the complexity and scope of many of the applications of AISs introduces additional assurance considerations: the altered role of the environment and the related operational context upon the behaviour of an AIS may further erode the validity of some of the assumptions underpinning existing software engineering best practices. This is particularly true for those activities and standards that make very strong assumptions about the relationship of a software system to its operational environment [81].

For example, aviation safety standards may be particularly exposed to these assurance concerns: they have long assumed that the operational environment is highly regulated and that the software systems are relatively 'closed off' to this environment. Both of these assumptions will be challenged with modern AISs – many of these systems are (sometimes dynamically) modelling their environments and receiving feedback from the environments they operate in. This can create an information challenges associated with the interaction of an AIS with their operating environments [81, 176, 177].

Furthermore, the nature of BNSs is such that they actively integrate highly uncertain information into their design, development and functionality [2]. As discussed in previous chapters, the lifecycle of a BNS is commonly characterised by highly iterative development practices and may require a significant degree of empirical and statistical validation – the

BN models are ultimately probabilistic representations of some aspects of the world and can often only be validated against the world [1, 2, 22]. Standard approaches of 'offline' software testing will be of limited use in identifying or mitigating hazards in many cases. In this sense the development of a BNS introduces considerations more akin to those common to hardware engineers, with reliability quoted statistically and verified through 'physical' testing. This raises two points:

1. The complexity and scope of modern AI applications coupled with the technical properties of the AI approaches themselves will generally limit the utility of defining a concrete set of development activities or prescriptive practices for these systems. Constraints on the development and implementation of these systems may actually hinder the capabilities of a BNS and the constraints may compromise the effectiveness (and safety and/or security) of the system. The inclination to apply such constraints to a BNS or AIS may inadvertently introduce more error modes than they may mitigate [178].[1]

2. The non-deterministic properties of these systems will further limit the ability of safety practitioners in applying conventional software assurance activities during the development of a BNS. It may not be possible to unequivocally satisfy some predefined set of objectives: the satisfaction of objectives will be more heavily based upon statistical confidence in the behaviour of a system as opposed to the concrete Boolean satisfaction of objectives common to conventional software systems. It may not be possible to know *a priori* the statistical confidence that is necessary and/or achievable prior to the development and implementation of the system [2, 6].

These points are generally symptomatic of standard BNS development lifecycles as being more of a 'dark art' than a traditional engineering process [2]. The development of a performant BNS requires an uncommon degree of latitude on the part of the BN developer in making design decisions for a given system. Many of these design decisions will only become known as development progresses. This indicates that any guidance on the assurance of a BNS should focus on providing a flexible framework that accommodates the iterative and somewhat erratic lifecycle that can typify BNS development activities.

[1] From a technical perspective, the introduction of additional constraints on a model may introduce problems with errors arising from *high bias* and over-regularisation. Chapter 2 discusses some of these aspects in more detail.

It is essential that the *rationale* for the use of specific techniques is made explicit, and the relevance of any information produced by these techniques is disseminated to all applicable stakeholders.

5.3 An Evidence Framework for BNSs

Accommodating novel designs or applications is not a problem unique to BNSs. There are many well-known problems within the safety domain in mapping existing software safety standards and certification practices to the development of unconventional systems [179, 180]. There have also been problems with the certification of legacy systems – those systems that were developed prior to the introduction of a certain standard or set of assurance practices and therefore cannot easily be brought into retroactive compliance with a given standard [181]. This has driven interest in establishing standards that avoid highly prescriptive and inflexible approaches to safety. One approach that has become increasingly accepted within the safety engineering community is that of evidence-based safety standards. These approaches provide reduced prescriptive guidance for assurance activities in favour of emphasising the explicit discussion of the rationale behind the activities and safety lifecycle adopted for a given system. Proponents of this approach to system safety propose two primary benefits:

- The approach supports the direct mapping of safety goals to evidence, enabling the explicit consideration of the sufficiency of individual items of evidence in the satisfaction of individual safety goals.
- The approach avoids the late-stage (and costly) re-development work sometimes associated with failing to meet prescriptive standards.

Practically, these are attractive benefits in the context of developing BNSs. As discussed throughout this thesis, the development of a BNS includes a number of activities and design decisions that may prove resistant to the application of conventional software engineering practices: the BN model design, development and evaluation phases of a BNS's lifecycle are often characterised by their highly iterative nature. This can make traditional requirements decomposition activities both challenging and potentially inefficient. This also means that the techniques or evaluation activities (and associated evaluation metrics) that will be used by assurance practitioners may not be known until well into the system's development lifecycle [2, 6, 66].

So called 'evidence-based' approaches to assurance may therefore provide a more appropriate means of assuring BNSs: they ensure that the rationale for specific development and evaluation techniques is made explicit (and by extension avoid mandating the use of techniques or practices that may be irrelevant or uninformative to a given application). This more readily accommodates the shifting design decisions and evaluation techniques employed during the development of a BNS and the more 'dynamic' lifecycles of these systems.

While many industry standards still adopt prescriptive approaches to system safety, standards are being introduced that encourage or require evidence-based approaches to assurance. Perhaps the most significant of these is the UK Ministry of Defence (MOD) Defence Standard 00-056 (DS 00-056). This standard focusses on defining a set of safety goals that are then refined for a specific system. It does not mandate a set of processes for the satisfaction of these goals but instead demands that an *argument* is made to justify the use of items of evidence in the satisfaction of these goals. Furthermore, DS 00-056 discusses the characteristics of evidence used in the satisfaction of a given safety goal. For example, it states:

> *"Explicit, objective evidence [items] are more compelling than those that appeal to judgement, custom or practice."*

This statement alludes to the implicit underlying hierarchy of evidence: not all items of safety evidence are created equal [70]. This hierarchy should be extended to evidence generated for the assurance of a BNS: a BN model walkthrough may not be as compelling as evidence for the presence of a given behaviour as an extensive analytical evaluation of a BN model's dynamics. However, it may not be technically possible or feasible to produce exhaustive analytical evaluations of a BNS, and so reliance on expert testimony may be unavoidable or even desirable in some cases. Understanding the interplay of evidence – what is feasible, available and sufficient – requires a framework for describing and understanding the relationship of evidence to the capabilities and limitations of the techniques that generate them, and to the objective the item of evidence addresses.

5.3.1 Evidence Characteristics

Discussing the characteristics of an item of evidence is somewhat unconventional [70, 100]. As indicated previously, the role of evidence (and the associated rationale behind a given

item of evidence) is not commonly discussed in many current software safety standards [100]. From a safety perspective, the assurance of a BNS will require the generation of items of safety evidence that are highly novel, abstract and BN-specific. Ensuring the transparency of the rationale behind the generation of an item of evidence and its relationship to a given goal or objective is therefore essential. In particular, ensuring that BN developers and assurance practitioners can effectively communicate about these novel items of evidence in the context of the safety of a BNS is key.

One approach to achieving this is to adopt a set of *evidence characteristics* for describing safety evidence generated for the assurance of a BNS. These can be used to provide a generic framework for discussing and communicating the rationale and role of evidence in the context of a given system. A number of evidence characteristics have been proposed within the safety engineering literature. Rather than develop a new set of evidence characteristics, this chapter instead adopts the evidence characteristics as proposed by Weaver *et al.* These are defined as follows:

- **Relevance** - The extent to which the evidence directly addresses the software safety goal.
- **Coverage** - The proportion of the software safety goal which the evidence addresses.
- **Independence** - The extent to which complementary evidence follow diverse approaches to fulfilling the requirement for evidence.

Two further characteristics are also adopted from the work of Menon *et al.* These are:

- **Trustworthiness** - The likelihood that the evidence is free from errors.
- **Replicability** - The ease with which the evidence could be replicated.

In the context of Weaver's work, the former set of characteristics were defined with respect to the relationship of an item of evidence to a safety goal. In the context of this chapter, these five characteristics are used to describe role of evidence in the satisfaction of RM-BNS verification and validation objectives. The satisfaction of an individual objective may rely upon several distinct items of evidence, each of which addresses a specific aspect of that objective.

For example, it is common practice within the BN domain to apply a suite of evaluation techniques to a BN model without providing any explicit rationale for why these techniques

were used and what *specifically* has been demonstrated as a result [73, 75]. Furthermore, few BN evaluation techniques and development frameworks adopt a parallel *adversarial* position on the evaluation of a system in order to explicitly consider the weaknesses of the evaluation techniques used: considerations such as the coverage, replicability and trustworthiness of evidence is rarely explicitly discussed.

In the context of BNSs, each of these five characteristics should be explicitly associated with every item of safety evidence generated during the development of a system. While the definitions provided by Menon and Weaver are sufficiently general to be applied to evidence generated for BNSs, there are a number of additional considerations relevant to BNSs. To illustrate this, Table 5.1 re-defines these characteristics in the context of the RM-BNS objectives, and provides high-level examples of key considerations related to each characteristic.

Perhaps the two most important considerations shown in Table 5.1 are those related to the *Coverage* and *Trustworthiness* characteristics respectively. These characteristics touch on two aspects of AI development that are increasingly recognised as key considerations in the development of safety-critical AI systems. The former characteristic has conceptual links to the work of the DeepXplore team and others within the AI community. These teams are developing techniques that adopt new notions of coverage that specifically target key aspects of AISs [113, 166]. These techniques can be regarded as falling under a new category of coverage testing: '*Model Coverage Testing*'. From this perspective, many of these techniques may also be mapped onto the notion of *Coverage* as defined here with respect to the RM-BNS Model Viewpoint objectives: the degree of evidential coverage may be closely related to the Model Coverage established by BNS developers.

The latter characteristic explicitly addresses common issues with misleading or flawed evaluation outputs – outputs that may be used as safety evidence in safety-critical BNSs. An example of the considerations exposed by this characteristic would be that of the Tank Detection AIS introduced in Chapter 3. This AIS encountered issues with underlying biases in the data and the development process that produced evidence that provided false-confidence in the system's behaviours.

Table 5.1: Evidence characteristics and example considerations for evidence generating techniques in BNS development.

Characteristic	**Description**	**Example Considerations**
Relevance	The extent to which the evidence directly addresses the RM-BNS objective.	(a) In what *specific* way does a given technique produce evidence that addresses a given objective? (b) Are there any alternative techniques that may produce more relevant evidence for a given objective?
Coverage	The proportion of the RM-BNS objective which the evidence addresses.	(a) What degree of *Model* coverage has been established? (b) What factors influence the evidential coverage of a given technique?
Independence	The extent to which complementary items of evidence follow diverse approaches to fulfilling a given RM-BNS objective.	(a) To what degree is an evaluation technique vulnerable to technical limitations that are shared with other techniques used to address the same objective? (b) What degree of correlation exists between the *items of evidence* generated by two or more techniques used to address the same objective?
Trustworthiness	The likelihood that the evidence is free from errors.	(a) In what ways can the evaluation technique be 'fooled'? (b) In what ways can an item of evidence be misleading?
Replicability	The degree to which the evidence could be independently replicated.	(a) What factors may influence the replicability of the evidence generated by a given technique? (b) Are there alternative (more replicable) methods for generating evidence?

Examples of other considerations related to this characteristic include whether or not an evaluation technique controls for problems related to 'reward hacking', or more generally the degree to which an evaluation technique is exposed to technical issues related to so-called 'class-biases' or other related problems.[2]

The remaining three characteristics are also important considerations when generating safety evidence for BNSs. Of these three, the *Replicability* characteristic may be the most challenging aspect for BN developers and assurance practitioners to address. Many of the problems for which BNSs are used are inherently uncertain, and in many cases the assurance of these systems is likely to be at least partially dependent on testing activities that *cannot* be wholly replicable. This is once again a distinction between conventional software systems and BNSs. This characteristic is therefore aimed at exposing these considerations in the context of the *safety* evidence.

5.3.2 Evidence Classifications

These evidence characteristics have been selected to encourage the recognition and description of the limitations and implications of BNS evaluation and testing activities in the production of safety evidence for safety-critical BNSs. By explicitly describing every item of safety evidence in terms of the evidence characteristics, the role of an item of evidence and the rationale behind its use in the satisfaction of a given objective should be more readily exposed to BN developers and assurance practitioners.

However, as outlined in DS 00-056, there is a 'hierarchy of evidence' to consider when assuring a system. This is particularly important when considering a BNS – the context and application of a BNS will dramatically alter the availability and scope of possible evaluation techniques and associated items of evidence. Having the ability to explicitly articulate this hierarchy is therefore the next step in describing the role of evidence in BNSs. One approach to developing such a hierarchy is to rely on *evidence classifications*.

There are several existing evidence classification frameworks within the safety domain [76, 100, 184]. As was the case with the evidence characteristics, this chapter adopts two of these existing classifications instead of introducing a proprietary BN-specific set of classifications. These classifications have been selected (and modified where appropriate) to maximise the relevance of the classifications to BNS safety-evidence. The first set of

[2]Good discussions of the concepts of 'reward-hacking' and 'class-bias' can be found in [63,182] and [183]. An overview of the possible safety impact of 'reward-hacking' can be found in [21].

classifications addresses the *type* of evidence items and is drawn from the classifications proposed in the SHIP [185]. The classifications are defined as follows:

- **Deterministic** – Relying upon axioms, logic, and proof.
- **Statistical** – Relying upon probabilities and statistical analysis. [3]
- **Qualitative** – Relying upon adherence to standards, design codes etc.

These three classifications provide enough information to begin to develop a basic hierarchy of evidence: evidence produced by *Deterministic* techniques should be prioritised over evidence produced by either *Statistical* or *Qualitative* methods. Similarly, *Statistical* techniques should generally be favoured over *Qualitative* techniques. However, establishing the *relationship* of an item of evidence to a given verification and validation objective is also necessary to establish context for that evidence.

For example, an analysis of a model's structure may provide *Deterministic* evidence of a particular property. This may suggest that this item of evidence should have precedent over evidence associated with the two other classifications. However, this can mask important information. The evidence generated by this process may only *indirectly* address a given objective or may require additional evidence to provide supporting context to the implications of the evidence. In these cases, evidence that more directly addresses an objective may be prioritised by developers – even if this other evidence is classified as belonging to one of the other two classes. Three further classifications are therefore adopted here to capture these additional considerations. The classifications are adopted from Weaver and are defined as follows [76]:

- **Direct** – evidence that directly shows the [verification and validation objective] is fulfilled.[4]
- **Backing** – evidence that shows that the direct evidence is soundly based.

[3] This classification was defined in [185] as the Probabilistic classification. Given the overloaded use of this term in the context of probabilistic graphical models – and therefore BNSs – it has been renamed here to 'Statistical' evidence. It is felt this also more appropriately captures the nature of many BNS analysis techniques.

[4] The definition refers to safety goals. To support the clarity of the content of this chapter, this has been substituted for the verification and validation objective.

- **Reinforcement** – evidence that shows that direct evidence can be extrapolated to meet higher integrity requirements than can be met with direct evidence alone.

To distinguish between these two classifications, the former set of classifications will be referred to as the Evidence Type Classification (ETC), while the latter set of classifications are referred to as the Evidence Role Classification (ERC). Together, these classifications can be used to provide a more expressive hierarchy of evidence. In general, those items of evidence that are both *Direct* and *Deterministic* will be the most desirable as they should be fully replicable and most closely align with the sentiment expressed in DS 00-056. However, much of the evidence associated with the assurance of the BN Model Viewpoint will fall into the Statistical classification, and the processes used to generate these items of evidence will influence their evidence classifications.

5.3.3 Evidence-Generating Processes

There are a plethora of AI- and BN-specific techniques that can be employed during the development and evaluation of a BNS with a similarly extensive variety of metrics and considerations associated with their use [2, 65]. Assigning an ETC to an item of evidence (and by extension to the evidence-generating process that produced it) may therefore become challenging for assurance practitioners. This section looks at a selection of evidence-generating techniques commonly used during BNS development and evaluation and classifies them according to the ETC of the evidence they produce.

5.3.3.1 Statistical Processes

Within the BN domain, the most prevalent form of analysis and evaluation techniques are those that produce statistical information. In the context of BNSs used for diagnostic and prognostic applications, two of the most widely used techniques that produce items of evidence that fall into this classification are Receiver Operating Characteristic (ROC) curve analysis, and Cross Validation (CV) testing. Both of these techniques can be used to assess and compare the diagnostic performance of a given BN model against other diagnostic models, or to tune a single model to meet the required performance characteristics.

In the case of the ROC analyses, the aim is to quantify (and typically visualise) the trade-off of a given model in terms of the rate of occurrence of true-positives with respect

to false-positives.[5] Simpler accuracy metrics may provide misleading indications of the performance of a system. For example, a model that is designed to diagnose a single disease may end up diagnosing every patient as having that disease. The accuracy metric for how often such a model correctly diagnoses the disease in a patient can be 100%. However, the false-positive rate would be very high, and the corresponding diagnostic power of the BN would therefore be very low – despite the seemingly impressive headline statistic.

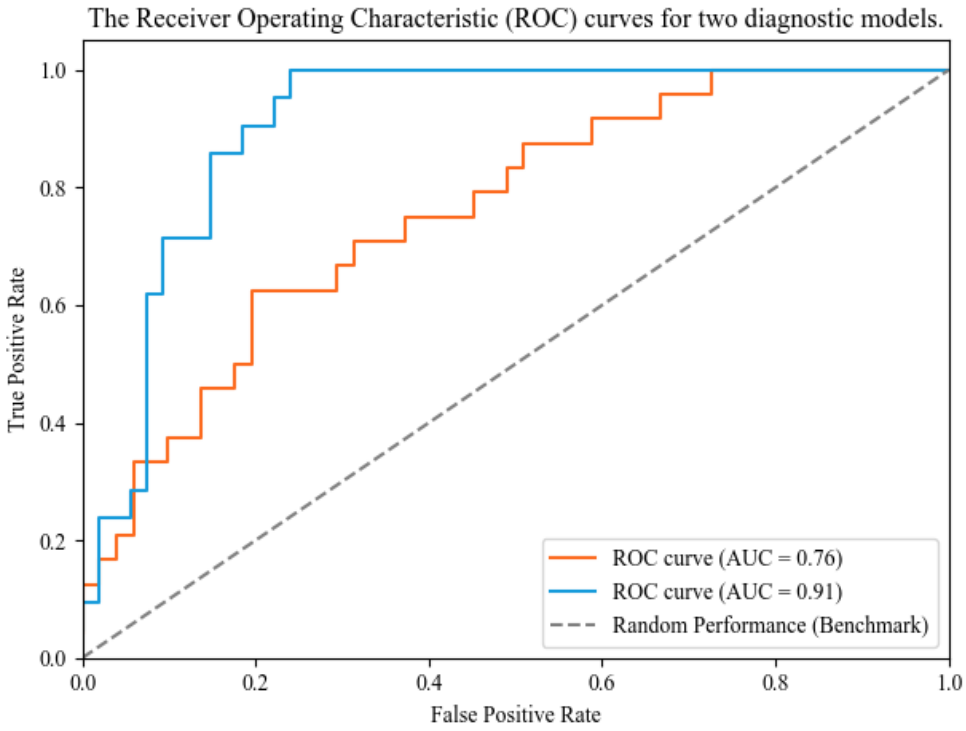


Figure 5.1: A comparison of two Receiver Operating Characteristic (ROC) curves used to evaluate two diagnostic models.

The visualisation of these analyses is generally achieved using a ROC curve. These curves can be used to evaluate and tune the diagnostic performance of a BN model [186]. Figure 5.1 shows an example curve for two simple BN models. Qualitatively, the closer the ROC curve is to the top left corner in Figure 5.1, the higher the performance of a diagnostic model. More technically, the area under the curve (AUC) of each of these ROC curves provides a single metric for the diagnostic performance of each of these models – the larger the value the better the model.

[5] In this context, a true-positive reflects a correct diagnosis, while a false-positive reflects a specific type of incorrect diagnosis.

Consequently, given the information in Figure 5.1 a BN developer is likely to favour the selection and use of Model A over Model B for this task. From an assurance perspective, these analyses may be used to evaluate safety trade-offs between model architectures. As is often the case in the safety domain, the safety implications of a false-positive (e.g. 'drug administered when not required') may have radically different implications than a false-negative in a given context and vice versa. The ROC curves and ROC metrics can be used to directly address these considerations from a model-centric perspective.

A second prominent analysis technique in the AI domain is so-called Cross Validation (CV) testing. The aim of these analyses is not to simply assess the diagnostic performance of a model, but to assess the *generality* of the model.[6] This helps to ensure that a model is developed in such a way that it is not excessively sensitive to the data used to train it.

A practical example of this may occur in a medical diagnosis setting. For example, a diagnostic model may become extremely accurate at making diagnoses *within* the patient data used to train it. However, it may be extremely poor when applied to patient cases outside of this data. Models that demonstrate such behaviours are said to have 'overfit' to their training data. A rough analogy to human learning would be the distinction between 'rote learning' and learning a deeper understanding of a problem: generally, the former should be deprecated in favour of the latter. The same is true for BNs and other AI techniques.

From an assurance perspective, CV testing can be used to provide confidence in the behavioural characteristics of a BN model outside of the cases used to design and train it. In general, CV testing can help to ensure the robustness of the representation of a domain learned by a BN. However, CV testing should not be used alone except in the most trivial of use-cases (or in the context of this thesis: the lowest criticality models). This is because CV testing will not identify biases that persist across all available data, certain forms of human biases in the model development process and generally will not mitigate issues related to the evolution of a domain over time [63, 187, 188, 189]. It may be possible to adapt existing CV testing techniques to target safety-related model aspects – perhaps through the integration of information derived from a Model Criticality Analysis (MCA) or similar technique. Such a modification may prioritise the generality of safety-related model aspects over the broader notions of generality currently used in CV testing.

[6] CV testing is sometimes referred to as out-of-sample testing as it involves exposing a model to 'unseen' data instances.

As the classification suggests, both of these analyses are fundamentally based upon statistical assessments of the behaviours and properties of a model. They each address complementary but distinct BNS development considerations. Evidence generated using these techniques is associated with a degree of inherent aleatoric uncertainty, and the evidence produced will not generally be in a Boolean format. There will be explicit uncertainty in the utilisation of these techniques for the satisfaction of any goals or objectives targeting BN model aspects. In many cases, the bulk of the safety evidence generated for the assurance of a BN model will be derived from evidence generating processes of this class. Unlike conventional software systems, there is therefore a possibility of generating meaningful failure rate statistics for BNSs.

5.3.3.2 Deterministic Processes

Processes that are associated with the production of items of evidence that are designated as *Deterministic* according to the ETC definitions share some similarities with evaluation and analysis techniques commonly referred to as 'Formal Methods' within the software engineering domain. Two of the most prominent techniques that produce this class of evidence are some forms of sensitivity analysis and a group of techniques that could be described as 'independence analyses'. Both groups of techniques are based upon exact mathematical proofs of the properties of BN models.

In the case of sensitivity analyses, there are a range of techniques available to practitioners. Of these, a subset of techniques focus on establishing constraints on parameters within a BN model. For example, they can be used to guarantee certain parameters within a BN model fall within a certain range, or provide information on what value other parameters must take in order to produce certain behaviours in a model [190]. From an assurance perspective, these techniques can be used to provide evidence that a BN model cannot produce certain behaviours. However, they should typically be used alongside other evidence that indicates *why* the absence of such behaviours is desirable.

The second group of 'Deterministic Processes' – independence analyses – broadly focus on identifying similarities and distinctions between one or more BN models, and on providing mathematical proof of the independence properties of these models. For example, they can be used to demonstrate that a group of BN models share independence relationships such that one model may be a more complex super-set of the variables in a second model. They may also be used to show that a variable within a model can never

interact with other variables within that model, and can provide the precise mathematical states in which this holds true. For assurance practitioners, this could be used to demonstrate unequivocally that Safety-Related Variables within a model are *always* conditionally independent of a set of variables for which limited safety evidence is available. Practically, this may mean that structures within a BN model are compartmentalised into those that are strictly safety-related, and those that are not, and the mathematical contexts in which these two parts of the model may interact are precisely defined and used to justify modelling decisions related to the structure and parameterisation of a model.

5.3.3.3 Qualitative Processes

Finally, there are two prominent examples of qualitative evaluation activities commonly used during the development of a BNS. A popular approach to the evaluation of BN models in some applications is to perform a model walkthrough with one or more domain experts. A BN model is often transparent to non-BN developers: in many contexts, experts may interpret the structure and parameterisation of a model with guidance from the BN development team. In this approach, an expert can assess the degree to which a BN model conforms to their knowledge of the domain being modelled. As in other domains, using multiple independent walkthroughs should be favoured in order to mitigate the risk of oversights, biases or other errors on the part of the assessing expert [61, 64, 121]. However, this is not always a straightforward task, especially for larger BN models [61].

Therefore, an alternative approach has been to develop tools that enable a BN model to 'explain' its reasoning given a test case. This can be used to produce an explanation of a model's properties and behaviours in natural language [191]. This can eliminate the need for a detailed review of underlying mathematical constructs in some contexts and can enable a higher-level review of how a model aligns with expert knowledge: it can be used to produce sets of explanations that can be divided between multiple experts, which in turn can reduce the burden when assessing large models [61, 191].

A second evaluation technique comes from the Expert Systems (ES) domain. Knowledge Base Reviews (KBRs) are frequently advocated as an essential part of the development of an ES [85, 192, 193]. In the context of a modern BNS, the aim of a KBR is to provide a detailed review of all data artefacts available pertaining to the ES being developed. Part of the motivation for this is the reduction of the epistemic uncertainty associated with capturing the reasoning of experts within a software system. In particu-

lar, ES developers aim to identify potential gaps in the knowledge of the experts used to develop the ES. This can include inconsistencies in expert opinion or data artefacts, as well as identifying more general epistemic uncertainties relating to potential practical or fundamental limitations o knowledge or data relating to a given domain.

In the utilisation of BNs for ES roles, the experience of the ES developers remains relevant. Detailed qualitative evaluations of data artefacts can be used to provide evidence of many properties of a data artefact, including the integrity and scope of the artefact. With respect to the structure and parameterisation, KBRs can provide evidence that a model has captured the intended domain. Without the careful selection of data artefacts, a BN model may be trained on a data set that does not adequately reflect the domain being modelled. The KBR is aimed at mitigating these issues.

5.3.3.4 Mapping Evidence Classifications to Techniques

These processes represent a subset of the techniques available to BN developers. As with any fast-growing research field, techniques are regularly adopted and modified. The evidence classes (and evidence characteristics) were introduced as a general framework for classifying and describing evidence generated by all of these techniques. The adoption of the framework for the development of mission-critical BNSs could also further strengthen the systematisation of the development of these systems. A complete enumeration of these techniques is beyond the scope of this chapter. Furthermore, such an enumeration would rapidly become obsolete given the pace of change and innovation within the AI domain.

Instead, Table 5.2 shows a mapping of some of the most common analysis and evaluation techniques applied to BNS models to their corresponding ETC. The aim here is to provide a reference point for BN developers and assurance practitioners when describing and classifying other techniques, or modifications to existing techniques within the table. The table does not list the ERC assignments for these processes. These can only be meaningfully defined in the context of a given safety goal or objective that a given item of evidence is used to address.

Table 5.2: Technique-Evidence Mapping for Model Viewpoint

Process	**Classification (ETC)**	**Evidence Items**
Model Walkthroughs	Qualitative	Structure Review, Parameter Review, Independent Elicitation & Corroboration
Architectural Walkthroughs	Qualitative	Model Assumptions Review, Model Design Review, Architectural Assumptions Review, Model Structure Review
Uncertainty Analyses (Qualitative)	Qualitative	Knowledge-base Review, Qualitative Risk Analysis, Data Acquisition/Management Audits
Developmental Reviews	Qualitative	Model Development Review, Operational Review, Developer Qualification Review
Uncertainty Analyses (Statistical)	Statistical	Bayesian Credible Intervals, Confidence Intervals, Volatility Analyses
Quality-of-Fit Analyses	Statistical	Fisher-F test, Student-T test, Kolmogrov-Smirnov test, Chi-squared test, Box-Cox test, Confusion Matrices, Receiver Operating Characteristic (ROC)
Generality Tests	Statistical	k-Fold Cross Validation, Leave-one-out Cross Validation, AIC, BIC
Generative Analyses	Statistical	Markov Chain Monte Carlo (MCMC) test-case generation analysis (Implicated Variables, Covariance Metrics [161])
Structural Analyses	Deterministic	Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Barren Nodes, d-separation, I-map
Sensitivity Analyses	Deterministic	Sensitivity to Findings, Sensitivity to Parameters, Parameter Bounds Checking
Robustness Testing	Deterministic	Model Fault Injection Performance Metrics, Adversarial Model Testing, Parameter Bounds Checking

5.4 Relationship of RM-BNS Objectives to Evidence

In Chapter 3, a set of verification and validation objectives for BNSs was introduced. These objectives represent *generic* objectives for the assurance of a BNS. As outlined in that chapter, they require refinement to *specific* objectives over the course of a BNS's development. Each objective addresses an assurance-related aspect of a BNS's design and development. For each objective, multiple BN analysis and evaluation techniques may be utilised to address it. By relating the RM-BNS objectives to BN analysis and evaluation techniques in this way, a generic mapping of objectives and evidence can be constructed.

To illustrate the concept of this mapping process, Table 5.3 shows an example mapping of the objectives to the techniques and associated items of evidence. The table maps a number of generic RM-BNS objectives to corresponding generic techniques and evidence. As with the objectives themselves, pre-emptively refining the specific variant of a technique or the metric that will be used as an item of evidence from a technique should not be attempted without consideration for the specific context of a given BNS and BN model, and should therefore not occur prior to the instantiation of a system-specific RM-BNS model.

Attempting to refine the mapping in Table 5.3 without this system-specific contextual information may prove to be misleading and inefficient. This is because the operational, developmental or technical context of a given BNS or BN model may preclude the use of any number of techniques for certain objectives: the exact implications of these contexts is unlikely to be known until development is well underway. For example, consider the mapping for objective MV-1.1:

"Establish and justify the basis for using structural variant for the [Model]."

In Table 5.3, this objective is mapped to four general groups of evidence items that could be used to support the satisfaction of this objective. These items may be produced by a range of techniques and may be used in various combinations as determined by the system-specific instantiation of an objective. As indicated, it may also be the case that for a system-specific objective, only a subset of techniques – and therefore items of evidence – will be available to BN developers.

It is important to note that the difficulty in generating evidence items given these techniques can vary dramatically: some items of evidence require a qualitative review by an expert, while others demand computationally intensive simulations and evaluations.

This of course raises the question: why not simply generate an item of evidence that is easiest to generate? As alluded to previously, BN developers and assurance practitioners must evaluate the confidence in the items of evidence generated by a technique. This will involve assessing the adoption of a technique based on the evidence classification (i.e. ETC, ERC) assigned to the evidence generated by that technique is assigned, as well as considering the criticality of the system aspect addressed by the objective an item of evidence is used to address. Assurance practitioners should not hold an objective addressing a highly critical system aspect as satisfied based upon the opinion of a system expert.

In general, assurance practitioners should favour *Direct*, *Deterministic* evidence in the first instance. However, as indicated in the previous section, many items of evidence generated during BN development will fall into the *Statistical* classification. Techniques that generate this evidence should be considered after *Deterministic* evidence, but prior to *Qualitative* evidence. Typically, it will be necessary to adopt multiple items of evidence to ensure comprehensive evidential coverage of an objective [70]. The selection of evidence should be performed in conjunction with the assessment of criticality. Example heuristics for utilising evidence based upon MCA-derived criticality metrics will be introduced in subsequent sections.

Table 5.3: Objective-Evidence Mapping for Model Viewpoint (*Structure View*)

Identifier	**Objective**	**Process**	**Evidence Item**
MV-1.1	Establish and justify the basis for using the structural variant for the [Model].	Model Walkthrough	Structure Review
		Uncertainty Analysis (Qualitative)	Knowledge-base Review, Qualitative Risk Analysis
		Quality-of-Fit Analysis	Fisher-F test, Student-T test, Kolmogrov-Smirnov test, Chi-squared test, Box-Cox test, Confusion Matrices, Receiver Operating Characteristic (ROC)
		Structural Analyses	Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Barren Nodes, d-separation, I-map
MV-1.2	Establish and justify the any assumptions in the structure of the [Model].	Model Walkthrough	Structure Review
		Uncertainty Analysis (Qualitative)	Knowledge-base Review, Operational Context Review
		Quality-of-Fit Analysis	Statistical Test Scores (e.g. Fisher-F, Student-T, Kolmogrov-Smirnov, Chi-squared)
		Structural Analyses	AIC, BIC, Barren Nodes, d-separation, I-map
		Architecture Walkthrough	System Design Review, Architectural Assumptions Review, Model Independence relations

Table 5.3: Objective-Evidence Mapping for Model Viewpoint (*Structure View*, continued)

Identifier	**Objective**	**Process**	**Evidence Item**
		Development Reviews	Model Development Review, Developer Qualification Review
MV-1.3	Establish and justify the implemented structure of the [Model].	Model Walkthrough	Structure Review
		Sensitivity Analysis	Sensitivity to Findings, Sensitivity to Parameters
		Dependency Analysis	I-map, d-separation
		Dev. Process Analyses	Model Assumptions Review, Developer Qualification Review, Modelling Setup Review
		Structural Complexity Analyses	Information Criterion Metrics (e.g. Akaike, Bayesian), Model Complexity Score (e.g. Fisher-F score)
MV-1.4	Establish and justify confidence in the structure of the [Model].	Model Walkthrough	Structure Review
		Architecture Walkthrough	System Design Review, Architectural Assumptions Review, Model Independence relations
		Dev. Process Analyses	Model Assumptions Review, Developer Qualification Review, Modelling Setup Review

To illustrate this point, consider the following two example BNSs:

System A - A BNS developed to reason about the population dynamics and ecological properties of a region (similar to the work of Pollino *et al*) [59].

System B - A BNS developed for information-discovery that models biological and chemical interactions in drug-discovery tasks, and that utilises a number of 'hidden' nodes within its structure to improve certain model performance attributes [40].

In the case of the first application (System A), it is typical for developers to have access to only very limited quantities of data: the development of BNs for ecological modelling applications is often mired in challenges associated with the absence of data on certain ecological variables. Information on the populations of certain fauna or flora, or other environmental data is often scarce [59]. In these cases, developers may be limited only to items of evidence derived from *Qualitative* processes. An example of a system-specific version of objective MV-1.1 could be generated for the model outlined in the work of Pollino *et al*, and would be defined as follows:

> *Establish and justify the basis for using a multivariate normal distribution for the 'Current Diversity of Fish Population' Random Variable.*

In this work, this variable was highlighted as a variable for which no quantitative information was available during development. It is therefore likely that expert evaluations – model walkthroughs – would constitute the majority of the evidence available for the satisfaction of this specific objective. This evidence would be classified as direct, qualitative evidence. This indicates that it may be particularly necessary to evaluate the *trustworthiness* of a given item of evidence in circumstances such as this, and to focus on establishing the diversity of expert opinion on the BN aspect. It also raises the consideration of whether or not an assurance practitioner should tolerate the inclusion of this model aspect in the first instance.

The second example BNS (System B) may present the opposite problem: the model generated entirely through parameter and structure learning approaches that minimise the ability of a human expert to pass judgement on the nature of the model in question. A variable within this model may have no semantic value and may be used only as a purely abstract mathematical modelling tool. For example, an example system-specific version of objective MV-1.1 for this application could be defined as follows:

Establish and justify the basis for using a NoisyOR distribution for the 'Latent Variable A' Random Variable.

In this circumstance, evidence derived from model walkthroughs may be impossible to produce, or otherwise uninformative; empirical testing, statistical analysis and mathematical proofs may therefore be the only route open to generating evidence to satisfy this objective. From an assurance perspective, both of these circumstances have relative advantages and disadvantages. In cases such as System A, assurance practitioners can gain a transparent understanding of a BN model from an expert's perspective. However, this will lack the mathematical rigour and statistical insights that are generally desirable for BNSs, and these items of evidence are likely to be exposed to the standard human biases and errors that commonly accompany expert elicitation activities [61, 116]. In the case of System B, the availability of data can ensure a stronger mathematical basis for the testing of the system's BN models, though the implications of the output of these techniques may be hard to translate into meaningful assurance insights.

5.4.1 Refining Objectives and Evidence

In the context of developing a mission-critical BNS, BN developers must therefore carefully consider what techniques and items of evidence are available to their specific system, and the implications of this information for the assurance of that system. In general, the diversity of applications and development contexts in which BNSs have been – and will continue to be – developed limits the availability or relevance of specific techniques and evidence for a given objective. In general, only a subset of possible techniques and their corresponding items of evidence will be used in the satisfaction of any given system-specific objective. A lower-level objective-evidence mapping is therefore unlikely to be meaningful until the development of a BNS is well underway.

Establishing this subset of techniques and associated items of evidence requires the refinement of the mapping in Table 5.3 *in parallel with* the refinement of the RM-BNS objectives as outlined in Chapter 3 and the development of the BNS more broadly. The general process for this refinement is shown in Figure 5.2. Assurance practitioners will begin with the generic RM-BNS objectives provided in Chapter 3 and the framework outlined in this chapter. They will then iteratively refine this mapping over the course of the development of the BNS.

The previous section indicated how two example systems (Systems A and B) could be

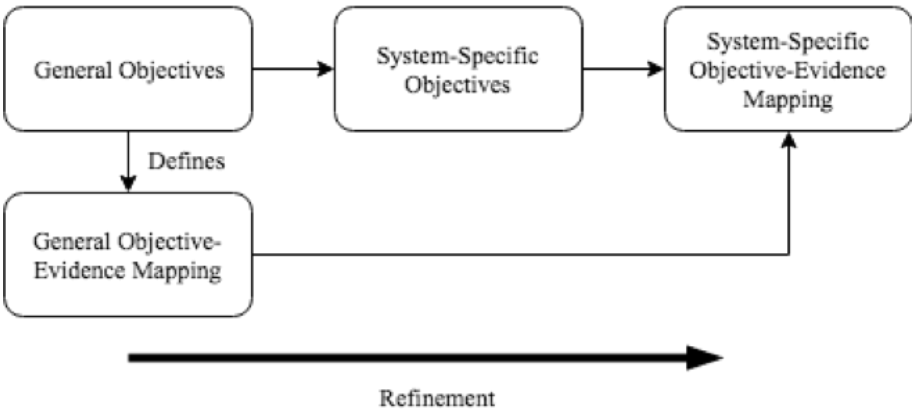


Figure 5.2: The objective-evidence refinement process.

associated with system-specific instantiations of the same objective (MV-1.1), and how the techniques associated with each of these instantiations may vary. By considering the context and technical basis of these objectives, BN developers and assurance practitioners can then identify *specific* technique variants and items of evidence that may be used to satisfy them. This can produce a refined mapping of objectives to their associated evidence items and techniques.

Table 5.4 shows an example of a refined, system-specific objective-evidence mapping for objective MV-1.1 in the context of Systems A and B. From an assurance practitioner's perspective, this mapping provides insight into what evidence is available, and by comparing the mapping to the technique-evidence mapping in Table 5.2 (specifically the evidence classifications), an understanding of how compelling this evidence may be can be obtained. The quality and integrity of the evidence can be described and assessed, but the sufficiency of this evidence must still be addressed. This can be achieved by understanding the *criticality* of a given objective.

5.4.2 Integrating Criticality Metrics

In Chapter 4, the Model Criticality Analysis (MCA) approach to analysing BN model aspects was introduced. This approach can be used to provide an assessment of the criticality of BN models and their components. As the metrics produced by MCA are associated directly with individual models and model aspects, these metrics can be mapped across to system-specific instantiations of the RM-BNS Model Viewpoint objectives. By extension, these metrics can be used to generate an objective-evidence mapping.

Table 5.4: Refined objective-evidence mapping.

Objective ID	**Objective**	**Processes**	**Evidence Item**
...	...	...	...
MV-1.1-A	Establish and justify the basis for using a multivariate normal distribution for the 'Current Diversity of Fish Population' Random Variable.	Independent Walkthroughs of Ecological Model of Fish Populations (Global) definition and scope, Independent Walkthroughs of definition and scope of 'Current Diversity of Fish Population' variable.	Expert Opinion
...	...	...	...
MV-1.1-B	Establish and justify the basis for using a NoisyOR distribution for the 'Latent Variable A' Random Variable.	Empirical testing of the Drug Discovery model (global), statistical analysis of 'Latent Variable A', Structural analysis of the NoisyOR distribution for 'Latent Variable A', parameter bounds checking for 'Latent Variable A'.	Statistical Metrics, Empirical Results, Mathematical Proof
...	...	...	...

For example, MCA could be used to assess the previously introduced BNS 'System A'. This would produce a Model Criticality Index (MCI) for the BN in System A, and a set of Variable Criticality Indices (VCIs) for each variable in the BN model. Each of these indices could then be mapped directly onto the RM-BNS objectives associated with the model aspect. Consider again the 'Current Diversity of Fish Population' Random Variable in System A's BN model. The application of the MCA methodology to System A's BN models will assign this variable a VCI. This VCI can then be assigned to *all* system-specific RM-BNS objectives that directly address this variable. Given the additional information from the MCA, this could be used to produce a mapping such as that shown in Table 5.5.

The consideration of additional criticality metrics for other system aspects associated with the remaining RM-BNS Viewpoints is beyond the scope of this chapter. However, there many existing approaches to establishing the criticality of software systems. For example, methodologies for establishing the criticality of software components are proposed in the defence standard MIL-STD-882E [167]. Other techniques for establishing the criticality of broader system aspects are provided or discussed in [94, 194] and [195, 196]. The Data Safety Guidance document also provides approaches to establishing notions of risk for data items [144]. Extensions to this guidance may enable its integration with the techniques presented here.

These techniques can be used to produce criticality metrics that can be mapped to other RM-BNS objectives, and by extension the associated evidence and evidence classifications. Figure 5.3 shows an updated representation of the refinement process for the objective-evidence mapping that includes the criticality analyses introduced here.

Table 5.5: Refined objective-evidence mapping with integrated criticality metrics.

RM-BNS Object	**Objective ID**	**Objective**	**Criticality**	**Process**	**Evidence Item**	**Type (ETC)**	**Role (ERC)**
...	...	...	...	...	...	...	
Current Diversity of Fish Population (RV)	MV-1.1-A	Establish and justify the basis for using a multivariate normal distribution for the 'Current Diversity of Fish Population' Random Variable.	VCI-4	Model Walk-through	Two independently elicited expert opinions on the use of the given statistical model to represent the variable.	Qualitative	Direct
...	...	...	...	...	...	...	

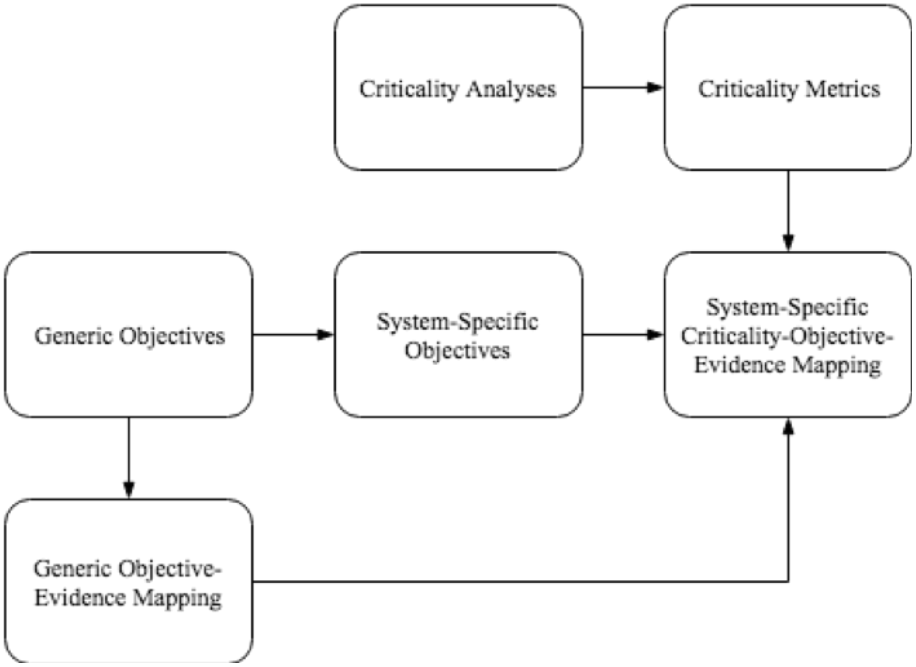


Figure 5.3: The objective-evidence refinement process with integrated criticality metrics.

5.5 Establishing the Sufficiency of BNS Assurance Efforts

Achieving 'absolute safety' for a given system is impossible. Assurance practitioners focus on establishing the safety of a system in terms of *confidence* in the safety of a system's behaviour. This is typically predicated upon establishing the *sufficiency* of the activities and processes carried out during the development of a system. As discussed in Chapter 2, strategies aimed at establishing confidence in a system can broadly be divided into two principal groups: those strategies that derive confidence from compliance with a given safety standard, and those that derive confidence through the development of argument-based assurance documents. In the context of BNSs, there are no existing standards that adequately address the considerations necessary to assure them – though as discussed there is extensive guidance on establishing confidence in the more conventional software aspects of a BNS. Similarly, while an argument-driven approach to assurance could (in principle) be more flexible, the absence of existing guidance on *what* safety practitioners should look for (and *why*), and techniques for the safety-focussed analysis and evaluation of BNSs, also renders these approaches potentially problematic.

The intention of this thesis has been to outline a structured framework for the description of BNSs, and an approach to addressing the BNS-specific assurance considerations

associated with these systems. The objectives introduced in Chapter 3 and the safety-focussed analysis technique introduced in Chapter 4 have been integrated in this chapter alongside an evidence classification framework based on best-practices in existing work. The resulting framework is aimed at providing the basic components for an approach to building up a comprehensive picture of the state of confidence in a BNS. Within the scope of the work presented here, the criticality of the RM-BNS Model Viewpoint aspects of a given BNS can be established and mapped to system-specific objectives.

In the software safety domain, the sufficiency of assurance efforts is often established with respect to some pre-defined set of goals or objectives. Commonly, the number and associated rigour of the objectives varies according to the risk associated with the operation of a given system. In the case of BNSs, the sufficiency of assurance efforts will be predicated on at least two factors: the variability of the objectives themselves, and the approach adopted to the satisfaction of the RM-BNS objectives.

5.5.1 Variability of RM-BNS Objectives

The properties of the techniques used to analyse and evaluate a BNS are atypical of conventional software analysis and testing techniques. Perhaps the most important distinction between BNS testing techniques and conventional software testing techniques is the heavy reliance on *Statistical* evidence produced during BNS testing. This has important consequences for the satisfaction of objectives dependent upon these techniques. For example, consider again the objective MV-1.1:

"Establish and justify the basis for using [structural variant] for the [Model]."

At a high-level, the motivation for this objective is to establish the necessity for a given *Model*, and the appropriateness of the selected structural variant. The satisfaction of this objective will regularly be predicated on providing compelling *Statistical* evidence of the properties of a given *Model* structure. For example, the ROC analysis outlined in section 5.3.3.1 could be used to compare the selected structure against alternative structures. This could be used alongside some form of CV testing to justify the selection of a given variant.

However, from a technical perspective, the satisfaction of the objective based upon evidence produced by these two techniques will not be a Boolean proposition: the depth of analysis and evaluation activities that may be performed to address an objective will often be variable. For example, in the case of CV testing, there are a number of 'levers'

that an assurance practitioner may control: they may choose to change the number of 'folds' in a dataset, the cross-validation variant, the measure-of-fit, or the target threshold for model performance, to name a few.[7]

Practically, CV testing will produce statistical evidence for the performance of a BN model. The confidence associated with this measure will vary as a product of these 'levers': more folds, different measures-of-fit or different model performance targets will modify the degree of confidence gained from this technique. However, it is rarely possible to define an *objectively* optimal CV testing configuration: for any given model there are generally an extremely large number of potential CV techniques or configurations available to developers. Consequently, the sufficiency of CV testing should only be established in the context of a given objective, the criticality of that objective and the *rationale* behind the use of the given CV testing variant or configuration.

These considerations are not limited to the techniques listed above. Many BN analysis and evaluation techniques require the exploration of extremely large parameter- or state-spaces or otherwise may have no objectively optimal test configuration. A simple example of such a method is the sensitivity analysis technique outlined in Chapter 4. For large models, the application of this technique will become computationally intractable. The question of depth is then a key consideration – it may not be possible to render the outcomes of a BNS test down to a yes-no answer *without* masking important assurance information. Another example is the selection of confidence intervals as outlined in the previous section of this chapter. Again, the optimal analysis will be heavily dependent on the system begin developed. This is a recurrent problem for many aspects of BNSs, particularly those associated with the Model, Data and Computational Viewpoints. Consequently, many of the RM-BNS objectives associated with these viewpoints are potentially exposed to a high degree of variability in the efforts required to satisfy a given objective.

5.5.2 Reducing Objective Variability

One approach to eliminating or reducing this variability would be to introduce fixed target metrics within (or associated with) the RM-BNS objectives. Many existing standards have pre-defined quantitative targets for systems developed in compliance with that standard

[7] In this context, the number of folds in a CV test refers to the number of sets of unseen data a model is tested over. Each of these folds is a subset of the dataset. A detailed review of CV testing for Machine Learning (ML) can be found in [63].

[69,94]. In the case of BNs and the RM-BNS objectives, it is technically *possible* to generate such metrics. However, the temptation to set pre-defined quantitative targets should generally be avoided for BNSs (and AISs more broadly). Conventional software systems are often developed in a top-down, sequential process with designs that can be largely defined at the outset. The development of a BNS is typically more of an iterative process of discovery and refinement. It is rarely possible to define *meaningful* quantitative targets for many aspects – particularly lower-level aspects – of a BNS *prior* to the development of the system.

Awareness of the *specific* properties, behaviours and context of a BNS is essential if any degree of confidence is to be derived from setting the objectives – or in this case, a given metric. Furthermore, in cases where it is possible to produce meaningful target metrics *a priori*, the wisdom of doing so is debated [77, 197]. From an assurance perspective, the 'dark art' aspects of many AI development practices demands that the metrics and evaluation techniques should be *closely* and *explicitly* coupled to the specific developmental and operational context of the BNS. In some ways, this could be considered to be an AI-centric manifestation of Goodhart's Law:

> *"When a measure becomes an objective, it ceases to be a good measure."*

For example, an obvious target metric may be to define a minimum level of statistical confidence for BN model aspects for varying levels of model criticality. However, the dynamics of a specific BN model may render such metrics highly inefficient: the topology of the model may make it highly insensitive to statistical error, thereby making the target values excessively conservative [171]. Alternatively, the model's topology may make it extremely sensitive to certain forms of statistical error, which may in turn make the target values too lenient [190]. There are documented cases of these issues in the BN literature [2, 190]. This applies to multiple aspects of an AIS development lifecycle. This has been noted in parallel contexts in Google's work on AI safety [21]. The satisfaction of a given objective with respect to a metric should therefore only be determined when the full context of a model is known.

5.5.3 Varying Objectives

Beyond the variability of the objectives themselves, there remains the challenge of adopting an approach that demonstrates the overall satisfaction of the RM-BNS objectives. Many

industry standards adopt a similar approach to establishing the sufficiency of assurance activities: they offer a set of objectives for a given class of system and vary the quantity of objectives in accordance with the criticality of the system in question.

For example, DO-178C mandates that developers satisfy 71 objectives for systems developed to the highest Development Assurance Level (DAL), while requiring that only 26 objectives must be satisfied for the lowest safety-related (i.e. system aspect with safety effect) DAL. The *independence* of these objectives also varies across the levels.[8]

In the case of DO-178C, the DALs can be seen to address Principle 4+1 of the Software Safety Principles and therefore offer one approach to solving the problem of establishing sufficiency in assurance efforts for a given system. This suggests the first conceptual approach to establishing the sufficiency of assurance activities in the context of BNSs: the number of assurance objectives may be varied in accordance with the criticality of the system aspect associated with that objective. This can be achieved through the satisfaction of only a subset of the RM-BNS objectives as determined by the criticality of the BNS aspect addressed by a given objective. For example, at the lowest levels of criticality a small subset of objectives may be satisfied, while at the highest levels of criticality all objectives should be satisfied.

Support for this approach was considered when the RM-BNS objectives were defined. They have been defined to address aspects of a BNS such that the objectives can be considered become progressively higher-level and build upon preceding objectives. Consider objective MV-1.1:

MV-1.1 - Establish and justify the basis for using the structural variant for the [Model].

This objective is defined to address low-level considerations that are likely to arise early in a BNS's development. It addresses the utilisation of a given model or variable in the abstract, potentially *before* a model has been constructed. Using the language of the work of Przytula *et al* (as outlined in section 2.2.4), this objective targets the modelling activities carried out as part of development Stages 2 and 3 (Subsystem Definition and Subsystem Modelling, respectively). In these stages individual variables are selected and defined by developers. The *probabilistic structure* of the variable is also commonly defined

[8] In this context, 'independence' refers to the independence of the software verification and validation activities from the development teams. It is assumed that this separation ensures the 'objectivity' of these processes.

at this point (i.e. prior to integration with the complete BN model). From an assurance perspective, it is defined to *explicitly* capture and document lower-level modelling decisions that may influence confidence in a model at later stages of development.

However, for low-criticality system aspects, it may not be necessary to satisfy this objective. For such systems, these low-level assurance activities may be deemed superfluous – provided the BN model/s still perform well overall on higher-level tasks. This is the aim of objective MV-1.4:

MV-1.4 - Establish and justify confidence in the [structural variant] of the [Model].

This objective addresses the need to establish overall confidence in a system aspect: in this case the structure of a model. The three preceding objectives are aimed at ensuring development steps, design decisions and evaluation activities that specifically address those system aspects are explicitly documented and reviewed, and this leads naturally into the final objective. By satisfying only MV-1.4, developers would leave activities and decisions associated with the other objectives implicit, undocumented or simply unperformed, but could still demonstrate that confidence in a structure has been achieved by appealing to higher-level model performance metrics and other evaluation approaches.

Such an approach would bring the objectives into approximate alignment with the evaluation activities of a high-standard existing BN development programme [6,52]. These programmes typically look to model-level evaluation metrics and design decisions and do not generally tackle the model's performance and characteristics on a more granular level. This seems appropriate for models associated with the lowest levels of criticality. In contrast, the higher levels of criticality demand a more granular, rigorous enumeration of design decisions, model evaluation and integration activities. Technical verification and validation of a BNS at these levels is atypical within the existing literature. These aspects are tackled by objectives addressing lower-level considerations (in this example typified by objectives MV-1.1 to MV-1.3).

For example, considering only the RM-BNS Model Viewpoint, it would be possible to use this general approach to map objectives to model criticality metrics. An example of how this could be achieved is shown in Table 5.6. A summary of the total number of objectives is shown in Table 5.7. As with DO-178C, at the highest levels of criticality, all objectives should be satisfied, and the total number of objectives then declines alongside the criticality. This should be extended to include similar notions of verification

Table 5.6: A suggested breakdown of how the number of compulsory Model Viewpoint RM-BNS Objectives may be varied as a function of the criticality of a system aspect.

Crit. (MCI/VCI)	**View**	**Objectives (MV)**	**No. Objectives**
0	Structure	1.1, 1.2, 1.3, 1.4	4
	Parameters	2.1, 2.2, 2.3, 2.4	4
	Definition	3.1, 3.2, 3.3, 3.4	4
	Dynamics	4.1, 4.2, 4.3, 4.4, 4.5	5
1	Structure	1.2, 1.3, 1.4	3
	Parameters	2.2, 2.3, 2.4	3
	Definition	3.2, 3.3, 3.4	3
	Dynamics	4.2, 4.3, 4.4, 4.5	4
2	Structure	1.3, 1.4	2
	Parameters	2.3, 2.4	2
	Definition	3.3, 3.4	2
	Dynamics	4.3, 4.4, 4.5	3
3	Structure	1.4	1
	Parameters	2.4	1
	Definition	3.4	1
	Dynamics	4.4, 4.5	2
5	Structure	None	0
	Parameters	None	0
	Definition	None	0
	Dynamics	4.5	1

Table 5.7: Total number of suggested compulsory Model Viewpoint RM-BNS Objectives.

Criticality	**Viewpoint**	**No. Objectives**
0	Model	17
1	Model	13
2	Model	9
3	Model	5
4	Model	1

and validation independence as set out in DO-178C: at the highest levels of criticality a significant proportion of the RM-BNS objectives should be satisfied with independence from the development team. As with the rest of this chapter, while the discussion here has focussed on the Model Viewpoint RM-BNS objectives, the approach can be applied to all objectives across all viewpoints.

5.5.4 Varying Evidence

A second possible approach to establishing the sufficiency of assurance activities for BNSs is to adopt a strategy in which the confidence in the evidence used to satisfy an objective varies. Adherents of this approach would therefore explicitly address every RM-BNS objective for a given system but would vary the quality and/or quantity of evidence used to satisfy a given objective as a product of the criticality of the system aspect the objective addresses. An approach that explicitly varies the evidence with respect to the criticality of an objective would be in closer conceptual alignment with standards such as DS 00-056 in which the emphasis is placed upon assurance practitioners to justify the sufficiency of their assurance efforts with direct reference to the system at hand and the techniques used [97].

The contributions of this chapter could be used to facilitate this approach: the proposed evidence classifications and objective-evidence mappings could be used to provide a structured basis for rationalising the sufficiency of the evidence addressing a given objective. Assurance practitioners can target those models and model components directly according to the respective criticality of those aspects, and describe the role and type of evidence in this context. Using the previous approach (varying the number and independence of objectives), assurance practitioners may choose to omit this objective from consideration for low-criticality system aspects. However, as discussed throughout this chapter, it is important that the rationale for the use of a given technique with respect to a given objective is made explicit. By instead varying the confidence in the evidence used to satisfy an objective, assurance practitioners can provide explicit, end-to-end rationale for the design, evaluation and testing decisions made during the development of a BNS.

While this confidence in the evidence used to satisfy system-specific objectives will vary across each BNS, it will ensure that each system aspect addressed by the RM-BNS objectives is explicitly considered by assurance practitioners. This should encourage a more critical evaluation of the use of evidence in the development of a BNS and avoid

the 'masking' of uncertainty and decision making that may become a problem with the previous approach.

From a practical perspective, this would produce a situation in which the confidence in the evidence used to satisfy a given objective is proportional to the criticality of the objective the evidence addresses: an objective pertaining to low criticality system aspects may require little evidence, or evidence with significant aleatoric or epistemological uncertainty, while the same objective addressing a higher-criticality system aspect would demand more evidence and less uncertainty in that evidence.

A further practical benefit of this approach may also be its ability to adapt to the highly dynamic, iterative lifecycles common to BNSs. Specifically, in the event the criticality of a system aspect changes in the presence of modifications to a model's design, or the role of data in the system, for example, this approach would allow assurance practitioners to revisit and modify the existing evidence rather than be forced to adopt new evaluation techniques and develop new evidence. An approach that varies the confidence in the available evidence could be seen to encourage the utilisation of a single test-harness for a given BNS, but in which the rigour and depth of the analyses of system aspects varies proportionally with the criticality of the system aspect being evaluated.

5.5.5 Satisfying RM-BNS Objectives

To illustrate how these concepts may be applied to a BNS, consider the following example. Within the BN literature, 'common wisdom' suggests that small errors in the parameterisation of a model do not seriously influence the output of the model, and therefore BN developers need not focus much effort on evaluating the accuracy of individual parameters within a BN model [171]. However, others have noted that while this may generally be true, small variations in the parameters of a model can produce dramatic variations in the outputs of a model [198]. The RM-BNS Objective MV-2.2 is aimed at explicitly addressing this issue:

"MV-2.2 – Establish and justify the accuracy of the parameterisation of [Model]."

From an assurance perspective, the satisfaction of this objective will provide assurance practitioners with confidence that the dynamics of a given BN model are robust to errors in the parameterisation of the system aspect addressed by the objective. To achieve this, BN developers and assurance practitioners will typically rely on a range of statistical analyses,

mathematical proofs and expert-elicited information.

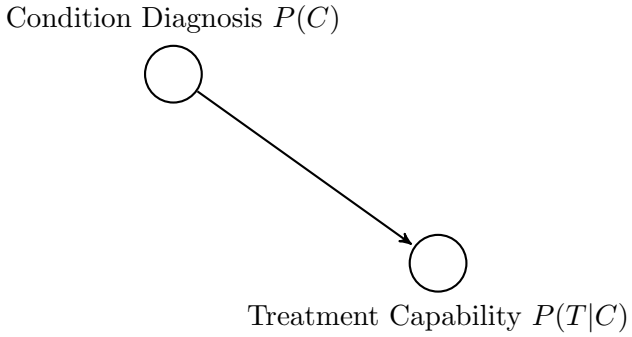


Figure 5.4: A fragment of System C's medical diagnosis BN model.

Consider a third BNS: System C. System C provides diagnosis and prognosis capabilities for critically ill patients and can administer drugs to patients if certain medical conditions are detected and diagnosed. Figure 5.4 shows an example fragment from a BN model that is used to drive the system. The figure shows two variables: The *Condition Diagnosis* and *Treatment Capability* RVs. The latter variable represents an output from this model that is used to inform System C to administer drugs to a patient, while the former represents an internal diagnosis variable that identifies specific medical conditions. The system-specific refinement of MV-2.2 for the *Condition Diagnosis* CPD would then be defined as follows:

> *"MV-2.2-C – Establish and justify the accuracy of the parameterisation of the Condition Diagnosis RV."*

This objective can be mapped directly to a criticality metric using the MCA technique introduced in Chapter 4. The aim of assurance practitioners will then be to satisfy this objective such that the confidence established is proportional to the criticality of the system aspect. In the context of the model in Figure 5.4 and the above objective, the primary aim of this objective is to address the need for assurance practitioners to demonstrate that errors in the parameterisation of the *Condition Diagnosis RV* do not affect the output of the *Treatment Capability RV*.

5.5.5.1 Evaluating Statistical Confidence

One strategy that BN developers and assurance practitioners may adopt to tackle this challenge may be to analyse the statistical confidence in the parameters learned by the

model. Consider a simple scenario in which BN developers have access to one hundred patient records pertaining to a particular condition. Of these records, 33 are diagnosed as having the condition, and 67 are not. A simple BN parameter-learning algorithm would then assign the probability of a patient having the condition as a 33% chance, and of the patient not having the condition as a 67% chance.

However, this is an *estimate* of the frequency of the occurrence of the condition. It is drawn from a small sample of individuals, and other samples may produce different estimates. For cases such as this, it is therefore useful to evaluate the *statistical confidence* in the learned parameter and estimate upper and lower bounds on the plausible values of the parameter. A common approach for estimating these bounds is to compute the so-called Wilson Confidence Intervals [199]. These bounds are given by:

$$\frac{p_0 + \frac{t}{2}}{1+t} \pm \frac{\sqrt{p_0 q_0 t + \frac{t^2}{4}}}{1+t} \tag{5.1}$$

Where p_0 is the rate of occurrence of some observation, q_0 is the rate of the non-occurence of some observation, and t is defined as:

$$t = \frac{z^2}{n} \tag{5.2}$$

Where z is the probit function for a specific confidence interval, and n is the total number of overall observations. This establishes bounds on either side of a parameter estimate.

These confidence intervals can be set based upon the tolerances of assurance practitioners and the BNS in question. A standard confidence interval is the 95% interval, though other common choices include 75%, 90%, 99% and 99.9% respectively. This interval can be considered as defining the range within which the 'true' value of a parameter is likely to exist – the wider the interval the greater the statistical confidence that a parameter falls within that range. Table 5.8 shows these confidence intervals computed for the *Condition Diagnosis RV* parameter estimate using Equation 5.1.

It is also common to vary these intervals, and assurance practitioners may choose to do so based on the development context of a given model and the differing criticality assignments associated with the particular parameter estimate.[9] Specifically, this can be

[9] A discussion on the use and interpretation of confidence intervals in statistics is beyond the scope of this chapter, though a good introduction can be found in [20].

extended to the concept of criticality – higher criticality system aspects would be associated with higher statistical confidence. An example mapping of confidence intervals to MCA criticality metrics is also shown in Table 5.8. However, in practice the precise confidence intervals used should be carefully considered in the context of the BNS in development and only selected based upon the specifics of the system as discussed in section 5.5.2.

Table 5.8: Example mapping of Variable Criticality Indices (VCI) to statistical confidence intervals for System C's model fragment.

VCI	**Interval**	**Parameter**	**Lower Limit**	**Upper Limit**
4	75.0%	67.0%	61.4%	72.1%
3	90.0%	67.0%	59.0%	74.2%
2	95.0%	67.0%	57.3%	75.4%
1	99.0%	67.0%	54.2%	77.7%
0	99.9%	67.0%	50.5%	80.1%

In the case of System C, this approach could be adopted to select the 95% confidence intervals for the parameters of the Condition Diagnosis CPD. As can be seen in Table 5.8, this would produce a lower confidence bound of 57% and an upper confidence bound of 75%. This can be considered as indicating that there is high statistical confidence that the 'true' value of this parameter falls within the range 57-75%. This could be used to provide *Direct*, *Statistical* evidence of establishing the accuracy of the parameters, thereby partially fulfilling the objective.

However, there remains the need to justify this level of accuracy. One approach to addressing this second aspect of the objective could be to utilise the confidence intervals to evaluate how the dynamics of the model change over this range of plausible values, and whether the model outputs are sensitive to these changes. If they are, this may indicate assurance practitioners may need to take more measurements and gather more records to improve the statistical confidence in these parameter estimates, or perhaps modify the model itself to ensure its robustness to these errors.[10]

[10]This example is of course for an extremely simple BN model, and therefore other techniques may be necessary to hold this objective as satisfied for more complex models.

5.5.5.2 Analysing Parameter Bounds

A technique that could be used for this purpose was introduced by Chan *et al* [190]. This approach allows BN developers to evaluate the dynamics of a BN model in order to establish the robustness of the outputs of a model to variations in parameters within the model. The previous statistical analysis can be utilised to guide this analysis. By demonstrating that a model (or in this case: model output) is insensitive to changes in the parameter over the range identified previously, practitioners could justify that the accuracy of the parameters obtained is sufficient for the system.

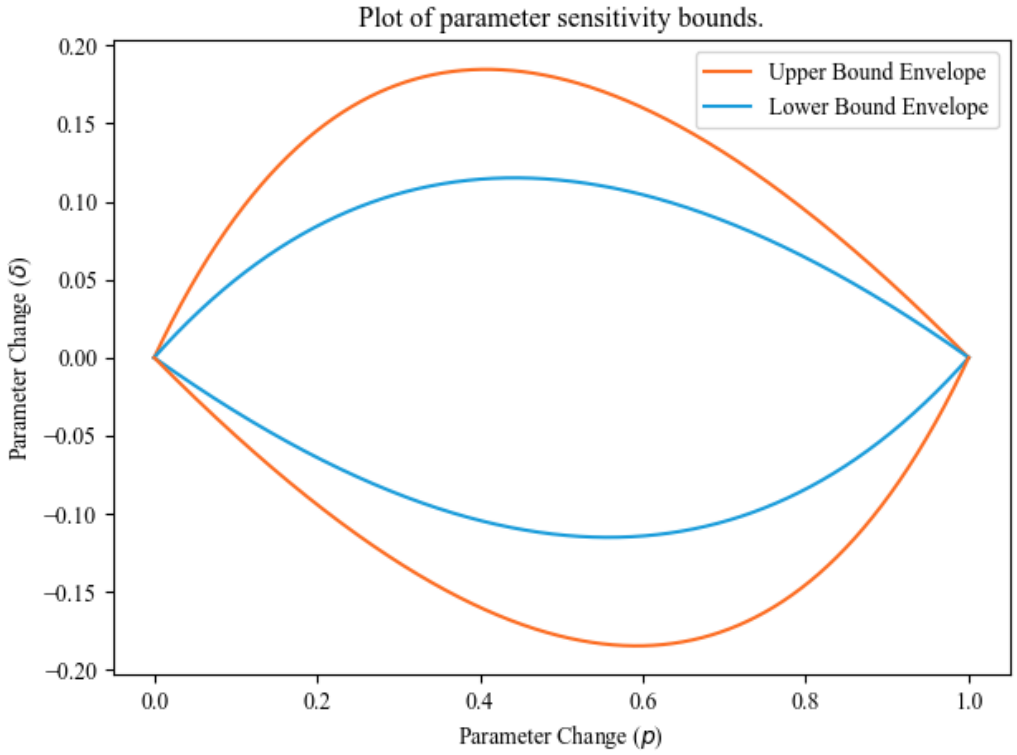


Figure 5.5: An example of the output from Chan's parameter bound analysis technique.

An example output from Chan's technique is shown in Figure 5.5. The figure shows how the degree to which the value of a parameter (p) in the *Condition Diagnosis RV* can be allowed to change (δ) as a function of the value of that parameter while maintaining the desired output properties for the model (in this case the *Treatment Capability RV*). The inner 'envelope' guarantees that the output will remain above a specified lower bound, while the outer 'envelope' guarantees the output will remain below a specified upper bound.

Concretely, the figure can be used to indicate the range of possible changes to the parameters of the *Condition Diagnosis RV* that will *not* affect the output of the model.

Technically, Figure 5.5 allows practitioners to deduce that for the *Condition Diagnosis RV* parameters to modify the *Treatment Capability RV* outputs, the value of the *Condition Diagnosis RV* parameters must lie either below 50% or above 81%. This is *Direct*, *Deterministic* evidence for the satisfaction of this objective.

When this information is combined with the statistical analysis introduced earlier, an assurance practitioner can establish that the 95% confidence intervals for the parameter lie well within this range. Indeed, the 99.9% confidence intervals also lie within this range, suggesting that the evidence for the accuracy of the parameterisation of the variable would be acceptable even at high criticality levels.

Practically, this can indicate to an assurance practitioner that the *Treatment Capability RV* SRV outputs of the model would not change over the entire range of plausible parameter values for the *Condition Diagnosis RV*. This directly justifies the established accuracy of the parameterisation of the *Condition Diagnosis RV* and may thereby satisfy the objective. Taken together, these two techniques can provide complete evidential coverage of the objective in this specific context. In other circumstances, one or both of these techniques may not be practicable, or may not provide complete coverage of the objective.

5.6 Conclusion

The primary aim of this chapter has been to provide guidance on the considerations and strategies an assurance practitioner must consider when developing and evaluating a mission-critical BNS. The highly iterative design and development activities and statistical properties of these systems introduce a number of novel assurance concerns that must be carefully considered and addressed by assurance practitioners.

In particular, this chapter has focussed on establishing a framework for describing and classifying evidence used in the assurance of a BNS. This framework draws upon existing work in the safety domain to build a single framework that can facilitate the description of any safety evidence generated during the development of a BNS.

The chapter then proceeded to outline the utilisation of this framework in the context of the RM-BNS objectives. Specifically, it used the proposed evidence framework to show how individual items of evidence may be mapped explicitly to individual system-specific objectives, and that through consideration of the characteristics and classifications of this evidence, assurance practitioners can begin to address the sufficiency of the evidence.

The final section focussed on potential strategies for demonstrating the sufficiency of

BNS assurance efforts. Both strategies essentially focussed on providing a mechanism for addressing the 4+1 software safety principle by varying either the number of objectives (in a manner similar to DO-178C) or varying the confidence in the evidence used to satisfy the objectives (more akin to DS 00-056).

The contributions of this chapter are therefore as follows:

- It has provided a framework for characterising and classifying evidence generated for the assurance of a BNS.
- It has introduced an approach to mapping individual objectives to items of evidence and criticality metrics built upon this framework.
- It has outlined an approach to the refinement of objective-evidence mappings over the course of the development of a BNS.
- It has provided guidance on approaches that may be used in establishing the sufficiency of assurance efforts undertaken during the development of a BNS.

Chapter 6

Evaluation

6.1 Introduction

The Thesis Hypothesis as introduced in Chapter 1 was given as follows:

> *It is possible to provide targeted assurance for BN-based Systems through the analysis and evaluation of underlying probabilistic models, data artefacts and computational techniques that have the potential to affect confidence in the safety of a system.*

This hypothesis was defined with the principal aim of addressing the assurance concerns associated with the motivating use-case of this project: the utilisation of an advanced Bayesian Network-based Prognostic Health Monitoring (PHM) system aboard an Unmanned Aerial System (UAS) in mission-critical applications. While this has been the motivation behind this project, the concepts, language and techniques presented in this thesis have been generalised wherever possible to accommodate a broader range of possible Bayesian Network-based System (BNS) variants and architectures than are likely to be encountered in this specific application. In accordance with this hypothesis, the principal contributions of this thesis are as follows:

- A generic, comprehensive framework for describing and modelling BNSs that also aims to facilitate the communication of the many unconventional aspects of these systems to both BN developers and safety practitioners;
- A set of generic verification and validation objectives that provide a flexible, comprehensive framework for the verification and validation of BNSs, and expose safety considerations that are poorly addressed in existing standards;

- A systematic, generic approach to establishing the criticality of model-centric aspects of BNSs that integrates safety information with BN model analysis techniques; and
- Guidance on establishing the sufficiency of BNS assurance efforts through the extension of existing assurance concepts to the techniques and concepts introduced throughout the thesis.

Due in part to the sensitive nature of the use-case of this thesis, the application of these techniques to a 'real-world' system has not been possible over the course of the project. This chapter utilises a combination of public-domain resources to construct a case study that outlines the application of the techniques and concepts introduced in this thesis and supports the evaluation of these contributions in the context of this research project. Insights drawn from this case study are then extended to evaluate the scope and utility of the contributions of this thesis in the context of 'real-world' BNSs, and in the context of the safety domain more generally.

The chapter is structured as follows: a case study is provided that applies the concepts and methodologies introduced in previous chapters to an example system based upon existing systems within the BN literature; the contributions of the thesis are then evaluated in the context of this case study with respect to the thesis hypothesis; the contributions are then further evaluated with respect to the scope of the thesis and its motivating example; and finally, the contributions will be evaluated against existing standards and BN development practices to explore what has been gained through the contributions of this thesis.

6.2 Case Study

This section looks to the existing BN literature to introduce a BNS that represents an amalgamation of existing concepts, architectures and models that showcase the utility and application of the contributions of this thesis in a more tangible setting.

6.2.1 System Definition and Methodology

Several candidate systems and development approaches were reviewed for use in this case study. The system presented here represents an combination of multiple distinct BNSs and corresponding BN models that are drawn from a range of domains [6, 59, 200]. The system has been defined to represent a BNS that is used for medical diagnosis and prognosis

alongside the delivery of a limited set of medical treatments – principally the administration of drugs to patients. It is assumed that this latter aspect of the system includes direct responsibility for the *unsupervised* administration of drugs to a patient, and that incorrect use of these drugs could have serious consequences for the health of a patient.

There is a precedent for this type of device [201]. There are existing software-intensive medical devices that use software to moderate the administration of drugs to patients. A typical example of a safety-critical software-based medical device is a system dedicated to automatically dispensing insulin to diabetic patients [201, 202]. The system devised for this case study is intended to reflect an extrapolation of this form of software to one that utilises BN models in place of conventional control logic when treating patients. Practically, such a system could support the diagnosis and treatment of a more diverse set of medical conditions and scenarios.

For the purposes of this case study, the system is defined to enable the provision of only two forms of treatment: the administration of anaesthetic/pain management drugs, and the administration of drugs to combat anaphylactic shock. Both of these treatments represent plausible treatments given the scope of the BN models upon which the system is based [136]. The example system will be referred to throughout the case study as the Autonomous Monitoring and Treatment System (AMTS).

6.2.1.1 Source Material

The AMTS system architecture is based upon architectures drawn from public-domain research and documentation. The systems upon which the AMTS is based were selected based upon three criteria:

- The system must have been deployed in a 'live' operational environment.
- The system must not be a 'toy' research system – it must be sufficiently complex that it illustrates the application of the methods introduced in this thesis.
- The system must have a *direct* application to a complex diagnosis or decision making problem.

The system architecture presented here is based principally on two system architectures. The first is that outlined in the work of Przytula *et al* on the development of diagnostic support systems for satellites [6]. The second draws upon the work of Velikova

et al who looked into the development of a medical prognosis system for use in medical applications, specifically in monitoring pregnancy disorders [35]. As discussed in Chapter 2, the former work outlines an approach to developing the architecture of BNSs in fault diagnosis roles that involves the utilisation of multiple distinct BN models to support the diagnosis of specific subsystems. The latter work outlines an approach to using more novel model architectures for patient prognostics and provides an overview of the integration of a BNS into a broader treatment platform.

Finally, due to the constraints of this chapter, only a single BN model is discussed and analysed in detail. The model selected is the widely-used Alarm ICU model developed by Beinlich *et al* that was introduced in Chapter 3. As discussed in that chapter, this model was used to support the monitoring of patients in a hospital's Intensive Care Unit (ICU) ward [136]. From a technical perspective, this model was selected as it represents a good 'baseline' BN model: it consists of only discrete, table-structured CPDs and it is a low-to-medium complexity model.[1] From a practical perspective, it was selected as the model is publicly available, and reflects a complex, near-safety-critical diagnosis problem and is relatively ubiquitous in the BN domain. A graphical representation of the ICU model is shown in Figure 6.1. A copy of the exact structure and parameterisation used is presented in Appendix B.

[1]This model is commonly used for benchmarking various BN learning, inference and modelling approaches for precisely this reason: it is a practically useful model with sufficient complexity to test algorithms and approaches, but also sufficiently transparent to support meaningful analysis.

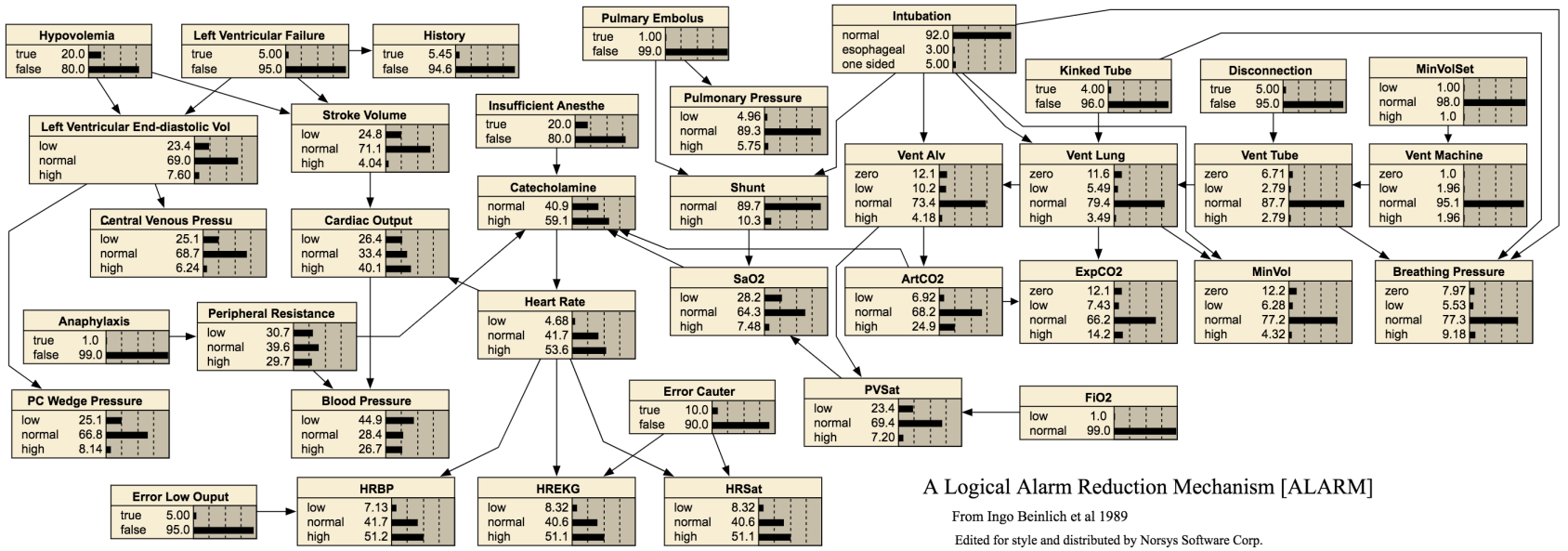


Figure 6.1: A visualisation of the Alarm ICU network and its local CPD structures. [136]

6.2.1.2 Assumptions and Scope

The limitations in the scope and detail of the materials used to define the AMTS architecture force a number of assumptions to be made. The most significant of these are as follows:

- Models in the AMTS were trained on complete data using a simple (Bayesian) Maximum Likelihood Estimation (MLE) for parameter learning;
- These model structures were obtained through expert elicitation only, and no structure-learning algorithms were utilised [136];
- There are no adaptive/on-line learning aspects of the system;
- All models contain only discrete, table-structured CPDs (i.e. they are of the same form as Beinlich's ICU model);
- All models in the system are 'pure' BNs: there are no undirected or hybrid models, only Directed Acyclic Graphs (DAGs) [2];
- All models used in the system are static – there are no time-dependent models in the system;
- The system uses only exact inference algorithms (some form of Junction Tree algorithm);[2] and
- The system functions without direct oversight from a medical practitioner.

Operationally, the AMTS is defined as providing three primary functions: an alarm system for summoning medical practitioners; a BN-controlled drug administration system and an 'enhanced' patient health monitoring function (e.g. an AI-enhanced User-Interface that provides additional information on a patient's health – much like current Electrocardiograms (ECGs) but with additional analysis and prognosis capabilities). These three functions are carried out fully autonomously by the system. To achieve this, the AMTS utilises four distinct BN models in the provision of these functions. These models are as follows:

[2]This is a relatively strong assumption. Many practical applications of BNSs use some form of Markov Chain Monte Carlo (MCMC) or variational inference. For a technical introduction to approximate algorithms, see [51].

- Two models are used to reason over a diagnostic model of patients. These models provide redundant signals on whether to summon medical practitioners. These models will be referred to as *Alert Model A* and *B* respectively.

- A model for reasoning about a patient's health status. For the purposes of this case study, this model is Beinlich's ICU model. This model will be referred to as the *Diagnosis Model*. This model informs whether or not a patient should receive treatment from the AMTS.

- A model for integrating, processing and enhancing input data from medical devices and providing this to medical practitioners when required. This will be referred to as the *Data Fusion Model*.

6.2.2 System Description and Modelling

The first step in the application of the ideas presented in this thesis is the generation of a RM-BNS architecture model that captures the components of a given BNS and the corresponding interactions between these components. The definition of model objects proceeds according to the methodology outlined in Chapter 3. Figure 6.2 shows an example of a fragment of a RM-BNS architecture generated for the AMTS. The model shown in the figure is a high-level representation of the system. Even at such a comparatively high level of abstraction, the model captures a number of important interactions within the system.

For example, consider the 'ICU Ward' Environment object. As discussed in Chapters 2 and 3, BNSs are exposed to their environments in a manner atypical of conventional software systems. This exposure can produce errors in a system's functionality related to distributional drift in the *model*, systemic biases, or other errors in the *data*. The fragment in Figure 6.2 highlights the impact of the environment upon data acquisition activities and which data artefacts are produced by these activities.[3] For example, the fragment shows that all four BN models are exposed to the environment, how they are exposed, and that three of those models are directly influenced by a single data acquisition activity.

[3]Note that the diamond symbols discussed in Chapter 3 are used here to denote 'undeveloped' aspects of the model. In this case, they are simply used for conciseness.

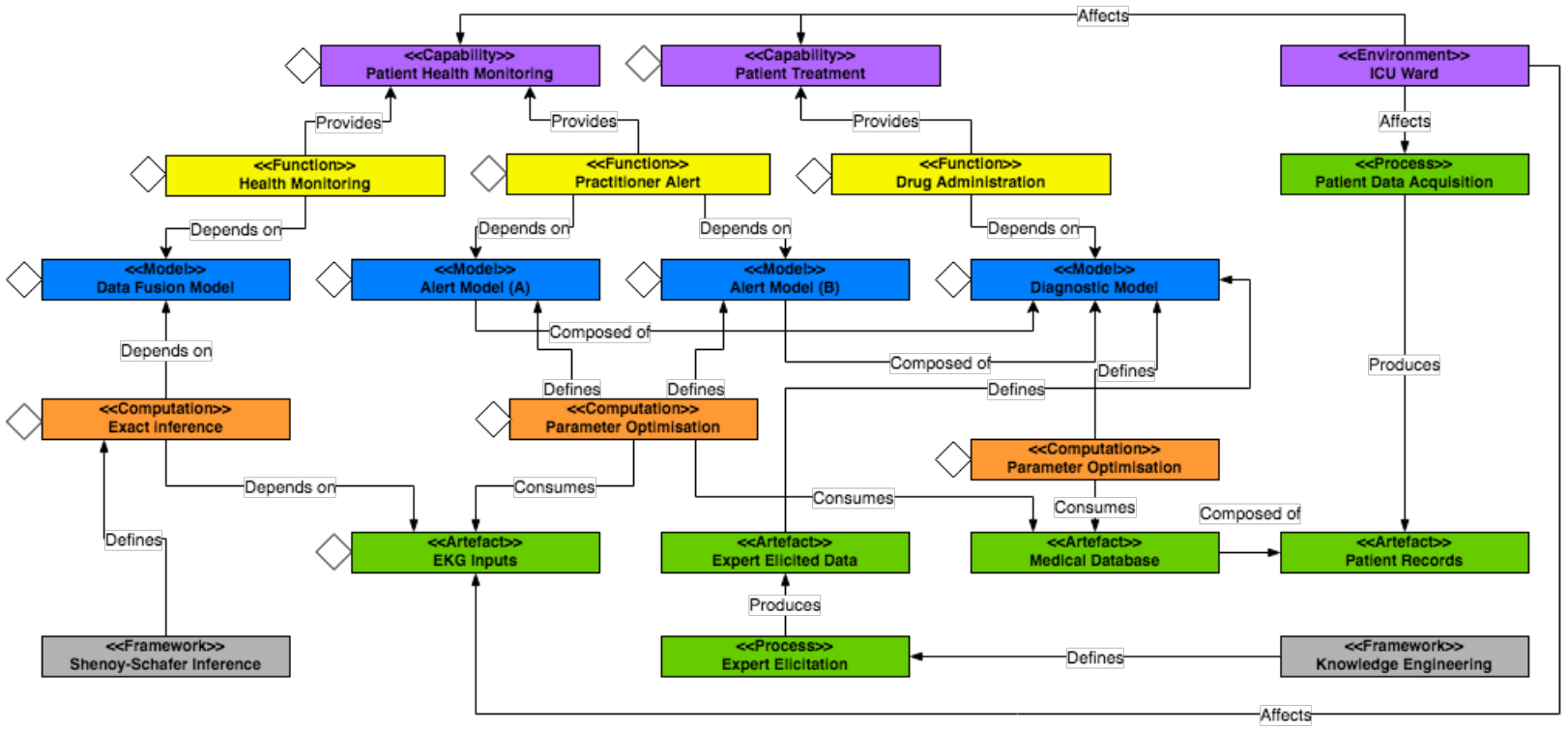


Figure 6.2: A high-level representation of (a fragment) of the AMTS architecture represented in the RM-BNS framework.

This suggests the system may be vulnerable to a data-centric equivalent of a common-cause failure. Examples of how such data-centric failures could occur may include stress-related errors on the part of medical practitioners in the acquisition process[4], or changes in operational processes on the ICU ward that effect the data acquisition process for the BNS. If the BN model is 'unaware' of these changes or potential errors (i.e. it has not been designed to accommodate changes or mitigate errors), the system will continue to function – from a conventional software perspective – normally. However, its diagnostic accuracy may *gradually* decline and this may erode safety margins. In this sense a BNS is likely to be less 'brittle' than a conventional software system, but this may also mask more subtle errors in the system's functionality. The RM-BNS fragment in Figure 6.2 explictily captures the interactions and dependencies between these aspects precisely so BN developers and assurance practitioners can discuss and analyse the implications of these interactions.

To cast these concepts in terms of the motivating example of this thesis, differences in the habits or practices of maintenance teams on different shifts or across different geographical regions may produce data of varying degrees of integrity or with different underlying patterns or properties [64, 116, 121]. This may include producing data that is in some way biased towards the environment in which they are working, or other operational differences between teams. For example, significant variations in temperature, precipitation, humidity or altitude may produce different failure profiles for an autonomous vehicle. A disproportionate or uncharacteristic failure rate may skew a BN model towards erroneous behaviour if it is not identified by practitioners and/or mitigated by the BN model's design.[5] Chapters 2 and 3 provide a more detailed overview of the potential effect of environmental changes on BNSs.

A second consideration indicated by Figure 6.2 is the degree of exposure of the system to expert-elicited data; the corresponding role of the 'Knowledge Engineering' framework selected to produce this data; the interactions between inference algorithms, model frame-

[4]This may include cutting corners with medical measurements and observations.

[5]This may include uncharacteristically *low* failure rates, perhaps associated with data gathered early in a vehicle's lifecycle, during testing or other similar, idealised scenarios.

works, models and input data[6]; the utilisation and role of *specific* BN models in the provision of *specific* software functions; and, more generally, the role of computational aspects of the system in producing and facilitating the utilisation of these models. From an assurance perspective, this introduces a degree of traceability into the architecture of the AMTS: an assurance practitioner can see – at a high-level in this case – the interactions between system aspects and potentially also the relative importance of those aspects on certain system behaviours and capabilities.

For subsequent analysis and objective-generation activities, lower-level representations generated using the RM-BNS will be necessary. Figure 6.3 shows an example RM-BNS fragment for the system aspects associated directly with the *Drug Administration* function and *Patient Treatment* capability. This fragment exposes a further set of interactions that may also be of interest to assurance practitioners. For example, the fragment highlights more precisely how the *Diagnosis Model* is used by the system. It indicates the dependence of this model on specific data sources, and once again highlights the role of the environment in the optimisation and structure of the model, and ultimately in the provision of the *Patient Treatment* capability. Furthermore, the fragment highlights the importance of the *Diagnosis Model* and *Parameter Optimisation* objects in the provision of the *Drug Administration* software function. Qualitatively, these two functions are associated with far more interactions with other objects than any other objects in the model. This highlights the centrality of these system aspects in the architecture of the AMTS.

[6]This can be seen in the fragment with the interactions between the *Data Fusion Model*, the *Exact Inference* object and the *EKG Inputs* object. In this case, the model makes explicit the dependency of the *Exact Inference* algorithm upon (the structure of) the *Data Fusion Model* and the *EKG Data*. For a discussion of the role of models and data in BN inference algorithms, see [1, 2, 60].

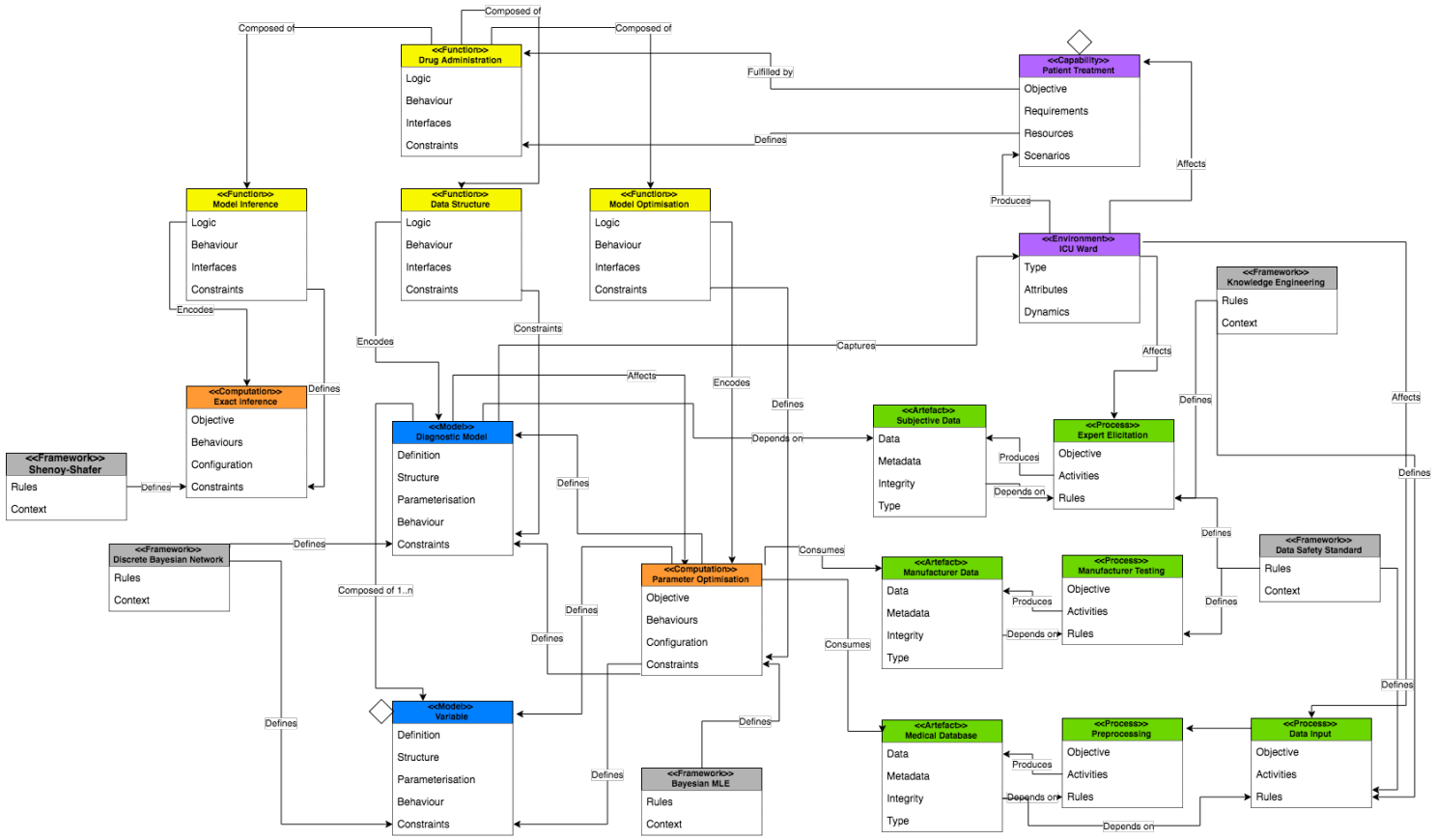


Figure 6.3: A lower-level RM-BNS fragment capturing system aspects associated only with the *Diagnosis Model*.

Finally, as discussed in Chapters 2 and 4, BN models can be considered to be models composed of a set of smaller models in a *factorised* representation. The RM-BNS approach is intended to reflect this. In Figure 6.3, a sub-model has been broken out from the *Diagnosis Model* object. In this case, this object is generically titled 'Variable', and the RM-BNS fragment indicates that the parent model is composed of '1 to n' 'Variable' objects. Subsequent modelling activities should break this down further such that each variable within a BN model is explicitly represented as a 'Model' object in the RM-BNS architecture model. The resulting RM-BNS instantiation may then explicitly represent the internal structure of the BN model and expose this to qualitative analysis.

6.2.3 Definition of System-Specific Objectives

With an initial RM-BNS model generated, it is possible to begin generating system-specific objectives for the AMTS. A full enumeration of system-specific objectives is beyond the scope of this chapter, however. Instead, a selection of objectives have been provided for discussion. These objectives have been generated in accordance with the methodology set out in Chapter 3, and were generated specifically for the RM-BNS fragment shown in Figure 6.3 and deal with only Model and Data Viewpoint system aspects.[7] Table 6.1 shows the set of selected objectives.

These four objectives represent a small cross-section of a full set of objectives that would be generated for the AMTS: each object in the architectural model would have a set of objectives generated that addresses it. However, these four objectives have been selected as they highlight a number of important considerations that must be brought to the attention of assurance practitioners. For example, consider objective DV-2.2 in Table 6.1:

> *Establish and justify the necessity of the discretisation pre-processing technique applied to the Patient Record data.*

This objective addresses the need to ensure that all pre-processing activities that apply transformations to the *Patient Records* are absolutely necessary. A practical example of such an approach could be as simple as translating a patient's heart rate from a continuous-valued variable (e.g. a heart rate in the range of 40-200 beats-per-minute) into a discrete-

[7] The generic RM-BNS objectives can be found in Appendix A.

valued variable (e.g. low, medium or high).[8] As discussed in previous chapters, this approach is associated with information loss, and can potentially influence the performance of a BN model.

For example, in the case that a patient's heart rate is categorised as either low, medium or high, an assurance practitioner should be interested in *why* this scheme was selected – there may be performance or safety implications (both positive and negative) if a fourth category (e.g. very high) were to be introduced. This may *allow a model to learn* more subtle relationships, perhaps that 'very high' heart rates are much more likely than 'high' heart rates to indicate a certain disorder. However, it may also *reduce* the performance of the system by making a model excessively sensitive to the peculiarities of the data used to train it.[9]

In the context of this objective, assurance practitioners will need to demonstrate the practical utility of the discretisation scheme selected, and that the presence of this pre-processing step has desirable assurance implications (e.g. discretisation improves the performance of the system on *safety-related* tasks, and/or improves the interpretability of the system from a medical practitioner's perspective). Assurance practitioners will need to explicitly consider the effect of the information loss associated with the application of any discretisation approach to a continuous-valued dataset.[10]

[8]This process is often called binning. There is an extensive body of literature on strategies to perform various types of discretisation on data sets [52, 54, 61, 203].

[9]This is another specific example of the bias/variance trade-off. See Chapters 2 and 3 for more information, or [137, 145] for a review of *local* bias/variance considerations.

[10]Two of the most commonly used transformations that are applied to data used in BNSs are data discretisation and dimensionality scaling. The objective discussed here addresses the former transformation. The latter transformation is concerned with ensuring a form of consistency across data. In this context, 'dimensionality' refers specifically to the *units* of a measurement. Failure to maintain consistent dimensionality within a BN model may lead to erroneous reasoning by the model. Such errors in reasoning may be difficult to identify as BN models are often robust to errors in inputs of this kind. System developers may therefore satisfy this objective by demonstrating the need to use the scaling process to avoid a BN-specific manifestation of the dimensionality error that led to the infamous loss of the Mars Climate Orbiter (MCO) [204]. The nature of the assurance activities used to satisfy any two system-specific objectives that are derived from the same generic objective may therefore vary significantly. However, in this context, the objective is designed to address both cases equally.

Table 6.1: System-Specific Objectives for the AMTS example.

Viewpoint	**View**	**RM-BNS Object**	**Objective**	**ID**
Data	Artefact	Medical Database	Establish and justify the integrity of the Medical Database data artefact.	DV-1.2
Data	Processing	Data Discretisation	Establish and justify the necessity of the discretisation pre-processing technique applied to the Patient Record data.	DV-2.2
Model	Structure	Diagnostic Model	Establish and justify the basis for using a static Discrete Bayesian Network for the Diagnostic Model network.	MV-1.1 (Global)
Model	Structure	Anaphylaxis Variable	Establish and justify the basis for using a categorical distribution for the 'Anaphylaxis' Random Variable.[11]	MV-1.1 (Local)

[11] This objective represents a further decomposition of the *Variable* 'Model' object shown in Figure 6.3 into a specific Random Variable object.

Objective DV-1.2 addresses a second important data-centric consideration:

> *Establish and justify the integrity of the Medical Database data artefact.*

As discussed in Chapters 2, 3 and 5, the 'raw material' of a BNS is the data used in its construction. Data directly (or indirectly) influences the behaviour of BN models. This includes data that is not explicitly encoded within a deployed BNS. For example, it is unlikely that the AMTS would be deployed with direct access to the *Medical Database* used to train it. However, a *latent* representation of this data would persist in the parameterisation of the BN models within the AMTS. To establish confidence in this parameterisation, therefore, it is essential that the *integrity* of all data artefacts used in the design, construction, and training of a BNS are rigorously assessed.

This is the intent of this objective: to address the need to establish that the integrity of the *Medical Database* is assured – including in the event the database is not physically deployed as a part of the final system. The objective highlights the need to broaden the scope of what is considered to be a part of a software-intensive system in the context of BNS (and AIS generally). In the context of DO-178C, this is a subtle but important distinction from the provisions made in that guidance for 'Data Parameter Items' [69].

The remaining two objectives in Table 6.1 address model-centric system aspects. For example, consider objective MV-1.1 (Global):

> *Establish and justify the basis for using a static Discrete Bayesian Network for the Diagnostic Model network.*

This objective addresses the need to tackle the theoretical and practical motivation for using a static, discrete BN for the AMTS *Diagnosis Model*. It aims to ensure that a BN framework variant is selected with explicit consideration of the appropriateness of that BN variant for a specific problem. It is not uncommon for BN developers to select a BN variant without *explicitly* justifying *why* the variant was chosen. The selection of an inappropriate BN variant may prevent developers from capturing behaviours that may prove to be important in a given application, or from representing certain aspects of the domain being modelled [2]. This may constrain the completed BNS or produce undesired behaviours in the resulting system.

Conceptually, the objective also addresses a facet of the typical question of validity levelled at conventional software systems:

Has the right system been built?

For BNSs, this should be expanded to address a related but distinct question [64,159]:

Does the system need a BN model?

This objective (and objective MV-3.1) has been defined with this consideration in mind.[12] Specifically, it helps to address the question:

Does the system need the specific BN variant?

This aligns with a similar notion of validity discussed in the context of utilising Artificial Neural Networks (ANNs) in NASA's Intelligent Flight Control System (IFCS) programme [72]. Concretely, the objective addresses the problem of whether or not a given application *requires* a given BN variant to achieve the desired functionality. This includes evaluating whether or not alternative techniques or variants could be used to fulfil the same role.

The satisfaction of this objective will require BN developers to explicitly discuss the rationale behind the use of a specific BN variant – and therefore model – in their system. From a practical perspective, this objective is focussed on ensuring that no simpler approaches exist to solving the same problem. For example, a BN development team may use an elaborate, complex BN model for a given task when a simple Naive Bayes model (or something simpler still) would suffice for the task at hand. These simpler solutions may avoid the assurance challenges of more complex BNSs altogether.

The final objective in Table 6.1 addresses a similar, lower-level assurance consideration:

Establish and justify the basis for using a categorical distribution for the 'Anaphylaxis' Random Variable.

The objective is aimed at addressing the need for the inclusion of the variable-level structure of the 'Anaphylaxis' variable within the *Diagnosis Model*, and for choosing to use a given structure to represent an entity in the target domain. In this context, this objective could be seen to complement objective DV-2.2: it ensures that considerations related to potential sources of information loss are made explicit and considered from a model-centric perspective (while DV-2.2 addresses them from a data-centric perspective).

[12] MV-3.1 (Generic): Establish and justify the definition of the model.

It addresses the problem of selecting which distributional structure to represent a variable with: there may be modelling considerations that motivate the use of a discrete distribution to represent a continuous variable, such as reducing the complexity of a model, or improving its interpretability. A BN developer may choose a distribution to represent a variable that is known to be a crude approximation in the case of a variable, but generalises better to out-of-sample data than a more 'faithful' structure.

6.2.4 Model Criticality Analysis

While the objectives shown in Table 6.1 address the Diagnosis Model in particular, in practice similar objectives would be generated for each other model in the system, alongside each model's constituent 'component' models (i.e. variables). From an assurance perspective, each model may play a distinct role in the functional behaviours of the system. Consequently, the effort and rigour required to satisfy the objectives associated with models influencing safety-related behaviours will differ from those models that do not.

6.2.4.1 Failure Modes and Effects Analysis

The first step of the Model Criticality Analysis (MCA) process is to perform a failure analysis on any functional components of a BNS that utilise BN models. These components could be influenced by errors in the BN models – and ultimately in the data artefacts used to build these models. For this case study, a Failure Modes and Effects Analysis (FMEA) is used to illustrate how the MCA interfaces with existing safety analysis techniques. The FMEA was performed on three Functions defined in the system architecture model shown in Figure 6.2; these are the *Drug Administration*, *Health Monitoring* and *Practitioner Alert* functions.

The process of applying the FMEA proceeds as normal, with the selection of a function and the exploration of any associated failure modes, the effects of these failure modes and an enumeration of potential causes of these failure modes.[13] The results generated from the FMEA are shown in Table 6.2. For brevity, only a subset of those causes directly associated with errors in models have been listed in the analysis.

The FMEA indicates that the same model may be a contributing factor to failure modes associated with differing severity categories, and that distinct BN models may be

[13] An extensive review of failure analysis techniques can be found in the work of Pumfrey [83].

associated with failure modes with widely varying severity assignments within the same software system. For example, errors in the *Data Fusion* model may produce failure modes in the AMTS, but as this model is used only by software functions that provide advisory information (i.e. it does not execute any actions that directly affect the health of a patient), the severity of these failure modes is classified by the FMEA as negligible. In contrast, the case study assumes that the *Drug Treatment* function administers potentially dangerous drugs directly to patients without the possibility of human intervention. The FMEA highlights that the system may administer multiple different drugs to treat distinct symptoms. For example, consider the failure mode:

Failure to command administration of drug (Epinephrine) when required.

This failure mode is ranked as the most severe failure mode directly associated with a BN model in the AMTS. Practically, this would reflect a case where a patient is suffering from (potentially fatal) anaphylactic shock and would require an immediate dose of epinephrine as treatment. Failure to diagnose this state in a patient due to an error in the *Diagnosis Model* may directly lead to this outcome. Without further analysis, this could be attributed to an error in the state of the output variable associated with this diagnosis (i.e. the Anaphylaxis variable). However, the analysis sheds little light on precisely how this error may emerge. The following failure mode is also directly associated with a similar error in the *Diagnosis Model*:

Failure to command administration of drug (Morphine) when required.

In this case, the severity of the failure mode is lower. This highlights that the severity of failure modes associated with errors within individual models - and more importantly - individual variables, can vary significantly. As stated in previous chapters, the causes of these errors can not necessarily be identified through conventional analysis techniques: they may be the result of interactions between disparate parts of a BN model that may only be discovered through exploration of a model's structure and parameter-space [65,170,205].

Table 6.2: Failure Modes and Effects Analysis (FMEA) excerpts for an example applied to the RM-BNS architecture.

Function	**Failure Mode**	**Local Effect**	**System Effect**	**Severity**	**Cause**
Drug Administration: Treat Onset of Anaphylaxis (Software Component)	Failure to command administration of drug (Epinephrine) when required.	BNS does not inform drug management systems to administer drug (Epinephrine).	System does not administer patient with primary treatment for Anaphylactic Shock. Very high risk of patient death or serious physical damage.	Catastrophic	... c) Erroneous patient diagnosis (failure to recognise onset of Anaphylactic Shock) due to **Diagnosis Model** error (error in state of **Anaphylaxis Variable**). d) Erroneous patient diagnosis (failure to recognise onset of Anaphylactic Shock) due to algorithmic error (inference engine). ...
	Command to administer dose of drug (Epinephrine) when not required.	BNS informs drug management subsystems to administer epinephrine.	System administers dose of drug to patient. High risk of patient death (induced heart attack, organ failure), serious discomfort and physical damage.	Hazardous	... c) Erroneous patient diagnosis (diagnoses Anaphylactic Shock when not present) due to **Diagnosis Model** error (error in state of **Anaphylaxis Variable**). ...

Table 6.2: Failure Modes and Effects Analysis (FMEA) excerpts for an example applied to the RM-BNS architecture (continued).

Function	**Failure Mode**	**Local Effect**	**System Effect**	**Severity**	**Cause**
Drug Administration: Treat Discomfort (Software Component)	Failure to command administration of drug (Morphine) when required.	BNS fails to inform drug management systems to administer drug (Morphine).	System does not administer patient with treatment for moderate to severe discomfort. Risk of severe patient discomfort.	Minor	... b) Erroneous patient diagnosis (failure to recognise need for additional anaesthetic) due to **Diagnosis Model** error (error in state of **Insufficient Anaesthetic Variable**). ...
	Command to administer drug (Morphine) when not required.	BNS informs drug management systems to administer drug (Morphine) to patient.	System administers drug (Morphine) dose to patient. Risk of patient overdose, leading to serious physical damage or death.	Hazardous	... b) Erroneous patient diagnosis (erroneously diagnosing the need for additional anaesthetic) due to **Diagnosis Model** error (error in state of **Insufficient Anaesthetic Variable**). ...

Table 6.2: Failure Modes and Effects Analysis (FMEA) excerpts for an example applied to the RM-BNS architecture (continued).

Function	**Failure Mode**	**Local Effect**	**System Effect**	**Severity**	**Cause**
Health Monitoring (Software Component)	Failure to accurately indicate current Patient Health Status.	BNS does not reflect current status of patient.	System does not provide accurate information to medical practitioners on health of patient. Very low risk of erroneous treatment by medical practitioners.	Negligible	... d) Erroneous patient health indicators due to **Data Fusion Model error**. ...
Practitioner Alert (Software Component)	Failure to Alert Practitioner when required.	BNS does not signal system to raise alarm.	System fails to summon medical practitioner to patient. Risk of serious physical damage and/or discomfort.	Hazardous	... b) Erroneous alert recommendation due to **Diagnosis Model** error. c) Erroneous alert recommendation due to errors in **Alert Model (A)** and **Alert Model (B)**. ...

6.2.4.2 Assigning Model Authority Categories

The Model Authority Categories (MACs) and Model Criticality Indices (MCIs) provide a mechanism for describing the safety role of each model in a system at a lower level. The FMEA is used to provide the severity categories for the MCIs, while a RM-BNS system architecture model (similar to that provided in Figure 6.2) provides the necessary details for the assignment of MACs. For example, the MAC categories assigned to both *Alert Models* and the *Diagnosis Model* directly reflect the architecture. The initial MAC and MCI assignments are shown in Table 6.3. All three models are utilised directly by safety-related system functions, though the *Alert Models* would receive only a MAC-2 assignment, and the Diagnosis Model would receive a MAC-0 assignment. This arises as a consequence of the *Diagnosis Model* being used in a *monolithic* architecture. In contrast, the *Alert Models* are used in an *ensemble* architecture; this configuration ensures a degree of redundancy in how each model's outputs are used.[14] Ensemble configurations may reduce the potential for model-centric common-cause error modes and single point of failure error modes, the latter of which is the principal weakness of monolithic configurations.

The MCIs are then defined based upon the MAC assignment and the severity categories provided by the FMEA. The severity category used to define the MCIs is drawn from the most severe category of failure mode identified by the FMEA. Both *Alert Models* receive an MCI-1 assignment. This reflects the additional risk associated with the use of these models given the most severe category assigned to a failure mode associated directly with these models is defined as Hazardous. Similarly, the *Data Fusion Model* reflects the criticality of a model that despite indirectly influencing the state of a patient (captured in its MAC-3 assignment), the low severity of failure modes associated with this model produces the lowest category of MCI: MCI-4.

[14]The definitions of *ensemble* and *monolithic* models can be found in section 4.3.2.

Table 6.3: Model Authority Categories (MACs) and Model Criticality Indices (MCIs) for the AMTS system.

Model	**Description**	**Authority**	**Criticality**
Data Fusion	Model for synthesising data from all available input sources to provide additional insight into patient health and status (e.g. predicting vital signs, drug levels).	MAC-3 Supporting	MCI-4
Diagnosis Model	Model for diagnosing patients with the following health statuses: Left Ventricular Failure; Pulmonary Embolism; Anaphylactic Shock; Insufficient Anaesthesia; Tracheal Intubation Status; Damaged Tubing (Equipment error); Hypovolemia; Disconnected Devices (Equipment error).	MAC-0 Full	MCI-0
Alert Model (A)	Model (A) for alerting medical practitioners in the event of patient discomfort or requirement of treatment outside of AMTS's scope.	MAC-2 Partial (II)	MCI-1
Alert Model (B)	Model (B) for alerting medical practitioners in the event of patient discomfort or requirement of treatment outside of AMTS's scope.	MAC-2 Partial (II)	MCI-1

6.2.4.3 Assigning Variable Criticality Indices

The MCIs provide a mechanism for ensuring the *proportionality* of efforts to satisfy those system-specific objectives that directly address the *global* aspects of a model: the overall dynamics and properties of a model are the focus of the MCIs.[15] They reflect the level of required integrity and rigour of all processes associated with exploring these properties. However, the MCI assignments are not intended to address the same notions of integrity and rigour associated with *local* aspects of a model; this is the aim of the Variable Authority Categories (VACs) and Variable Criticality Indices (VCIs).

Table 6.4: Severity assignments for variables in the *Medical Diagnostic* model.

Variable	**Type**	**Identifier**	**Severity**
Anaphylaxis	Safety Related	ANAPHYLAXIS	0 - Catastrophic
Damaged Tubing	Safety Related	KINKEDTUBE	3 - Minor
Tracheal Intubation	Safety Related	INTUBATION	2- Major
Pulmonary Embolism	Safety Related	PULMEMBOLUS	0 - Catastrophic
Insufficient Anaesthesia	Safety Related	INSUFFANESTH	3 - Minor
Disconnected Device	Non-Safety Related	DISCONNECT	4 - Negligible
Hypovolemic Shock	Safety Related	HYPOVOLEMIA	1 - Hazardous
Left Ventricular Failure	Safety Related	LVFAILURE	0 - Catastrophic

Assigning VACs begins by identifying a set of Safety-Related Variables (SRVs). These are variables drawn from a given BN model that are utilised directly by a software function. The SRVs will therefore be a subset of variables that will be a subset of a model's output variables. Importantly, a model may contain outputs that are never utilised in support of safety-related software functions. The intent of the VACs and VCIs is therefore to provide a higher degree of resolution on which parts of a BN model are influential with respect to

[15]Though not attempted in this thesis, the MCIs were defined with the aim of providing a *general* framework for categorising AI models, including models other than BNs. Without change, the MCI and MAC classifications could theoretically be applied to the description of other AISs.

the SRVs.

From an assurance perspective, some variables within a model will be more important than others. As outlined in Chapter 4, the VCIs aim to help ensure that effort is directed towards those aspects of a system that most require them, and therefore avoid a 'blanket' approach to analysing a BN model indiscriminately. This is partially motivated by an aim to ensure the efficiency of any assurance activities used to evaluate a BN model and its components. However, it is also motivated by potential computational limitations: indiscriminate, exhaustive application of some evaluation techniques may not simply be inefficient, but also computationally intractable. Therefore, the VCIs aim to provide information on which aspects of a model are most important to the model's behaviours. This then facilitates the *targeted* application of these evaluation techniques (and moderates the scope and rigour of these techniques) to the aspects of a model that are of most interest to assurance practitioners.

The identification of SRVs can be achieved using the outputs of the FMEA, or through extensions of this analysis that specifically consider the role of model outputs in the functional behaviours of the system. The FMEA produced for this case study includes limited information on the role of variables, but this is sufficient to support the assignment of VACs. Table 6.4 shows all output variables for the *Diagnosis Model*, alongside their model identifiers (corresponding to the abbreviations used in the implementation of the model), the category of the variable (i.e. either an SRV or Non-SRV), and the severity of failure modes associated with each variable as identified by the FMEA.[16]

With the SRVs identified, Variable Criticality Analysis (VCA) may be performed. The VCA technique was applied to the *Diagnostic Model*, and the resulting VCI matrix is shown in Figure 6.4. The visualisation highlights the pairwise interactions within the model and the relative degree of influence of these interactions upon one another. These interactions are the basis of the VAC and VCI assignments. However, the visualisation can also be used to highlight clusters of variables that interact with one another. While the utility of this observation was not explored in detail in Chapter 4 and is beyond the scope of this case study, it is worth noting that information such as this may be useful to assurance practitioners in ensuring the targeted assurance and fine-tuning of a given BN model. Furthermore, the algorithm introduced in Chapter 4 does not provide an

[16]Non-SRV indicates a variable that receives a severity rating of zero.

exhaustive exploration of the state-space of a BN model, and therefore the absence of any strong interactions (represented as light-coloured areas in the visualisation) between two variables does not necessarily imply that no strong interactions *could* exist in some model topologies.

Table 6.5 shows a subset of the final VCIs for the *Diagnosis Model*. The 'Preliminary Criticality' assignments reflect the VCIs the variables would have received if interactions between variables were *not* accounted for.[17] This provides a measure of what has been gained by performing the analysis: by using the technique, an assurance practitioner can establish a clearer understanding of which aspects of the model influence safety-related system functionality, and the degree of contribution of *individual* variables to the state of the SRVs (i.e. the output of the model).

For example, the results of the VCA for the Patient History variable highlight this point. The interaction between the *'Patient History'* variable and the Left Ventricular Failure variable is assigned a VAC-3 classification and VCI-2 index respectively. This indicates that the former variable has a *Low* VAC over the latter variable. However, when this category is considered in the context of the severity of a failure mode associated with an error in the Left Ventricular Failure variable, the criticality of the Patient History variable is then higher. This information can be used by assurance practitioners to proportionally address assurance objectives associated with this variable.

The MCA methodology has been designed to provide insight when used as a standalone technique. It can provide assurance practitioners with safety-related insights into the dynamics of a BN model. For example, Figure 6.4 clearly highlights a number of strongly interacting groups of variables within the model. Many of these are ranked as low criticality model aspects, though some (such as 'LVEDVOLUME' and 'STROKEVOLUME') are high criticality and appear to be highly influential upon one another. By highlighting this assurance insight, assurance practitioners can begin to target development activities related to these variables to better understand the implications of these interactions.

[17] This reflects the state of information available to assurance practitioners prior to using the VCA technique.

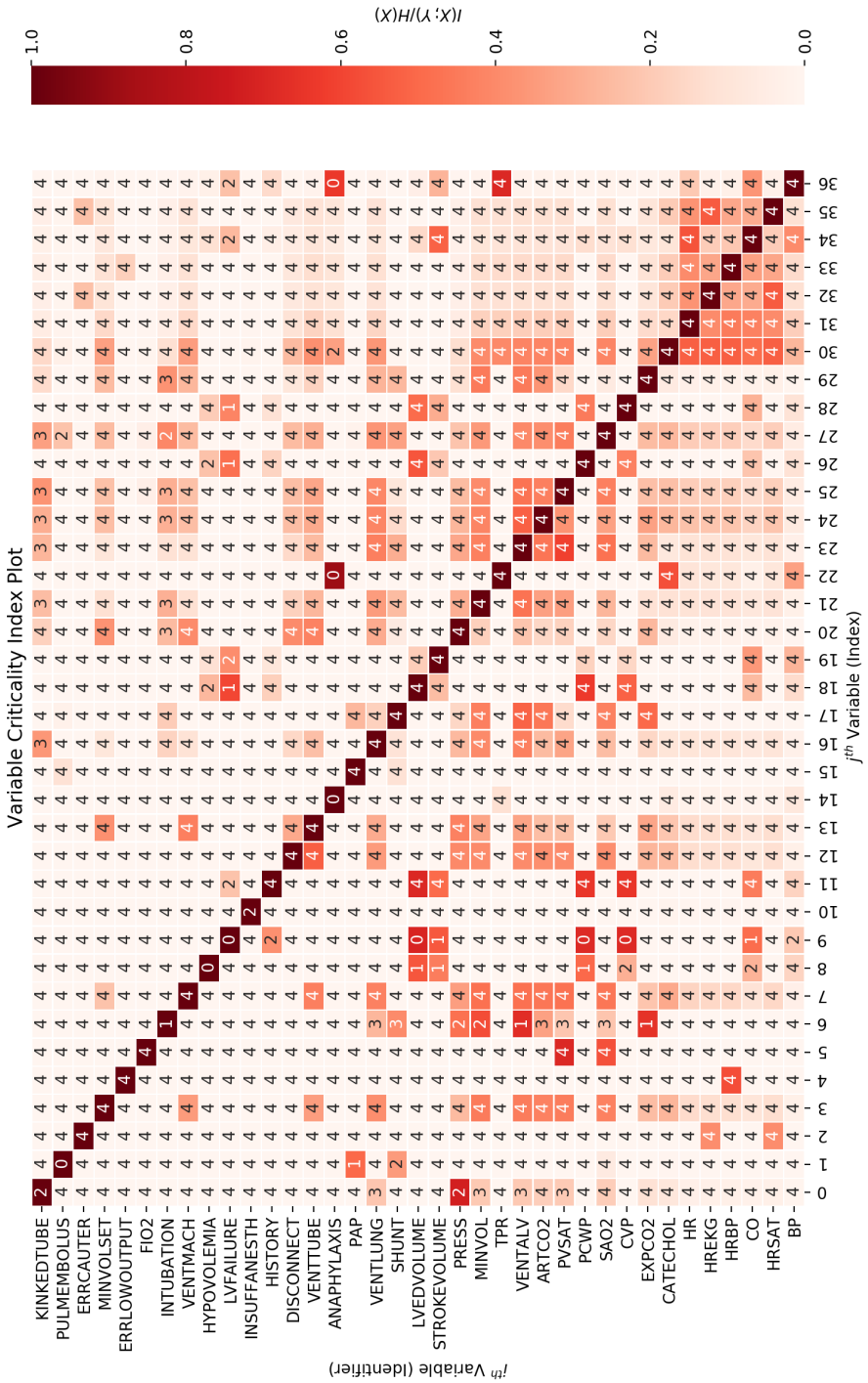


Figure 6.4: Visualisation of Variable Criticality Index (VCI) assignments for the *Medical Diagnostic* model.

Table 6.5: Criticality assignments for a selection of variables for the *Diagnostic Model*.

Variable	**Identifier**	**Preliminary Crit.**	**Criticality**
Anaphylaxis	ANAPHYLAXIS	VCI-0	VCI-0
Pulmonary Capillary Wedge Pressure	PCWP	VCI-4	VCI-0
Patient History	HISTORY	VCI-4	VCI-2
Lung Ventilation	VENTLUNG	VCI-4	VCI-3
Disconnect	DISCONNECT	VCI-4	VCI-4

The analysis for this section was performed using an inference engine developed and implemented over the course of this project. The MCA algorithm introduced in Chapter 4 was built upon this engine, and the results shown in Figure 6.4 and Table 6.5 represent outputs from this tool.

While MCA may be used as a standalone approach, it was design with the intent of supporting the assurance of BNS Model Viewpoint Aspects more broadly. This involves utilising the results of the MCA (and by extension, VCA) to enable the sufficiency of assurance efforts to be established. This is the subject of the remaining sections of this case study.

6.2.5 Mapping Objectives to Evidence Classifications

It is essential that the relationship and rationale linking objectives and their associated items of evidence is now made explicit. This can be achieved using the framework and concepts introduced in Chapter 5. Table 6.6 gives (an incomplete) set of evidence items and evidence classifications for the RM-BNS Model Viewpoint objectives introduced earlier in this case study generated using this framework. Consider the objective MV-1.1-A. The table enumerates five distinct items of evidence that may be used to justify the satisfaction of this objective. This is not an exhaustive list. However, for this single objective, the evidence classes define a hierarchy within the evidence items that aligns with the sentiments of standards such as Defence Standard (DS) 00-056. As discussed in

Chapter 5, this standard states:

> *"Explicit, objective evidence [items] are more compelling than those that appeal to judgement, custom or practice."*

For example, with respect to Table 6.6, Evidence Items (a) and (b) represent items arguably most in keeping with the sentiment of DS 00-056. These are 'objective' analyses that provide concrete statistical metrics for characterising the properties of the *Diagnosis Model*. In contrast, the remaining three objectives appeal to various forms of expert judgement. However, within these remaining three evidence items, the *role* of those items of evidence is made explicit and a hierarchy of these items is exposed. From an assurance perspective therefore, the evidence classifications (and characteristics) provide the foundation for an assessment of how compelling a given set of evidence items are with respect to a given objective.

This is also important as the scope and importance of evidence items will vary across system aspects and across specific objectives. For example, Table 6.6 also provides a subset of evidence items that may be used to satisfy the second objective (MV-1.1-B). As discussed in section previously, this objective represents an instantiation of objective MV-1.1 for a specific local model structure aspect of the *Diagnosis Model*.

Consequently, the technical analyses applied to this variable may well be narrower in scope and focus more on local considerations. By extension, additional distinct evidence items may be used in the satisfaction of this objective (MV-1.1-B) in comparison to that of the previous objective (MV-1.1-A). The evidence classifications once again support the development of an evidence hierarchy which can be used as the first step towards establishing the sufficiency of efforts to satisfy the given objective.

Table 6.6: A mapping of AMTS System-Specific Objectives to Evidence Items and Evidence Classifications.

View	**ID**	**Objective**	**Evidence Item**	**Class (ERC)**	**Class (ETC)**
Structure	MV-1.1-A	Establish and justify the basis for using a static Discrete Bayesian Network for the **'Diagnosis Model'** network.	(a) Statistical Analyses showing that the domain (ICU ward/patients) can be modelled effectively as a time-independent (static) problem.	Direct	Statistical
			(b) Statistical Analyses showing that the use of a discrete distribution is appropriate for the problem.	Direct	Statistical
			(c) Expert opinion that the domain can be modelled as a time-independent (static) problem.	Direct	Qualitative
			(d) A review of the available 'Knowledge Base' indicating that there are no underlying scenarios that may invalidate the 'static' and 'discrete' properties of the **'Diagnostic Model'**.	Backing	Qualitative

Table 6.6: A mapping of AMTS System-Specific Objectives to Evidence Items and Evidence Classifications (continued).

View	**ID**	**Objective**	**Evidence Item**	**Class (ERC)**	**Class (ETC)**
			(e) Expert justification that the domain *should* be modelled using a time-independent (static) BN instead of alternative modelling approaches.	Reinforcement	Qualitative
...	...	...	...	...	
Structure	MV-1.1-B	Establish and justify the basis for using a categorical distribution for the **'Anaphylaxis'** Random Variable.	(a) Statistical Analyses showing that the **'Anaphylaxis'** RV can be modelled as a discrete categorical RV.	Direct	Statistical
			(b) Expert opinion that modelling the **'Anaphylaxis'** as a discrete categorical RV is appropriate for the domain aspect (the state of the patient) being modelled.	Direct	Qualitative

Table 6.6: A mapping of AMTS System-Specific Objectives to Evidence Items and Evidence Classifications (continued).

View	ID	Objective	Evidence Item	Class (ERC)	Class (ETC)
			(c) Statistical Analyses showing that using a discrete categorical RV would not negatively affect the performance of the **'Diagnostic Model'**.	Backing	Statistical

As with the MCA technique, while the evidence classification framework introduced in Chapter 5 can be used alone, it was defined with the intent of being used alongside the rest of the contributions of this thesis. Specifically, whether or not a set of items of evidence are *sufficient* for the satisfaction of a given objective is dependent on the *criticality* of the system aspect addressed by that objective. As discussed in Chapter 5, this requires the integration of criticality metrics with the evidence classification framework.

6.2.6 Mapping Objectives to Criticality Metrics

With a framework for describing and communicating the role and nature of evidence in place, and the evidence items for the AMTS system aspects classified, the next step is to integrate criticality metrics to provide a clearer view of the safety implications of Model Viewpoint system aspects and consider how compelling the available evidence for a given objective is in the context of this information.

For the purposes of this case study, the outputs of the MCA are used to provide criticality metrics for each *Model* object within the RM-BNS system architecture model (such as that shown in Figure 6.3). Each of these objects will have a set of object-specific RM-BNS objectives associated with it. This enables the criticality of a system aspect to be mapped directly onto the objectives associated with that aspect. An example of such a mapping is shown in Table 6.7.

While the MCA technique is used for this case study, alternative techniques could of course be used to establish the criticality of other system aspects. Indeed, while there are a number of potential techniques for establishing the criticality of system aspects outside of the Model Viewpoint aspects addressed by the MCA technique, the selection and application of such techniques to the RM-BNS objectives has been beyond the scope of the project.

The mapping shown in Table 6.7 highlights a number of important considerations for assurance practitioners. For example, objective MV-1.1-B addresses aforementioned assurance concerns associated with the *'Anaphylaxis'* RV. The MCA carried out in section 6.2.4 assigned this variable a VCI-0 classification. From an assurance perspective, therefore, the assurance of the properties and behaviours of this variable is particularly important for the AMTS. In contrast, the objective MV-1.1-C addresses the *'Disconnected Device'* RV. This RV received a VCI-4 assignment in the MCA. Consequently, what may be regarded as *sufficient* for the satisfaction of each of these objectives may vary: the standard of

evidence required for the satisfaction of MV-1.1-C may be lower than that required for the satisfaction of MV-1.1-B.

6.2.7 Proportionality and Sufficiency

At this point in the case study, the following insights have been established:

- An understanding of the interactions between aspects of the AMTS (using the RM-BNS architecture model).
- A set of system-specific assurance objectives.
- An in-depth understanding of the assurance implications of the dynamics of the *'Diagnosis Model'*.
- An understanding of the criticality of individual (model-centric) system-specific objectives and the items of evidence associated with them.

Together, these insights can be used to support the discussion of the sufficiency of assurance efforts undertaken in the development of the AMTS. At a granular level, the case study has identified how important a limited set of system aspects are to the safety of the AMTS and described and classified the evidence available to satisfy the objectives associated with these aspects. There are two principal approaches that could be adopted by assurance practitioners in establishing the proportionality and sufficiency of efforts to satisfy any given objective/s.

Table 6.7: An Objective-Criticality Mapping for aspects of the AMTS.

View	**ID**	**Criticality**	**Objective**	**Evidence Item**	**Evidence Classes**
Structure	MV-1.1-A	MCI-0	Establish and justify the basis for using a static Discrete Bayesian Network for the **'Diagnosis Model'** network.	...	...
	MV-1.1-B	VCI-0	Establish and justify the basis for using a categorical distribution for the **'Anaphylaxis'** Random Variable.	...	...
...	...	...	...	...	...
	MV-1.1-C	VCI-4	Establish and justify the basis for using a categorical distribution for the **'Disconnected Device'** Random Variable.	...	...
...	...	...	...	...	...
	MV-1.1-D	VCI-2	Establish and justify the basis for using a categorical distribution for the **'Tracheal Intubation'** Random Variable.	...	...

6.2.7.1 Varying Objectives

The first approach outlined in Chapter 5 that may be adopted by assurance practitioners is an approach similar to that taken in DO-178C: to vary the number of objectives associated with different system aspects in accordance with the criticality of a given BNS. One way of implementing this approach could be to vary the number of RM-BNS objectives that must be satisfied given the criticality of a system aspect addressed by that objective. For example, in the context of the *'Anaphylaxis'* variable in Table 6.7, an assurance practitioner should seek to satisfy every RM-BNS objective associated with that system aspect. In contrast, it may be reasonable to satisfy only a relatively small subset of objectives for the *'Disconnected Device'* system aspect.

As outlined in Chapter 5, a systematic way of implementing such an approach may be to utilise the inherent structure of the RM-BNS objectives: the objectives associated with each object in the RM-BNS framework are ordered such that lower indexed objectives are the most granular objectives (typically dealing with specific technical system aspects), while the higher indexed objectives are associated with more generic notions of assurance for a given system aspect. The objectives have been defined to draw attention to system aspects that may be overlooked by BN developers and assurance practitioners, and to encourage stakeholders to explicitly discuss and justify the rationale behind key modelling decisions.

Table 6.8: Varying the number of objectives for a BNS system aspect in proportion to the criticality of the aspect.

Object	**Identifier**	**Criticality**	**No. Objectives**
Model	ANAPHYLAXIS	VCI-0	17
Model	HISTORY	VCI-2	9
Model	DISCONNECT	VCI-4	1

Table 6.8 shows how the number of variables may be varied given the criticality of the RVs introduced in section 6.2.4 in accordance with the suggestion outlined in Chapter 5. For example, the 'Anaphylaxis' RV would require the satisfaction of all 17 objectives

targeting that system aspect as it was assigned the highest criticality category. In contrast the 'Patient History' RV would require the satisfaction of only 9 objectives as it received a moderate criticality assignment. Practically, this corresponds to all but the first two objectives in each of the four Model Viewpoint Views.

Those variables with the lowest criticality assignments – such as *'Device Disconnect'* – may only need to demonstrate the satisfaction of a single objective. These objectives correspond to the highest-level objectives of each View. This would waive the need for developers to explicitly justify the design decision associated with the other objectives, and instead fall back to a higher-level (lower-confidence) stance on the assurance of the system aspect. In all cases, the satisfaction of an objective would require the presentation of equally compelling evidence.

Finally, as discussed in Chapter 5, there is scope for introducing additional considerations related to the *independence* of these objectives. This would involve some number of objectives being satisfied by independent development teams to – theoretically – ensure the objectivity of the evidence used to satisfy an objective.

6.2.7.2 Varying Evidence

Alternatively, assurance practitioners may choose to adopt an approach that instead varies the quantity and quality of assurance evidence for a given system aspect. In the case of the AMTS, this would involve satisfying *all* objectives for *all* system aspects in the AMTS. However, adherents of this approach would accept lower-confidence evidence for system aspects associated with lower levels of criticality and would require more compelling evidence as the criticality of a system aspect increased.

For example, consider again the objectives in Table 6.7. The objectives MV-1.1-B and MV-1.1-C represent objectives addressing two RV objects at the opposite ends of the criticality scale. In both cases, this approach would require the satisfaction of all 17 objectives for both objects. However, it may be the case that the evidence that is available to demonstrate the satisfaction of objective MV-1.1-C is limited to an item of evidence that would be classified as a *Backing*, *Qualitative* item of evidence. Given the low criticality of the *'Disconnect'* RV, an assurance practitioner may decide that this is sufficient to hold MV-1.1-C satisfied. In contrast, such evidence would not be sufficient to hold MV-1.1-B as satisfied.

A further example of this may be in the application of the techniques outlined in

section 5.5.5.1 on the application of statistical techniques to satisfying objective MV-2.2. This system-specific objective for the *'History'* RV may be defined as follows:

> *"MV-2.2 – Establish and justify the accuracy of the parameterisation of the History RV."*

As the AMTS parameterisation is learned from data, the statistical confidence in the parameterisation and Chan's parameter boundary checking technique can be applied directly to this problem. As Figure 6.4 indicates, the *'History'* RV influences the higher criticality SRV *'Insufficient Anaesthesia'* RV. An assurance practitioner may therefore seek to satisfy the above objective by first characterising the statistical confidence in the parameter estimate. The confidence intervals for a parameter in the *'History'* RV (assuming 50 observations) are shown in Table 6.9. The lower number of observations acts to broaden the confidence intervals, thereby making the parameter estimate more uncertain. The *'History'* RV was assigned a VCI-2 category earlier in this chapter. By adopting the proposed confidence interval for this criticality as outlined in Chapter 5 (95% interval), an assurance practitioner can have high statistical confidence that the parameter lies within the range 52.1% and 77.7%. This establishes the accuracy of the parameter estimate.

Table 6.9: Confidence intervals on the *'History'* RV parameter estimates.

Crit.	**Interval**	**Parameter**	**Lower Limit**	**Upper Limit**
4	75.0%	66.0%	57.9%	73.2%
3	90.0%	66.0%	54.4%	75.9%
2	95.0%	66.0%	52.1%	77.7%
1	99.0%	66.0%	47.8%	80.4%
0	99.9%	66.0%	42.9%	83.3%

As before, the next step is to *justify* this accuracy – why is it an acceptable accuracy for the given application? This could once again be achieved using Chan's technique. As before, assuming that assurance practitioners wish to demonstrate that the *'Insufficient Anaesthesia'* RV would not be significantly influenced by potential errors in the *'History'* RV, they can compute the upper and lower bounds on the possible values the *'History'* parameters may take. Using the figures in Chan's work, this would once again produce

upper and lower bounds of 81% and 50% respectively.[18]

Given this information, an assurance practitioner can then demonstrate that the accuracy of the *'History'* RV is sufficient as the confidence intervals lie within the bounds defined by Chan's technique. Practically, an assurance practitioner can expect the outputs of the *'Insufficient Anaesthesia'* RV to be robust to errors in the *'History'* RV. However, if subsequent analyses were to increase the criticality of the *'History'* RV for any reason, this evidence would *not* be sufficient: the range of plausible values would lie *outside* of the upper and lower bounds indicated by Chan's technique. In this case, it would force assurance practitioners to gather more data or run additional analyses to provide evidence for the satisfaction of the objective – they would need greater confidence in their evidence.

As discussed in Chapter 5, the research carried out over the course of this project suggests that an approach to the assurance of a BNS that focusses on the varying assurance evidence should be favoured by assurance practitioners. In general, these approaches should be able to accommodate changes within the BN field more readily, and can support the development of more esoteric BNS applications without changes to the underlying assurance framework – as may be the case with more prescriptive approaches.

In the context of this case study, varying the evidence instead of the objectives enables assurance practitioners to ensure that all system aspects are *explicitly* considered and targeted by their analyses. Moreover, it more readily accommodates the potential for shifts in the criticality of system aspects as a system develops. For example, if the *'Diagnosis Model'* were to be updated to include additional RVs, the dynamics of the model may be dramatically altered. This may produce different criticality assignments for variables within the model. For assurance practitioners adhering to such an approach, this may require only the enhancement of existing work, while those adhering to an objective-driven approach may have to undertake entirely new assurance activities.

6.2.8 Note on Generalisability

The case study presented in this chapter has been designed to be representative of a typical BNS used for a safety critical application. It was synthesised from several publications and therefore reflects aspects of a number of BNSs that have been – or currently are –

[18]This is adopted purely for convenience and brevity. The dynamics of the ALARM network would modify Chan's figures and this estimate would therefore be inaccurate. However, it serves the purpose of this example.

in active service. However, it is still a singular case study and by extension cannot cover all potential applications or operational scenarios of mission-critical BNSs. In particular, while the case study covers a range of important BNS assurance concerns, it does not provide evidence that the proposed approaches will cover all potential concerns in all scenarios.

For example, there are no known limitations or omissions within the generic RM-BNS objectives. However, it is possible that future work may identify concerns beyond the scope of those considered over the course of this research. These concerns may arise as a consequence of developing novel BNS applications, utilising BNS variants not considered in the course of this work or from other technological advancements within the BN/PGM modelling frameworks themselves. Importantly, it is unlikely that the emergence of new assurance concerns would require the modification of the framework's viewpoints or views, though they may rather require the refinement or extension of the proposed objectives instead. This would not invalidate the framework, and indeed the framework was designed with thought to ensuring extensibility in the event of such a case as this.

Beyond the generic objectives themselves (justification of which has been provided in Section 3.6), there remains the challenge of deriving a set of system-specific objectives. When deriving objectives for this case study, no practical challenges were encountered, and every important aspect of the system was covered by at least one derived objective. However, as discussed in Chapter 3, the derivation of these objectives requires judgment on the part of an assurance practitioner, and it is possible that a future case study may highlight a case in which an objective is not applicable to a particular application, or that may require modified or additional objectives to effectively cover a particular concern. In the former case, practitioners should feel free to drop objectives provided a clear, compelling rationale for doing so can be provided. In the latter case, the generic objectives could be extended to accommodate these findings.

With respect to the application of the the MCA technique introduced in Chapter 4 to this case study, no modification was required. In cases utilising 'standard' BN variants, the technique should generalise well to other models, though extremely large models may encounter issues related to scalability. This will be discussed further in Chapter 7. More generally, some modifications to the core Sensitivity Analysis (SA) technique may be necessary to accommodate some BN model variants, or other non-standard architectures. In these cases, the general concept and insights will remain applicable and as outlined

in Chapter 4, the MCA process was designed with the aim of enabling a practitioner to replace the core SA algorithm with one of their choosing, provided a set of simple constraints are satisfied.

In conclusion, the threats to the external validity of the techniques applied in this case study and their ability to generalise to real systems revolve primarily around the possibility that future work identifies omissions or shortcomings of the framework in the light of an application that is beyond the scope of the applications considered in this thesis. However, this would not be fatal to the framework and the contributions of this thesis as a whole, though it may require the extension or modification to more effectively account for such an outcome. Ultimately, this work has demonstrated the feasibility of the provision of evidence for a subset of objectives. Future work will need to evaluate whether the methodology presented here is equally feasible for a wider range of applications and evidence.

6.2.9 Summary

This case study has provided an overview of the application of the techniques proposed in this thesis to an example system. It has indicated the potential utility of the RM-BNS framework in modelling BNSs and for generating comprehensive objectives that aim to explicitly address unconventional system aspects and considerations. It has also provided an example application of the MCA technique to a BN model, and illustrated a number of the insights that may be gained through the utilisation of this approach. Finally, it has outlined two potential strategies for ultimately assuring a BNS in the context of the example system.

6.3 Practicability of Application

The utility of a new assurance technique or concept lies in the ability of assurance practitioners to apply them to an engineering problem and observe the relative merits and challenges associated with them. The application of the techniques and concepts proposed in this thesis to a in-development BNS was not possible over the course of this research project. However, it is possible to evaluate the practicability of these techniques in the context of this chapter's case study, and to explore some broader considerations associated with the application of the contributions of this thesis to the development of a

mission- and/or safety-critical BNS.

6.3.1 Application of the RM-BNS

The RM-BNS introduced in Chapter 3 was developed to support the description and modelling of BNSs similar to that of the motivating example outlined in Chapter 1: a fault diagnosis and prognosis system that directly influences the functional behaviour of a system. The priority of the work presented throughout this thesis has been to support the assurance of this class of system in the first instance. However, the RM-BNS framework has been defined with the aim of accommodating a range of commonly used BN model architectures and variants; it can be considered to be a general framework for modelling BNSs. Consequently, it is envisioned that with relatively minor modifications, the RM-BNS framework could be applied to any BNS.

The case study provides an indication that the RM-BNS framework can be applied in a straightforward manner to the same class of system as the system in the motivating example of the thesis. The case study also indicates that the techniques introduced in this thesis can expose system aspects that may be overlooked in conventional software safety analysis and certification approaches. In particular, it highlights the centrality of Model Viewpoint and Data Viewpoint system aspects in the architecture of a BNS, and indicates mechanisms through which a BNS may be exposed to its operational environment and the potential impact of changes to this environment upon the functional behaviour of the system. A primary function of the RM-BNS framework is to visually capture *interactions* between system aspects. These interactions are crucial to understanding the behaviour of a BNS and by extension to assure such a system.

One of the key concepts – and contributions – of the RM-BNS is the need to address the assurance of BNSs using a multi-viewpoint approach. The viewpoints proposed in Chapter 3 were defined to address specific BNS assurance challenges. The application of these viewpoints both throughout this thesis, in section 6.2 and in other work suggests that this may be a convenient and intuitive way to communicate key BNS assurance challenges to relevant stakeholders [156]. Furthermore, the viewpoints (and to some extent the RM-BNS framework more broadly) are intended to be relatively generic, and may be used to describe assurance challenges associated with other AI techniques. They may therefore provide a starting point for a generic framework for the assurance of AISs. However,

further exploration of this work has been beyond the scope of this thesis.[19]

Finally, the RM-BNS provides a framework for sharing information between assurance practitioners and BN developers about the implications of design and implementation decisions upon a BNS using a shared language. The aim of the framework is to facilitate an informed debate about the assurance implications of these decisions in a transparent, interpretable format. The application of the framework to the AMTS (and other systems within literature) indicates that the RM-BNS can capture and relay information about the architecture of the system and that this can be used for subsequent analysis and testing of the system. The experience of applying it to the AMTS also indicates that it is highly flexible in terms of capturing novel system architectures.

6.3.2 Application of the RM-BNS Objectives

Much like the RM-BNS framework itself, the verification and validation objectives introduced in Chapter 3 were defined to address assurance concerns associated with the system in the motivating example of this thesis. The proposed objectives have been refined to help ensure their generality to alternative BNS use-cases and variants. The objectives for the Model Viewpoint have been defined with the hope of supporting the description of other AI approaches too. This was aimed at providing a mechanism for defining a set of generic AIS verification and validation objectives to support the assurance of AISs in general. However, the development of these ideas was also beyond the scope of this research project, though these ideas have been discussed in other work [207].

With respect to the case study, the RM-BNS objectives were applied and refined according to the processes outlined in Chapter 3. The resulting system-specific objectives facilitated the exposure of several important assurance considerations in the context of the AMTS. The example objectives provided in section 6.2.3 highlight the need to consider the role of various data pre-processing steps on the expressiveness of the BN models used in a BNS, and the distinct assurance considerations that are introduced when refining a Model Viewpoint objective for *local* and *global* model assurance considerations.

From a practical perspective, the generation of objectives for the AMTS was a slow process. Objectives were generated for each object in the model fragment shown in Figure 6.3. This corresponded to more than 200 objectives, though the granularity and scope of

[19] Some of the concepts related to this idea were presented in [206, 207].

these objectives is much narrower than in existing standards.[20] As the case study focussed on only a subset of these objectives, a full exploration of the operational overhead associated with satisfying all of these objectives was not attempted. However, the objectives can be compartmentalised and tackled on an atomic basis, meaning that – in the limited experience provided by the case study – the generation and management of system-specific objectives for a larger system should not be an overwhelmingly complex problem.

More broadly, the application of the RM-BNS objectives to a BNS provides a structured mechanism for the development of a BNS that is unique within the BN domain [206]. The challenge of assuring a BNS lies in 'unmasking' the 'dark art' aspects of BNS development for assurance practitioners and ensuring that all relevant stakeholders are fully aware of the implications of design and implementation decisions for a given system. They provide a first step towards systematising the assurance of BNSs.

6.3.3 Application of the Model Criticality Analysis Technique

The Model Criticality Analysis (MCA) approach introduced in Chapter 4 was developed to provide a general technique for exploring the dynamics of any BN model and to integrate safety information into existing BN analysis techniques in order to provide a safety-focussed assessment of a BN model. This assessment can then be used to derive *Model* and *Variable* criticality metrics that can be used to provide assurance practitioners with insight into the safety implications of BN model design decisions.

From the outset, MCA was developed with the aim of supporting the analysis of any BN variant or BNS architecture: it does not assume the number or configuration of the models used in a BNS, nor does it assume the internal structure of these models. Within the case study, it was demonstrated that the MCA approach can be applied to models that represent 'basic' architectures. This demonstrates that the MCA approach is applicable to the most common forms of BN models. The implementation of the MCA algorithm required only relatively small modifications to standard exact inference and sensitivity analysis algorithms.

For this project, the MCA code was written as a standalone tool built upon a proprietary BN library that was developed specifically for this research project. The MCA tool can parse many common BN model file formats and utilises a variant of the HUGIN algo-

[20] This does not include the generation of objectives for each variable in the *Diagnosis Model*.

rithm for inference and sensitivity analysis. The implementation of the inference algorithm was verified against both the SAMIAM BN tool developed by University of California, Los Angeles (UCLA), and a trial version of the HUGIN software package (HUGIN LITE). This tool will be made available alongside this thesis, and a more detailed description of the implementation can be found in Appendix C. A more practical long-term solution would be to re-implement the MCA algorithm on top of an existing BN tool. An attempt was made to implement a version of MCA using the libDAI package developed by Stanford University, though this was ultimately abandoned due to time constraints. Integration with other tools was not attempted due to intellectual property and cost considerations. However, implementation of the MCA technique atop more feature-rich tools could enable the extension of the method to other BN architectures beyond those discussed in this thesis.

With respect to the case study, the analysis performed in section 6.2.4 indicated how the MCA technique could be used to help identify that a number of variables within the *Diagnosis Model* were highly influential with respect to the state of several Safety-Related Variables (SRVs). The output of the MCA (shown in Figure 6.4) provided additional information on safety-related aspects of the model, and provided information beyond that available through a simple inspection. Furthermore, MCA can be used to provide assurance practitioners with a methodology for prioritising assurance efforts when targeting both global and local aspects of a BN model. Practically, this enables assurance practitioners to work more efficiently when evaluating a BN model – they need not treat the entire model as a 'black box', and can instead focus efforts aspects of a model most relevant to assurance efforts. A further side-effect of the MCA technique is that it provides assurance practitioners with a visualisation of the *safety-related* dynamics of a BN model. No other tool currently exists to provide this information.

However, as indicated in Chapter 4, the MCA approach as defined in this thesis should be considered to provide an *overview* of the dynamics of a model rather than a detailed analysis of interactions between variables. The MCA as it is proposed in Chapter 4 cannot explore combinations of interactions within the model and provides comparatively coarse-grained insight into the underlying dynamics of the model. However, the case study indicates that the analysis can be used to highlight that important assurance information can be obtained and used to support the assurance of the BNS.

The modified SA approach that drives the MCA may be adapted in order to more com-

prehensively explore the parametric space of a BN model. Indeed, provided the output of a sensitivity analysis technique can be captured using pairwise Normalised Mutual Information (NMI) scores, any valid SA approach could be substituted for the specific approach introduced in Chapter 4. From this perspective, the MCA technique provides a general approach for supporting the targeted assurance of BN models through the utilisation and extension of SA.

A final contribution of the MCA technique is in demonstrating the potential value and practicability of adapting existing BN and AI analysis techniques to support safety-focussed analyses of BN models. Rather than attempt to adapt existing safety techniques with the aim of addressing the assurance of BNSs, it may be more useful to instead adapt BN techniques for targeted evaluation activities.[21]

6.3.4 Application of the Evidence Framework

The evidence framework introduced in Chapter 5 was developed using existing sources from within the safety engineering domain. The resulting framework is therefore intended to be general, and should be used to characterise and classify any items of evidence generated over the course of the development of an AIS. However, the scope of this project prevented the exploration of other avenues of research associated with the characterisation and classification of evidence generated from other AI approaches.

In general, within the AI domain there is a need to provide a more comprehensive basis for the description and *understanding* of evidence drawn from the plethora of evaluation activities and metrics used within the AI domain. From an assurance perspective, the utility of these techniques can be hard to translate into useful information about the safety and security of a system, and the relative value of this evidence can similarly be hard to discern.

This contribution therefore takes a small step towards a more expansive and robust framework for managing evidence in the context of AISs. It provides a standardised approach to classifying evidence generated during the development of a BNS that can be used to construct a hierarchy of evidence, and to communicate the relative value of evidence between assurance practitioners and BN developers.

In the context of the case study, the application of the evidence framework was straight-

[21] This is likely to be particularly important for Model Viewpoint and Data Viewpoint aspects.

forward given the generation of the system-specific RM-BNS objectives. By directly mapping the evidence classes onto objectives and relating these to the criticality of a given objective, it enables a clearer understanding of the role and availability of evidence for that objective. During the development of the case study, it became clear that the evidence framework can also be used to prioritise the selection of analysis techniques to maximise the confidence gained with respect to the satisfaction of a given objective. This further contributes to the aim of providing a targeted approach to the assurance of BNSs as outlined in the thesis hypothesis.

6.3.5 Application of the Proportionality and Sufficiency Concepts

In the context of AISs, establishing the proportionality and sufficiency of efforts to assure an AIS may prove to be the most challenging single aspect of the assurance of a given system. In the language of the RM-BNS, answering the question: 'How much is enough?' for almost every Viewpoint is a technically challenging proposition. This is particularly true for the 'Operational', 'Model' and 'Data' Viewpoints.

For example, the sufficiency of available data is an omnipresent issue in the field of AI in terms of quality, quantity and the underlying patterns that will be exploited by the system. Likewise, the modelling process for any AIS is typically fraught with intuition-driven decision making and ad hoc development practices that make the rigorous evaluation and assessment of the sufficiency of these activities challenging prospects for assurance practitioners. These insights could be used to help build up a compelling justification for the sufficiency of available data-centric aspects, and for the adequacy of a BN model for a particular application.

The contributions of Chapter 5 are aimed at taking a step towards mitigating some of these difficulties. By integrating the contributions of the previous chapters with the previously mentioned evidence classification framework, the aim of this chapter was to establish a conceptual basis upon which to more rigorously tackle these challenges. While the discussion focussed most heavily on model-centric considerations of assurance, the concepts introduced and discussed in this chapter should be extensible to other system aspects.

The case study indicates that the application of these ideas is tractable for a system similar to the AMTS. It shows how the two approaches to utilising the RM-BNS objectives may be applied to a BNS, and further reiterates the apparent benefits of utilising

an approach that varies evidence in the satisfaction of both the RM-BNS objectives, and for assuring a BNS in general. However, from a practical perspective, the utilisation of the objective-varying approach to establishing sufficiency may overcome some of the (potentially significant) operational overheads associated with managing the large number of system-specific objectives that may be generated using the RM-BNS approach. By only considering subsets of objectives as defined by the criticality of a system aspect, assurance practitioners adopting this approach may need to manage a much smaller number of objectives – thereby reducing the complexity of the assurance challenge from one perspective. However, this should be carefully considered with respect to the loss of information that will accompany the adoption of such an approach and the resulting effect on the confidence of a BN developer in their system.

More broadly, the objectives themselves necessarily lose information about the system they are applied to. They can never define and address a *truly* exhaustive set of assurance concerns. The adoption of objectives with the understanding of the potential imperfection of these objectives is standard practice within the safety domain. Objectives are defined such that they address aspects of a system's design and development that are perceived to be the most pressing assurance concerns, or the most likely sources of error. This is also true for the RM-BNS objectives: they have been defined to address aspects of BNSs that are commonly discussed as being the most probable sources of error within the BN literature, and based upon the operational experiences of AI developers using other roughly comparable techniques.

However, it is likely that the 'variable evidence' approach to the assurance of BNSs will be most appropriate for many applications. As discussed throughout this thesis, the assurance of a BNS will be predicated upon the assessment of a broader range of safety evidence and unconventional design decisions and will demand a more holistic approach to assurance than is typical for software systems. Evidence driven approaches should more readily accommodate the iterative nature of BN development and limit the risk of extensive redevelopment work in the event that the criticality of a system aspect changes over the course of the development, or other design decisions dramatically alter the scope and role of the system.

Finally, the objectives and concepts introduced in this thesis have been defined with the aim of minimising the worst-case outcomes for a BNS. The properties of a BNS and the scope of the applications they are commonly applied to introduces serious concerns

related to assurance deficits. The objectives have been defined to attempt to minimise these assurance deficits by addressing the most serious potential shortcomings of a BN development programme. Prior to this thesis, no existing work has unified these ideas into a single document.

6.3.6 Scalability of Application

The case study presented in this chapter outlines the application of the methodologies and concepts presented in this thesis to a relatively simple BNS. The process of applying these ideas to this system was relatively straightforward. However, as discussed in Section 6.2.7.1, depending on the approach to establishing the sufficiency of evidence selected, the number of RM-BNS objectives that were generated (and therefore must be satisfied) can vary dramatically. This appears to be the principal challenge to the scalability of the framework and techniques presented here.

In the case of this case study, more than 200 objectives for aspects pertaining to the Model Viewpoint alone were produced. While many of these objectives would be relatively straightforward to address, much larger BNSs may ultimately produce many thousands of objectives. In general, it is likely that the number of objectives generated for system aspects related to the Model Viewpoint will grow linearly in the number of variables within a network.

For example, another widely studied BN model – the MUNIN network – consists of over 1000 variables [208]. This is therefore significantly larger than the 37 variable ALARM model presented previously and represents one of the largest publicly available models. Applying the technique proposed in this chapter may produce over ten thousand unique objectives. This may make the RM-BNS and its associated evidence framework unattractive to practitioners attempting to tackle the problem without any degree of automation. In such cases, the challenge of managing and processing evidence associated with each objective would be possible but highly laborious, particularly in low-criticality applications.

However, simply having a large number of objectives is not sufficient to abandon the them and the associated framework. In high-criticality applications, the utility of the contributions of this thesis should outweigh their cost in labour, and there are clear opportunities for the automation of objective and evidence management within the proposed framework. Indeed, it may be possible to frame some of the Model Viewpoint objectives

as more akin to test objectives for which an automated test harness could be constructed for a given application.

Ultimately, there are no known theoretical limitations on the scalability of the contributions made within this thesis, though there may arise practical scalability challenges related to the sheer quantity of information that could be generated through their application to a complex BNS. However, large quantities of assurance data are not uncommon in the development of mission critical systems, and in many cases these difficulties could be resolved or mitigated through the development of new automated tooling and testing strategies based upon the frameworks and techniques presented here.

6.4 Evaluation Against Thesis Hypothesis

The thesis hypothesis introduced in Chapter 1 was defined to address a number of assurance challenges that were defined early in the course of this research project. This section looks at each of the main aspects of the hypothesis and evaluates them in the context of the final contributions of this thesis.

6.4.1 Targeted Assurance

In Chapter 2, the potential shortcomings of existing safety standards and guidelines when applied to a BNS were introduced. The development of a BNS for mission-critical applications must mitigate these shortcomings by targeting the BNS- or AIS-specific system aspects. This was the aim of the framework introduced in Chapter 3: to provide an approach to describing and modelling BNSs in a format that comprehensively captures all aspects of the system that are pertinent to the system's functional behaviours. Consequently, the RM-BNS framework provides a flexible approach to representing BNS architectures that should ensure the unconventional aspects of the system are known to assurance practitioners, and the roles of these aspects are analysed from a safety-centric perspective.

The verification and validation objectives also introduced in Chapter 3 build on this framework. The objectives are designed to ensure that BNS-specific assurance concerns are explicitly targeted by assurance practitioners. The case study outlines how the RM-BNS framework and objectives can be used to ensure that system-specific aspects of a BNS are targeted by assurance activities, and that the assurance of two superficially similar

system-specific objectives may require the use of different analysis techniques and may involve fundamentally different assurance considerations.

The need to target the unconventional aspects of BNSs from an targeted assurance perspective was further extended in Chapter 4. This was achieved through the introduction of the Model Criticality Analysis (MCA) methodology that was designed to support the targeted analysis of system aspects associated with the Model Viewpoint. Practically, the MCA approach has been developed to ensure that the effects of a BNS's model dynamics are more transparent to assurance practitioners.

As indicated in the case study, the effect of this analysis is to avoid the need to treat a BN model as a black-box, or to otherwise take an indiscriminate approach to evaluating models and their constituent variables. Instead, the MCA is intended to support the targeted analysis of BN models and provide proportionate, targeted analysis of these models. Furthermore, it is not aimed at addressing typical model performance considerations, but rather supporting the evaluation of the role of individual models and variables on safety-related functional behaviours.

Taken together, these contributions provide a comprehensive framework for generating a highly system-specific representation of BNSs that includes information on the role and type of evidence, the criticality of a system aspect and system-specific assurance considerations related to that aspect. This representation can be used to identify specific parts of this representation that are likely to be of interest to assurance practitioners. An assurance practitioner can pinpoint a specific BN model variable within a BNS of interest and quickly extract information about the safety implications of that variable, the associated strengths and weaknesses of the available safety evidence, and the variable-specific assurance considerations that must be addressed. From this perspective therefore, the contributions of this thesis achieve this aspect of the thesis hypothesis.

Perhaps most importantly, the assurance of a BNS will require something of a paradigm shift with respect to existing software safety standards: the primary determinant of a system's functional behaviour often lies in the *mathematical models* encoded within the system. Current work on the assurance of BNSs has been piecemeal, and other work on alternative AI approaches has generally been narrow in scope. The contributions of this thesis in providing a comprehensive overview of BNS assurance therefore provide a first step towards enabling assurance practitioners to understand precisely *what* they need to consider when assuring a BNS.

6.4.2 Analysis and Evaluation of Underlying Probabilistic Models

A principle weakness of existing safety guidance in the context of BNSs was identified in Chapter 2 as being the role of models in the functional behaviours of this class of system. In Chapter 3, the shortcomings of assurance practices in identifying system failure modes arising as a consequence of errors in models was outlined. In both chapters, the role and importance of these system aspects was explored. In Chapter 2, the potential concerns associated with using existing BN model analysis techniques without the introduction of assurance information into these analyses was discussed. A case for the need to direct assurance efforts towards explicitly addressing these aspects was introduced in Chapter 3.

A first contribution towards the integration of assurance information into the analysis and evaluation of BN models was then introduced. The MCA approach provides a mechanism for both modifying existing BN model analysis techniques to account for safety considerations (e.g. focus on worst-case influence rather than average influence etc.), and the explicit inclusion of safety information into these techniques. This approach is therefore intended to provide a targeted analysis of a BN model's dynamics, and the potential effect of these dynamics on the behaviour of the BNS.

Finally, Chapter 5 introduced a number of assurance techniques that could be used to analyse and evaluate the models used in a BNS. The techniques are intentionally discussed in relatively general terms in order to capture the broad range of possible approaches currently available in literature. As with the modification of a standard Sensitivity Analysis (SA) approach into Chapter 4's MCA technique, many of these techniques could be adapted to more specifically address safety considerations. Chapter 4 also aims to stress the importance of establishing coverage of BN model aspects as an independent assurance process that is distinct from conventional software coverage activities.

6.4.3 Analysis and Evaluation of Underlying Data Artefacts

The scope of this thesis and the nature of the motivating example for this work has limited the amount of research that could be performed into the analysis and evaluation of data artefacts used both operationally and during the development of mission-critical BNSs. Furthermore, the focus of this thesis on model aspects has been spurred - in part - by the increasing acceptance of the need to adopt alternative approaches for data-intensive systems that has emerged over the course of the compilation of this work. This is perhaps best captured in the publication of the Data Safety Initiative Working Group's Data Safety

Guidance. While due to its scope this guidance does not rigorously explore the role of data in AISs from an AI-centric perspective, it does provide some initial guidance that overlaps conceptually with many of the concerns detailed in Chapters 2, 3, and 5 of this thesis.

However, the RM-BNS introduced in Chapter 3 does provide a means of performing qualitative analyses of the role of data artefacts in a given system. It also provides an approach that aims to explicitly capture the role of data gathering activities in the development and assurance of BNSs. This is likely to be the most challenging (and expensive) aspect of the development of mission-critical BNSs generally. In commercial and non-mission-critical applications, the need to rigorously evaluate all channels through which data is acquired and processed prior to use in the training of a given system is often diminished: there is currently limited work on the active analysis and evaluation of data and the auditing of data gathering practices to ensure the absence of accidental or deliberate errors in acquired data that may produce 'undesirable' emergent behaviours in the system using models trained on this data. This includes manipulation of data that may be used to train a BNS (i.e. attacks on BNSs that are effectively executed prior to the implementation of the system). This may have significant safety implications for the completed system. The RM-BNS therefore aims to ensure that assurance practitioners understand how errors introduced as a consequence of flawed data gathering activities may propagate through their system to produce erroneous functional behaviours.

Chapter 4 touched on an important by-product of the MCA technique. By establishing the criticality of individual variables within a model, and of models within a BNS, MCA can be used to guide assurance practitioners to identify aspects of their data that are used by high-criticality variables. While the MCA has not been extended to account for a notion of data criticality, such a notion may be developed based upon the results of this approach. This may be used to develop additional targeted analysis and evaluation techniques that explore the role of data in a mission-critical BNS. However, the thesis has identified and discussed in detail the need to understand the role of data in BNSs, and the MCA technique and the RM-BNS framework introduced here could be used as the basis of additional work in this area.

On a more general note, the Data Viewpoint aspects of the RM-BNS are potentially the most challenging aspects of a BNS to assure. The infrastructure and processes that will be required to develop a data-driven BNS for some applications will be extensive, and

will necessitate careful review of a number of 'Human Factors' considerations during the acquisition and management of data for such a system [61]. This will be further heightened for those organisations developing BNSs that utilise data from distributed sources, or for highly novel applications. Developing a robust set of data acquisition processes and the corresponding infrastructure may prove to be a limiting factor in the ability of an organisation to deploy a BNS.

6.4.4 Analysis and Evaluation of Underlying Computational Techniques

As with analysis and evaluation techniques targeting data aspects, the scope of the thesis has forced the deprecation of research into the development of approaches for analysing and evaluating computational techniques used in mission-critical BNSs. As with the data aspects, Chapters 3 and 5 have introduced the need to evaluate the role of learning and inference approaches in the behaviour of a completed BNS. In particular, the RM-BNS viewpoints introduced in Chapter 3 aim to ensure that the properties of a selected BN learning technique are understood, and the potential safety implications are known to assurance practitioners. This once again provides a framework for the qualitative analysis of the role of learning and inference approaches, and the interactions of these aspects with system aspects associated with the RM-BNS's Model and Data Viewpoints.

6.5 Conclusion

This chapter has demonstrated that the contributions of this thesis provide new mechanisms for facilitating the targeted assurance of the unconventional aspects of BNSs. It has provided a limited demonstration of these techniques on a case study, and evaluated the results of this case study against the thesis hypothesis and other practical considerations. This chapter – and the thesis generally – provide a first centralised overview of the assurance challenges of BNSs, and provide guidance on how some of these assurance challenges can be identified and addressed. Ultimately, it will be through the practical application of these ideas to a BNS that the effectiveness and utility of these contributions will be established.

Chapter 7

Conclusion

7.1 Summary of Thesis Contributions

This thesis has made the following contributions:

- It has identified the key challenges for the assurance of BNSs and identified a number of important distinctions between conventional software systems and BNSs. These insights have been condensed and organised into a structured set of viewpoints on BNSs that define these challenges and can be used to support targeted assurance efforts.

- It has provided a new, structured, comprehensive framework for describing and analysing BNSs. This framework has been designed to expose the interactions between disparate aspects of BN models and to help ensure BN developers and assurance practitioners can communicate effectively about the properties of a BNS. This framework was then used to generate a set of flexible verification and validation objectives that can be used to target and comprehensively address assurance challenges for BNSs.

- It has provided a new technique for performing targeted analyses of BN models, and outlined how this can be used to establish the criticality of model components within a BN model.

- It has provided an approach for describing and understanding safety evidence generated during the development of a BNS. It has provided guidance on the assurance of BNSs that may be used to support the satisfaction of verification and validation ob-

jectives and to address the challenge of sufficiency of evidence and assurance efforts in the development of BNSs used in safety-critical applications.

This chapter is structured as follows. An overview of the the key contributions of this thesis is provided. This overview discusses the novelty and necessity of the work introduced in this thesis. Next, it gives a short summary of potential limitations of this work in the context of the application of the techniques and concepts to the deployment of a BNS in a safety-critical role. The following section then builds upon this to suggest a number of potential directions for future work. Finally, the chapter concludes with some high-level considerations for assurance practitioners working towards the deployment of AISs in safety-critical roles.

7.1.1 Contribution - A Multi-Viewpoint Approach to BNS Assurance

Early in the course of the research carried out for this research project, it became clear that there were no existing approaches for explicitly communicating the distinctions between conventional software systems and BNSs to both safety practitioners and BN developers. The recognition of the necessity for a framework that supports the omnidirectional communication of BN-specific assurance concerns was therefore the motivation for the definition of the RM-BNS and its associated verification and validation objectives.

At the heart of the RM-BNS framework are the six proposed viewpoints on BNSs. These viewpoints have been defined to capture and communicate the full range of concerns that assurance practitioners will face when attempting to assure a mission-critical BNS. As discussed throughout this thesis, assuring a BNS will demand a more holistic approach to software assurance than is common for many existing classes of software system. For example, the role of the environment, the nature of the underlying mechanics that drive functional behaviour of a BNS and the role of ancillary activities and artefacts that may otherwise be regarded as outside the scope of a system's definition (e.g. data gathering activities) must be carefully considered in the context of BNSs.

Indeed, describing the assurance of a BNS (and AISs more generally) as a software assurance problem in the conventional sense may be misleading in the first instance. The assurance of AI is currently being approached as the assurance of a software system *with* unconventional aspects. However, the complexity of this class of system does not typically lie in the software design and implementation itself. As has been discussed in this thesis, the complexity typically lies in the models, data, environment and operational role of the

system being developed. Conventional software engineering concerns remain important but are unlikely to be the overriding focus of assurance activities for this class of system. Therefore, it may be more appropriate to describe the assurance of a modern AIS as a new class of system with software aspects, or perhaps as the assurance of a new form of software altogether.

Assurance practitioners working towards the assurance of these systems must consider a broader range of concerns than in conventional software systems. Work in other fields using alternative AI techniques repeatedly highlights the need to consider aspects of an AIS that may be overlooked if it were to be treated as a conventional software assurance problem. The proposed viewpoints and their constituent views therefore provide a basis for describing and communicating the properties and assurance concerns of BNSs and could be further generalised to accommodate other classes of AIS.

7.1.2 Contribution - Reference Model for Bayesian Network-based Systems

Chapter 3 introduced the RM-BNS framework for describing and qualitatively modelling BNSs. This framework was designed to expose the full range of challenges that assurance practitioners may face over the course of the development of a mission-critical BNS. As discussed in section 7.1.1, at the core of this framework are the six viewpoints on BNSs. The modelling framework built upon these viewpoints was developed with the aim of explicitly capturing the *interactions* between system aspects. Indeed, many of the most challenging design decisions during the development of a BNS arise during interactions between *Models* and associated *Computations* (for example the selection of inference algorithms with respect to the models the algorithm will operate on), or between *Data Artefacts* and *Models*, where the available data may determine certain model aspects, or vice versa. The case study in Chapter 6 indicates how the RM-BNS framework directly exposes precisely these sorts of interactions.

There are a number of prominent, existing reference models that have been successfully utilised in the development and analysis of complex systems. Indeed, the RM-BNS was inspired by the RA-SDS framework utilised by NASA for modelling complex space systems. The utility of these frameworks is well documented in systems engineering literature, and a number of architectural frameworks are in active use in several domains – though no such reference models existed for the description of a BNS prior to the work presented in

this thesis. The aim of the RM-BNS framework is therefore to provide a similar level of utility to BNS developers and assurance practitioners as other reference models, though this will of course require engagement with practitioners and BN developers to refine and expand the RM-BNS framework to meet their operational needs and to reach the standard of the aforementioned frameworks. However, the RM-BNS framework provides a first step towards a systematic basis for qualitatively modelling and describing BNS system architectures, and for capturing the important interactions that may exist within these systems.

Finally, the RM-BNS framework was used to help structure and define a set of generic verification and validation objectives for BNSs. These objectives also represent a novel contribution – no comprehensive set of assurance objectives has previously been produced for BNSs. The process of refining and satisfying the objectives was outlined in Chapters 5 and 6, and the case study presented in Chapter 6 indicated the utility of the approach for an example system. Further evaluation of the objectives in the context of 'real-world' systems will be necessary to establish their flexibility and utility from an assurance perspective. However, in their current form, the objectives provide a first step towards generating a comprehensive, flexible set of objectives for the assurance of a BNS that could be built upon in future work.

7.1.3 Contribution - Assurance-Focussed Model Analysis

The analysis technique introduced in Chapter 4 was developed with the aim of providing a technique for transparently evaluating BN models from an assurance-driven perspective. As defined in Chapter 4, Model Criticality Analysis (MCA) integrates concepts from both the AI domain and the systems engineering domain to produce a technique that can be used to map criticality metrics to abstract model aspects. As shown in the case study, this can be used to provide targeted assurance of model-centric aspects and to identify parts of a BN model that may be of particular interest from an assurance perspective.

More generally, it can be used to avoid the need to treat a BN model as a black box and, by extension, to treat all variables within the model as equally important from a safety perspective. On a practical level, this may produce intractable problems in some circumstances. Instead, assurance practitioners can target specific models and model components and address assurance concerns in proportion to the criticality of the model aspect. The development of techniques such as this is essential if assurance practition-

ers are to effectively expose and address the model-centric assurance concerns associated with the deployment of BNSs (and AISs more generally) in safety-critical roles. No technique existed prior to this work that explicitly integrated assurance information into BN evaluation techniques for the purposes of assuring the dynamics of a BN model.

At a high-level, Chapter 4 and the application of the MCA technique in Chapter 6 indicate the utility and practicability of modifying existing AI evaluation and analysis techniques to develop new, targeted, information-rich assurance-focussed analysis techniques. Indeed, going forward more developments of this type will be needed within the safety domain if the challenges associated with establishing *model coverage* and the need for *model assurance* are to be addressed. This will be explored more in the following section.

7.1.4 Contribution - Evidence Framework and Sufficiency Concepts

In Chapter 5, a novel approach to describing and classifying evidence produced by techniques utilised during the development of a BNS was proposed. The aim of this approach was to capture the hierarchy that exists within evidence, and to do this in a format that would enable communication of this information and the outputs of a particular analysis technique between BN developers and assurance practitioners. Prior to this guidance, no work had been performed into describing the output of BN-specific analysis techniques using assurance-focussed language. The proposed approach therefore provides a first step towards a systematic basis for describing and evaluating BN-specific development activities with respect to their assurance implications. An approach such as this has not previously been defined for BNSs.

Alongside this framework, Chapter 5 also provided guidance on possible approaches to establishing the sufficiency of assurance efforts for mission-critical BNSs. As with the evidence framework, there existed no prior guidance on how this may be achieved for a BNS. The guidance provided in Chapter 5 therefore provides a step towards understanding how the sufficiency of assurance activities may be established for BNSs and specifically what assurance practitioners should consider when addressing system aspects associated with the *Model* viewpoint.

7.1.5 Limitations of Thesis Contributions

While the work presented in this thesis has been defined with the aim of providing a general set of techniques and tools for developing BNSs for safety-critical applications, the contributions presented here have been developed primarily with the motivating use-case of this research in mind. Consequently, the work presented within this thesis has not been extensively reviewed in the context of other potential BNS applications or BN variants.

In certain contexts, practitioners may therefore encounter some disparities between the RM-BNS framework and its associated objectives, and the demands of their specific use-case. This may also be true for the MCA and evidence classification approaches introduced in Chapters 4 and 5 respectively. However, the contributions of this thesis are intended to be sufficiently flexible that practitioners encountering these issues should face little difficulty in adapting or extending these contributions to meet their needs.

Secondly, though the research presented here has been developed in collaboration with BAE Systems, the techniques introduced and defined within this thesis have not been tested as part of an active BNS development programme. As with any technique developed in relative isolation from their intended use, the contributions of this thesis will need to be evaluated as part of a broader engineering effort.

Finally, as discussed in Chapter 4, the MCA technique was developed to provide a 'baseline' analysis technique. The sensitivity analysis (SA) technique selected is among the most simple SA techniques in use, and has known limitations with respect to remaining a viable technique for larger BN models and with respect to the exploration of a model's dynamics. However, the aim of Chapter 4 was to establish a basis upon which more advanced techniques could be developed. The need for additional research in this direction will be discussed in more detail in the following section.

7.2 Future Work

The research conducted over the course of this project has identified a number of research avenues. The contributions of this thesis are novel within the assurance field, and therefore many of these avenues represent expansions or refinements of the work presented here. However, some represent general research avenues that need to be explored if the assurance of a BNS is to be achieved.

7.2.1 Generalisation of Viewpoints and RM-BNS

As discussed in section 7.1.1, the RM-BNS viewpoints have been defined to address the assurance concerns specific to BNSs. They were defined to be relatively general, and therefore should accommodate many different BN variants, architectures and applications. However, they may also be generalised further to accommodate AISs generally. Indeed, there is a need for a common systematic, comprehensive framework for structuring and describing the assurance challenges faced by AI developers and assurance practitioners. The viewpoints and their constituent views could therefore be expanded and generalised to capture the assurance challenges of AISs, and perhaps even to provide a conceptual basis for developing guidance and standards for the assurance of AISs.

Some work was performed to this end over the course of this research project, though it remained at very early stages of development. Future work in this area could look at the applicability of the proposed viewpoints to other classes of AISs, and how they may be refined or otherwise modified to address the considerations of other AI techniques or applications. The application of the viewpoints to other systems and BNSs could also be attempted to ensure their flexibility and utility in a practical setting.

7.2.2 Extension of Model Criticality Analysis

The MCA technique demonstrates the potential utility and practicability of adopting and adapting existing BN evaluation and analysis techniques for assurance-focussed uses. However, as previously discussed, the technique as outlined in Chapter 4 will run into limitations in terms of its ability to scale to large BN models, and in its ability to analyse more complex interactions within a BN model. As indicated in Chapter 4, the results of a MCA should be considered as a simple approximation of the underlying dynamics of a given BN model. For some models, this will be adequate. The MCA approach as outlined in this thesis should therefore be considered to be a 'baseline' approach that should be actively improved and extended.

For example, future work could explore the utilisation of more advanced sensitivity analysis techniques in place of the simple approach adopted in Chapter 4. More sophisticated techniques can be used to explore more interactions between variables within a BN model, and may be able to do so more efficiently than the 'baseline' approach. This may give a more extensive exploration of the dynamics of a BN model. As discussed in Chapter 6, any sensitivity analysis technique that can produce the metrics introduced in Chapter

4 could be used in place of the proposed SA technique. Another avenue of research could be the exploration of the application of the MCA approach to hybrid BN models and any strengths or weaknesses of the technique in these cases. Finally, the potential value of alternative metrics could be assessed and may provide further improvements to the 'baseline' technique introduced in Chapter 4.

7.2.3 Adaptation of BN Evaluation Techniques and Metrics

Beyond the MCA technique, there are other techniques that could be co-opted for BN-specific assurance-focussed analyses. Indeed, there may be scope for developing general-purpose adaptations of other techniques for use across multiple AI modelling approaches. For example, the Receiver Operating Characteristic (ROC) outlined in Chapter 5 provides an approach to analysing and quantifying the diagnostic performance of BN models. However, it can also be used generically to characterise the *classification* performance of other AI representational frameworks. A standard approach to integrating safety or security information into analyses such as these could enable these techniques to be used across AISs. For example, a safety-enhanced version of the ROC may be used to evaluate a range of systems, including those driven by BNs, Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs).

7.2.4 Refinement of the RM-BNS and Objectives

As discussed in section 7.1.2, the RM-BNS (and its associated objectives) represents a first step towards providing a structured, comprehensive basis for modelling and assuring BNSs. However, both the reference model and the objectives have not been tested on a 'real-world' project to evaluate their relative merit and limitations. Future work could therefore look into refining and enhancing the RM-BNS by evaluating its use during the development of a BNS and produce an updated version that reflects the input of BN developers and assurance practitioners working on 'real-world' BNSs. This could produce a more mature 'field-tested' RM-BNS that may be more readily adopted by assurance practitioners.

7.2.5 Role of Data in AI

Much of this thesis has focussed on assurance concerns related to the RM-BNS *Model Viewpoint*. Consequently, only a comparatively limited amount of research was performed

on system aspects associated with the other five RM-BNS Viewpoints. Of these five, perhaps the most rich avenue of research is likely to be around the utilisation of data in BNSs. Indeed, there are a number of commonalities between the assurance concerns associated with the *Data Viewpoint* and the concerns expressed in the Data Safety Initiative Working Group's Data Safety Guidance (DSG) document [144]. There may therefore be scope to more closely integrate the language and concepts outlined in Chapters 3 and 5 with those outlined in the DSG document.

More generally, the assurance implications and considerations associated with the processes used to gather, manage and store data used in AISs, and the properties of the data artefacts used to train and design a mission-critical AIS are still relatively unexplored. Furthermore, while the DSG document does provide some guidance specifically related to data used for Machine Learning (ML) applications, this guidance is comparatively limited. Indeed, there may be significant value in producing a DSG-like document specifically targeting AI-specific concerns. This could perhaps build upon the concepts introduced in Chapter 3 and outlined in the RM-BNS *Data Viewpoint*.

7.2.6 Addressing System Evolution

Early in the course of this research, the problem of *distributional drift* became apparent as a key assurance concern for BNSs. Within the AI field, distributional drift refers to situations when the representational framework and representational instantiation (i.e. model) at the core of an AIS begins to diverge from 'reality'. More concretely, the *learned* representation of the world may become obsolete as the operational environment of the system evolves over time. This may occur through planned changes to a BNS platform or the environment within which it operates, including integration of new sensors, datastreams or deployment to a new operational theatre, for example. It may also occur through unplanned changes, such as changing weather patterns, demographics or other environmental factors. In both cases, the performance of a BNS will fluctuate over time.

Many mission-critical applications will demand that the operational performance of an AIS is maintained over an extended period. An important avenue of research is therefore into effective ways of monitoring the *evolution* of a BNS with respect to its environment over time, and how these monitoring approaches may be integrated with safety or security information. One approach that was explored briefly during this research was the notion of developing a second standardised AI monitoring tool to track the performance of the

first AIS. It was envisioned that this would provide on-line validation and monitoring capabilities and may flag when the system required 're-educating'. The initial idea was to develop a second BN model to diagnose the performance of the first, though this was ultimately abandoned as it merely moved the assurance challenge from one network to another, without resolving the underlying technical challenge. However, there may be value in this approach if a standardised method for diagnosing distributional drift (and other evolution-driven errors) across distinct models can be devised.

7.3 Closing Remarks

The assurance of modern BNSs in safety-critical roles will require the adoption of techniques and concepts currently alien to the broader safety engineering community. The most radical conceptual distinctions between conventional software systems and AI systems lie in the role, capabilities and limitations of the representational frameworks (i.e. models), the data artefacts that drive them and how they may interact with the world around them. Being able to identify, describe and address these distinctions is an important step for the assurance of BNSs, and AISs more generally.

Of all the diverse areas of AI research, research into the assurance of AI must be unwaveringly clear on what *precisely* makes these systems different from conventional software systems and how these specific aspects can be addressed. This was recognised in work on the assurance of rule-based AI Systems in the 1980's, as Rushby states [86]:

> *"... the best way to develop credible and effective ... assurance and evaluation techniques for AI software will be to identify the facets of such software that are inherently, or essentially, different from conventional software, and to distinguish them from those facets that are only accidentally or inessentially different."*

The aim throughout this work has been to introduce concepts and methodologies that can be used to expose and address BNS-specific assurance challenges. There is no extant assurance-focussed work on these problems in the context of BNSs, and still comparatively limited technical work on the assurance of AISs more generally. Prior to this thesis, no single document provided a unified approach for addressing the assurance challenges related to the development of a BNS. The concepts and techniques presented here therefore

represent a first step towards the assurance of BNSs, and may help shape the thinking of assurance practitioners working within the field of AI assurance more generally.

Appendix A

A Reference Model for Bayesian Network-based Systems

This appendix contains a full enumeration of the constituent parts of the RM-BNS framework. It is broken into three sections: Viewpoints, Objectives and Reference Model.

A.1 RM-BNS Viewpoints

Table A.1 shows the full set of RM-BNS Viewpoints with their associated descriptions and objects.

A.2 RM-BNS Objectives

Table A.2 enumerates the full collection of RM-BNS Verification and Validation objectives.

A.3 RM-BNS Reference Model

Figure A.1 shows the RM-BNS reference model.

Table A.1: RM-BNS Viewpoints and Objects.

Viewpoint	**View**	**Description**	**Objects**
Model	Structure	Concerned with the structure (local and/or global) of a model.	Model
	Parameters	Concerned with the properties (e.g. confidence, quality etc.) of all parameters and hyperparameters used in a model.	Model
	Definition	Concerned with aspects of the model associated with the qualitative aspects of representation, including the context of the model and the definition of model components.	Model
	Dynamics	Concerned with the high-level properties and dynamics of a model.	Model
Data	Acquisition	The acquisition view defines the sources, processes and personnel used to obtain all data artefact.	Data Artefact, Process
	Transformation	This view describes all processing (e.g. normalisation, discretisation) applied to data artefacts.	Process
	Management	The management view describes the databases and assorted management activities used to store/archive and transfer data artefacts. It addresses data-specific aspects of configuration management (e.g. training and evaluation datasets).	Process

Table A.1: RM-BNS Viewpoints and Objects.(continued).

Viewpoint	**View**	**Description**	**Objects**
	Artefact	This view describes the properties of each resulting data artefact (e.g. the quality and integrity of an artefact). It is particularly concerned with identifying and addressing sources of uncertainty in the artefact/s.	Data Artefact
Computation	Optimisation	This view is concerned with all aspects related to the optimisation (learning) algorithms utilised in both the development and deployment of a BNS.	Computation
	Inference	This view is concerned with all aspects related to the inference algorithms utilised by a BNS during operational use.	Computation
	Data Flow	The data-flow view describes the passage of information through the system. This is aimed at addressing concerns that may arise through the use of certain classes of hierarchical or ensemble modelling approaches.	Computation
Technology	Infrastructure	This view defines the tools, processes and resources needed to develop the system.	Infrastructure
	Standards	This view is concerned with all standards relevant to the system and its operation.	Framework

Table A.1: RM-BNS Viewpoints and Objects.(continued).

Viewpoint	**View**	**Description**	**Objects**
	Frameworks	The framework view describes the modelling frameworks that will be used in the system. This includes Knowledge Engineering and any BN development frameworks, and the implications of these frameworks for the BNS (e.g. performance trade-offs, limitations etc.).	Framework
	Risk	The risk view is aimed at addressing BN technology-specific developmental and operational risks. It is concerned with identifying any risks arising from developing a BNS, in terms of both the project itself (e.g. time and expense), and any novel cultural, environmental, safety and/or security risks that may be introduced through the development and operation of a BNS.	Framework, Infrastructure
Operational	Requirements	This view represents concerns associated with establishing objectives for the system, and capabilities that must be achieved.	Capability
	Scenario	This view is concerned with how the system will be deployed, as well as demonstrable and potential latent properties of the system in those contexts.	Scenario, Capability

Table A.1: RM-BNS Viewpoints and Objects.(continued).

Viewpoint	View	Description	Objects
	Evolution	The evolution view refers to concerns associated with the change of the target platform over time, both in the terms of engineered platform change, and change due to age, operational tempo or location etc. See also OV-4 and OV-5.	Environment, Scenario
	Environment	Environmental concerns include the effect of ambient environmental aspects, including atmospheric properties and weather.	Environment
	Maintenance	This view is concerned with maintenance practices used with respect to the system, and the target platform. It is aimed at addressing concerns related to the standardisation of maintenance practices across all sites/operational theatres and the elimination of maintenance practices that may result in degraded performance of BNSs (e.g. introducing errors due to distributional drift).	Environment, Capability
Implementation	Software	This view is concerned with 'conventional' software aspects of the system. This includes all typical software design, implementation and testing concerns.	Function, Resource

Table A.1: RM-BNS Viewpoints and Objects.(continued).

Viewpoint	View	Description	Objects
	Hardware	This view addresses any hardware aspects of the system. In particular, sensors used to provide input into BN models and any pertinent features of the target platform/ environment into which the BNS is to be deployed including memory and compute power.	Function, Resource
	Architecture	The Architecture view addresses concerns associated with the high-level system architectural aspects of a BNS. This includes all 'conventional' system design considerations, including the top-level allocation of functionality to hardware and software components.	Function
	Integration	This view is aimed at addressing all activities necessary to – for example – transfer a system from a simulation environment into a target platform.	Function, Resource

Table A.2: RM-BNS Objectives.

Viewpoint	**View**	**Identifier**	**Objective**
Model	Structure	MV-1.1	Establish and justify the basis for using the [structural variant] for the [Model].
		MV-1.2	Establish and justify the any assumptions in the structure of the [Model].
		MV-1.3	Establish and justify the implemented structure of the [Model].
		MV-1.4	Establish and justify confidence in the structure of the [Model].
	Parameters	MV-2.1	Establish and justify the precision of the parameterisation of [Model].
		MV-2.2	Establish and justify the accuracy of the parameterisation of [Model].
		MV-2.3	Establish and justify any assumptions in parameterisation of [Model].
		MV-2.4	Establish and justify the confidence in the parameterisation of [Model].
	Definition	MV-3.1	Establish and justify the definition of the [Model].
		MV-3.2	Establish and justify any assumptions in the [Model] definition.
		MV-3.3	Establish and justify the context and scope of the [Model] definition.
		MV-3.4	Establish and justify the confidence in the [Model] definition.
	Dynamics	MV-4.1	Establish and justify the dynamics of the [Model].
		MV-4.2	Establish the justify any assumptions in the dynamics of the [Model].
		MV-4.3	Establish and justify any constraints on the dynamics of the [Model].
		MV-4.4	Establish and justify the necessity of the [Model] dynamics.

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
		MV-4.5	Establish and justify confidence in the dynamics of the [Model].
Data	Artefact	DV-1.1	Establish and justify the type of the [Data Artefact].
		DV-1.2	Establish and justify the integrity of the [Data Artefact].
		DV-1.3	Establish and justify the precision of the [Data Artefact].
		DV-1.4	Establish and justify the sufficiency of the [Data Artefact].
		DV-1.5	Establish and justify the accuracy of the [Data Artefact].
		DV-1.6	Establish and justify any assumptions in the [Data Artefact].
		DV-1.7	Establish and justify the confidence in the [Data Artefact].
	Process	DV-2.1	Establish and justify the type of the data transformation [Process].
		DV-2.2	Establish and justify the necessity of the data transformation [Process].
		DV-2.3	Establish and justify any assumptions in the use of the data transformation [Process].
		DV-2.4	Establish and justify confidence in the data transformation [Process].
	Acquisition	DV-3.1	Establish and justify the types of the data acquisition [Process].
		DV-3.2	Establish and justify the necessity of the data acquisition [Process].
		DV-3.3	Establish and justify any assumptions made during data acquisition [Process].

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
		DV-3.4	Establish and justify the resilience of the [Process] activities to operational change.
		DV-3.5	Establish and justify confidence in the data acquisition [Process].
	Management	DV-4.1	Establish and justify the integrity of the data management [Process].
		DV-4.2	Establish and justify the necessity of the data management [Process].
		DV-4.3	Establish and justify the resilience of management [Process] to operational change.
		DV-4.4	Establish and justify confidence in any 'physical' dependencies of the data management [Process].
		DV-4.5	Establish and justify confidence in the data management [Process].
Computation	Optimisation	CV-1.1	Establish and justify the necessity of optimisation [Computation].
		CV-1.2	Establish the dynamics of [Optimisation Algorithm].
		CV-1.3	Establish the accuracy of [Optimisation Algorithm].
		CV-1.4	Establish and justify the objective of the [Optimisation Algorithm].
		CV-1.5	Establish and justify any assumptions associated with the [Optimisation Algorithm].
		CV-1.6	Establish and justify confidence in [Optimisation Algorithm].
	Inference	CV-2.1	Establish and justify the necessity of [Inference Algorithm].

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
		CV-2.2	Establish and justify the dynamics of [Inference Algorithm].
		CV-2.3	Establish and justify any assumptions associated with the [Inference Algorithm].
		CV-2.4	Establish and justify confidence in [Inference Algorithm].
	Data Flow	CV-3.1	Establish and justify the necessity of data-flow/fusion technique.
		CV-3.2	Establish and justify all assumptions in data-flow/fusion technique.
		CV-3.3	Establish and justify confidence in data-flow/fusion technique.
Operational	Requirements	OV-1.1	Establish and justify the operational requirements of the [System].
		OV-1.2	Establish and justify the necessity of the operational requirements of the [System].
		OV-1.3	Establish and justify the contraints on the operational requirements of the [System].
		OV-1.4	Establish and justify confidence in the operational requirements of the [System].
	Scenario	OV-2.1	Establish and justify the necessity of the system for the [Scenario].
		OV-2.2	Establish and justify the manner of the utilisation of the system in the [Scenario].

Table A.2: RM-BNS Objectives (continued).

Viewpoint	View	Identifier	Objective
		OV-2.3	Establish and justify confidence in the functionality of the system in the [Scenario].
	Evolution	OV-3.1	Establish aspects of the system that will be exposed to evolution of the system or [Environment].
		OV-3.2	Establish and justify the necessity of the exposure of the system to evolving system or [Environment] aspects.
		OV-3.3	Establish and Justify any assumptions about the evolution of the system in the operational [Environment].
		OV-3.4	Establish and justify the [Process] for monitoring the validity of the system.
		OV-3.5	Establish and justify confidence in the [Process] for accomodating system evolution.
	Environment	OV-4.1	Establish and justify the operational [Environment] for the system.
		OV-4.2	Establish and justify the context and scope of the operational [Environment] for the system.
		OV-4.3	Establish and justify the necessity of interactions between the [Environment] and the system.

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
		OV-4.4	Establish and justify interactions between the [Environment] and the system.
		OV-4.5	Establish and justify confidence in the utilisation of the system in the target [Environment].
	Maintenance	OV-5.1	Establish and justify the [Process] used for the maintenance of the system.
		OV-5.2	Establish and justify the resilience of the maintenance [Process] to changes in the operational [Environment].
		OV-5.3	Establish and justify the standard of maintenance staff and facilities used to carry out the [Process].
		OV-5.4	Establish and justify the dependence of the system's performance on the maintenance [Process].
		OV-5.5	Establish and justify confidence in the [Process] for the maintenance of the system.
Implementation	Software	IV-1.1	Establish and justify the necessity of the software [Function].
		IV-1.2	Justify the implementation decisions for the software [Function].
		IV-1.3	Establish and justify confidence in the functional behaviour of the software [Function].

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
	Hardware	IV-2.1	Establish and justify the necessity of the hardware [Function].
		IV-2.2	Establish and justify the integrity of the hardware [Function].
		IV-2.3	Establish and justify confidence in the functional behaviour of the hardware [Function].
	Architecture	IV-3.1	Establish and justify the architecture of the system.
		IV-3.2	Establish and justify the integrity of the architecture of the system.
		IV-3.3	Establish and justify confidence in the architecture of the system.
	Integration	IV-4.1	Establish and justify the objective of the integration [Process].
		IV-4.2	Establish and justify the necessity of the integration [Process].
		IV-4.3	Establish and justify the utilisation of the integration [Process].
		IV-4.4	Establish and justify confidence in the integration [Process].
Technology	Infrastructure	IV-1.1	Establish and justify the necessity of the [Infrastructure].
		IV-1.2	Establish and justify the capabilities of the [Infrastructure].
		IV-1.3	Establish and justify the integrity of the [Infrastructure].
		IV-1.4	Establish and justify confidence in the [Infrastructure].
	Standards	IV-2.1	Establish and justify the standards [Framework] applicable to the system.

Table A.2: RM-BNS Objectives (continued).

Viewpoint	**View**	**Identifier**	**Objective**
		IV-2.2	Establish the limitations of the standards [Framework]with respect to the system.
		IV-2.3	Establish and justify the confidence in mitigating the limitations of the standards [Framework].
	Frameworks	IV-3.1	Establish and justify the selection of the chosen [Framework].
		IV-3.2	Establish and justify the necessity of the chosen [Framework].
		IV-3.3	Establish and justify the use-case of the [Framework].
		IV-3.4	Establish and justify confidence in the [Framework].
	Risk	IV-4.1	Establish and justify the risks of using the chosen [Framework].
		IV-4.2	Establish and justify the necessity of the risks associated with the chosen [Framework].
		IV-4.3	Establish and justify confidence in the acceptability of risks associated with the chosen [Framework].

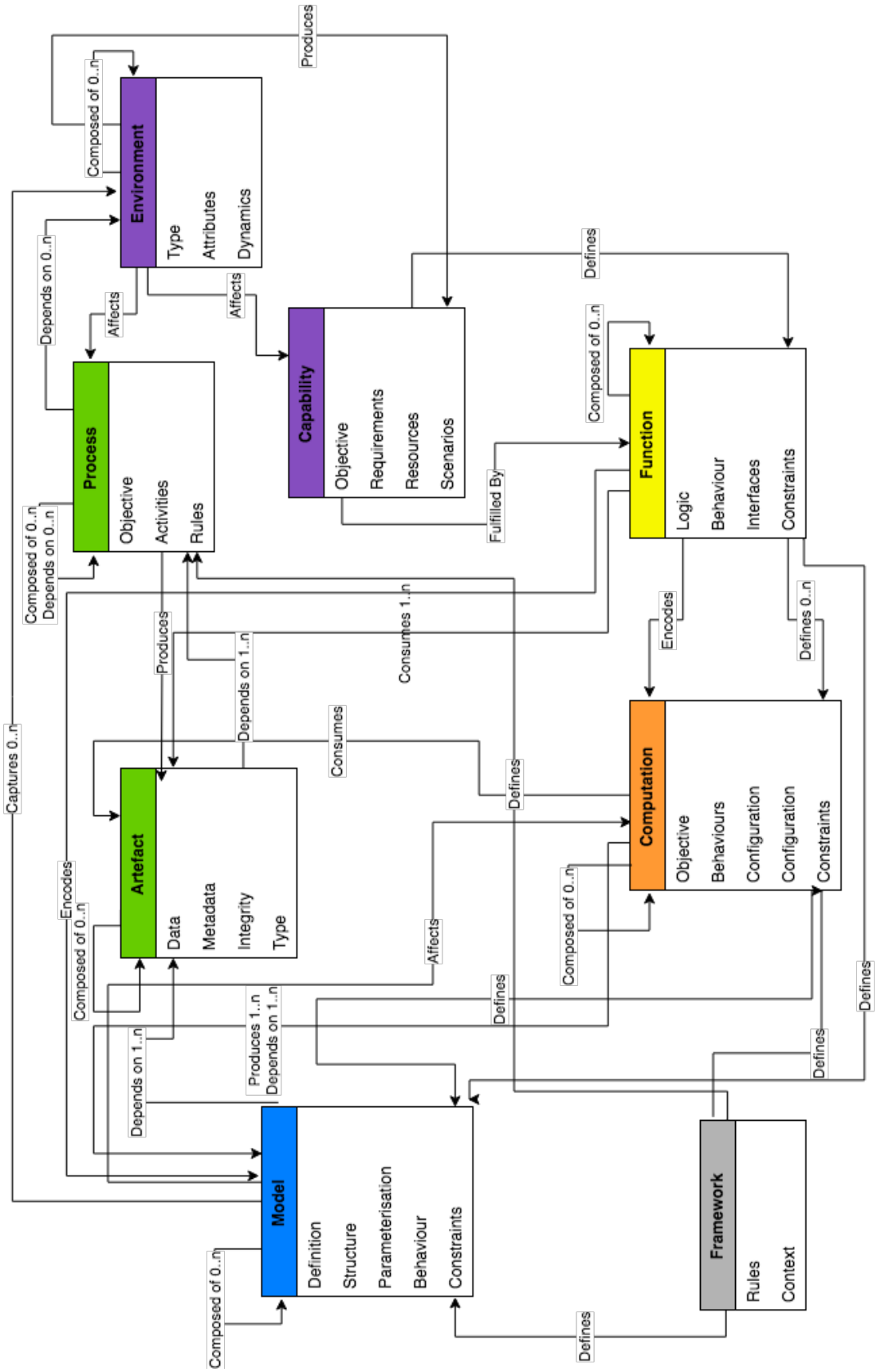


Figure A.1: The RM-BNS reference model.

Appendix B

Beinlich ALARM Model

This appendix contains the Bayesian Network model defined by Beinlich *et al* [136]. This model was used in the case study presented in Chapter 6. When used alongside the Apollo package (see Appendix C), this model can replicate the results of presented in that chapter.

B.1 Beinlich Model (HUGIN Format)

```
net
{
}
node HISTORY
{
  states = ( "TRUE" "FALSE" );
}
node CVP
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node PCWP
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node HYPOVOLEMIA
{
  states = ( "TRUE" "FALSE" );
}
```

```
node LVEDVOLUME
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node LVFAILURE
{
  states = ( "TRUE" "FALSE" );
}
node STROKEVOLUME
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node ERRLOWOUTPUT
{
  states = ( "TRUE" "FALSE" );
}
node HRBP
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node HREKG
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node ERRCAUTER
{
  states = ( "TRUE" "FALSE" );
}
node HRSAT
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node INSUFFANESTH
{
  states = ( "TRUE" "FALSE" );
}
node ANAPHYLAXIS
```

```
{
  states = ( "TRUE" "FALSE" );
}
node TPR
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node EXPCO2
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node KINKEDTUBE
{
  states = ( "TRUE" "FALSE" );
}
node MINVOL
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node FIO2
{
  states = ( "LOW" "NORMAL" );
}
node PVSAT
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node SAO2
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node PAP
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node PULMEMBOLUS
{
```

```
  states = ( "TRUE" "FALSE" );
}
node SHUNT
{
  states = ( "NORMAL" "HIGH" );
}
node INTUBATION
{
  states = ( "NORMAL" "ESOPHAGEAL" "ONESIDED" );
}
node PRESS
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node DISCONNECT
{
  states = ( "TRUE" "FALSE" );
}
node MINVOLSET
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node VENTMACH
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node VENTTUBE
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node VENTLUNG
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
}
node VENTALV
{
  states = ( "ZERO" "LOW" "NORMAL" "HIGH" );
```

```
}
node ARTCO2
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node CATECHOL
{
  states = ( "NORMAL" "HIGH" );
}
node HR
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node CO
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
node BP
{
  states = ( "LOW" "NORMAL" "HIGH" );
}
potential ( HISTORY | LVFAILURE )
{
  data = ((0.9 0.1)(0.01 0.99)) ;
}
potential ( CVP | LVEDVOLUME )
{
  data = ((0.95 0.04 0.01)(0.04 0.95 0.01)(0.01 0.29 0.70)) ;
}
potential ( PCWP | LVEDVOLUME )
{
  data = ((0.95 0.04 0.01)(0.04 0.95 0.01)(0.01 0.04 0.95)) ;
}
potential ( HYPOVOLEMIA )
{
  data = ( 0.2 0.8 );
}
```

```
potential ( LVEDVOLUME | HYPOVOLEMIA LVFAILURE )
{
  data = (((0.95 0.04 0.01)(0.01 0.09 0.90))((0.98 0.01 0.01)(0.05 0.90 0.05)
      )) ;
}
potential ( LVFAILURE )
{
  data = ( 0.05 0.95 );
}
potential ( STROKEVOLUME | HYPOVOLEMIA LVFAILURE )
{
  data = (((0.98 0.01 0.01)(0.50 0.49 0.01))((0.95 0.04 0.01)(0.05 0.90 0.05)
      )) ;
}
potential ( ERRLOWOUTPUT )
{
  data = ( 0.05 0.95 );
}
potential ( HRBP | ERRLOWOUTPUT HR )
{
  data = (((0.98 0.01 0.01)(0.3 0.4 0.3)(0.01 0.98 0.01))((0.40 0.59 0.01)
      (0.98 0.01 0.01)(0.01 0.01 0.98))) ;
}
potential ( HREKG | ERRCAUTER HR )
{
  data = (((0.3333333 0.3333333 0.3333333)(0.3333333 0.3333333 0.3333333)
      (0.01 0.98 0.01))((0.3333333 0.3333333 0.3333333)(0.98 0.01 0.01)(0.01
      0.01 0.98))) ;
}
potential ( ERRCAUTER )
{
  data = ( 0.1 0.9 );
}
potential ( HRSAT | ERRCAUTER HR )
{
  data = (((0.3333333 0.3333333 0.3333333)(0.3333333 0.3333333 0.3333333)
      (0.01 0.98 0.01))((0.3333333 0.3333333 0.3333333)(0.98 0.01 0.01)(0.01
```

```
      0.01 0.98))) ;
}
potential ( INSUFFANESTH )
{
  data = ( 0.1 0.9 );
}
potential ( ANAPHYLAXIS )
{
  data = ( 0.01 0.99 );
}
potential ( TPR | ANAPHYLAXIS )
{
  data = ((0.98 0.01 0.01)(0.3 0.4 0.3)) ;
}
potential ( EXPCO2 | ARTCO2 VENTLUNG )
{
  data = (((0.97 0.01 0.01 0.01)(0.01 0.97 0.01 0.01)(0.01 0.01 0.97 0.01)
      (0.01 0.01 0.01 0.97))((0.01 0.97 0.01 0.01)(0.97 0.01 0.01 0.01)(0.01
      0.01 0.97 0.01)(0.01 0.01 0.01 0.97))((0.01 0.97 0.01 0.01)(0.01 0.01
      0.97 0.01)(0.97 0.01 0.01 0.01)(0.01 0.01 0.01 0.97))) ;
}
potential ( KINKEDTUBE )
{
  data = ( 0.04 0.96 );
}
potential ( MINVOL | INTUBATION VENTLUNG )
{
  data = (((0.97 0.01 0.01 0.01)(0.01 0.01 0.01 0.97)(0.50 0.48 0.01 0.01)
      (0.01 0.97 0.01 0.01))((0.01 0.97 0.01 0.01)(0.97 0.01 0.01 0.01)(0.50
      0.48 0.01 0.01)(0.01 0.01 0.97 0.01))((0.01 0.01 0.97 0.01)(0.60 0.38
      0.01 0.01)(0.97 0.01 0.01 0.01)(0.01 0.01 0.01 0.97))) ;
}
potential ( FIO2 )
{
  data = ( 0.05 0.95 );
}
potential ( PVSAT | FIO2 VENTALV )
```

```
{
  data = (((1.0 0.0 0.0)(0.95 0.04 0.01)(1.0 0.0 0.0)(0.01 0.95 0.04))((0.99
      0.01 0.00)(0.95 0.04 0.01)(0.95 0.04 0.01)(0.01 0.01 0.98))) ;
}
potential ( SAO2 | PVSAT SHUNT )
{
  data = (((0.98 0.01 0.01)(0.98 0.01 0.01))((0.01 0.98 0.01)(0.98 0.01 0.01)
      )((0.01 0.01 0.98)(0.69 0.30 0.01))) ;
}
potential ( PAP | PULMEMBOLUS )
{
  data = ((0.01 0.19 0.80)(0.05 0.90 0.05)) ;
}
potential ( PULMEMBOLUS )
{
  data = ( 0.01 0.99 );
}
potential ( SHUNT | INTUBATION PULMEMBOLUS )
{
  data = (((0.1 0.9)(0.95 0.05))((0.1 0.9)(0.95 0.05))((0.01 0.99)(0.05 0.95)
      )) ;
}
potential ( INTUBATION )
{
  data = ( 0.92 0.03 0.05 );
}
potential ( PRESS | INTUBATION KINKEDTUBE VENTTUBE )
{
  data = ((((0.97 0.01 0.01 0.01)(0.05 0.25 0.25 0.45)(0.97 0.01 0.01 0.01)
      (0.20 0.75 0.04 0.01))((0.01 0.01 0.01 0.97)(0.01 0.29 0.30 0.40)(0.01
      0.01 0.01 0.97)(0.01 0.90 0.08 0.01)))(((0.01 0.30 0.49 0.20)(0.01 0.15
      0.25 0.59)(0.01 0.97 0.01 0.01)(0.20 0.70 0.09 0.01))((0.97 0.01 0.01
      0.01)(0.01 0.01 0.08 0.90)(0.97 0.01 0.01 0.01)(0.01 0.01 0.38 0.60)))
      (((0.01 0.01 0.08 0.90)(0.97 0.01 0.01 0.01)(0.01 0.01 0.97 0.01)(0.97
      0.01 0.01 0.01))((0.10 0.84 0.05 0.01)(0.01 0.01 0.01 0.97)(0.40 0.58
      0.01 0.01)(0.01 0.01 0.01 0.97)))) ;
}
```

```
potential ( DISCONNECT )
{
  data = ( 0.1 0.9 );
}
potential ( MINVOLSET )
{
  data = ( 0.05 0.90 0.05 );
}
potential ( VENTMACH | MINVOLSET )
{
  data = ((0.05 0.93 0.01 0.01)(0.05 0.01 0.93 0.01)(0.05 0.01 0.01 0.93)) ;
}
potential ( VENTTUBE | DISCONNECT VENTMACH )
{
  data = (((0.97 0.01 0.01 0.01)(0.97 0.01 0.01 0.01)(0.97 0.01 0.01 0.01)
      (0.01 0.01 0.97 0.01))((0.97 0.01 0.01 0.01)(0.97 0.01 0.01 0.01)(0.01
      0.97 0.01 0.01)(0.01 0.01 0.01 0.97))) ;
}
potential ( VENTLUNG | INTUBATION KINKEDTUBE VENTTUBE )
{
  data = ((((0.97 0.01 0.01 0.01)(0.97 0.01 0.01 0.01)(0.97 0.01 0.01 0.01)
      (0.97 0.01 0.01 0.01))((0.30 0.68 0.01 0.01)(0.95 0.03 0.01 0.01)(0.01
      0.01 0.01 0.97)(0.01 0.97 0.01 0.01)))(((0.95 0.03 0.01 0.01)(0.97 0.01
      0.01 0.01)(0.01 0.97 0.01 0.01)(0.97 0.01 0.01 0.01))((0.97 0.01 0.01
      0.01)(0.50 0.48 0.01 0.01)(0.97 0.01 0.01 0.01)(0.01 0.01 0.97 0.01)))
      (((0.40 0.58 0.01 0.01)(0.97 0.01 0.01 0.01)(0.01 0.01 0.97 0.01)(0.97
      0.01 0.01 0.01))((0.97 0.01 0.01 0.01)(0.30 0.68 0.01 0.01)(0.97 0.01
      0.01 0.01)(0.01 0.01 0.01 0.97)))) ;
}
potential ( VENTALV | INTUBATION VENTLUNG )
{
  data = (((0.97 0.01 0.01 0.01)(0.01 0.01 0.01 0.97)(0.01 0.01 0.97 0.01)
      (0.03 0.95 0.01 0.01))((0.01 0.97 0.01 0.01)(0.97 0.01 0.01 0.01)(0.01
      0.01 0.01 0.97)(0.01 0.94 0.04 0.01))((0.01 0.01 0.97 0.01)(0.01 0.97
      0.01 0.01)(0.97 0.01 0.01 0.01)(0.01 0.88 0.10 0.01))) ;
}
potential ( ARTCO2 | VENTALV )
```

```
{
  data = ((0.01 0.01 0.98)(0.01 0.01 0.98)(0.04 0.92 0.04)(0.90 0.09 0.01)) ;
}
potential ( CATECHOL | ARTCO2 INSUFFANESTH SAO2 TPR )
{
  data = (((((0.01 0.99)(0.01 0.99)(0.7 0.3))((0.01 0.99)(0.05 0.95)(0.7 0.3)
      )((0.01 0.99)(0.05 0.95)(0.95 0.05)))(((0.01 0.99)(0.05 0.95)(0.7 0.3))
      ((0.01 0.99)(0.05 0.95)(0.95 0.05))((0.05 0.95)(0.05 0.95)(0.95 0.05))))
      ((((0.01 0.99)(0.01 0.99)(0.7 0.3))((0.01 0.99)(0.05 0.95)(0.7 0.3))
      ((0.01 0.99)(0.05 0.95)(0.99 0.01)))(((0.01 0.99)(0.05 0.95)(0.7 0.3))
      ((0.01 0.99)(0.05 0.95)(0.99 0.01))((0.05 0.95)(0.05 0.95)(0.99 0.01))))
      ((((0.01 0.99)(0.01 0.99)(0.1 0.9))((0.01 0.99)(0.01 0.99)(0.1 0.9))
      ((0.01 0.99)(0.01 0.99)(0.3 0.7)))(((0.01 0.99)(0.01 0.99)(0.1 0.9))
      ((0.01 0.99)(0.01 0.99)(0.3 0.7))((0.01 0.99)(0.01 0.99)(0.3 0.7))))) ;
}
potential ( HR | CATECHOL )
{
  data = ((0.05 0.90 0.05)(0.01 0.09 0.90)) ;
}
potential ( CO | HR STROKEVOLUME )
{
  data = (((0.98 0.01 0.01)(0.95 0.04 0.01)(0.30 0.69 0.01))((0.95 0.04 0.01)
      (0.04 0.95 0.01)(0.01 0.30 0.69))((0.80 0.19 0.01)(0.01 0.04 0.95)(0.01
      0.01 0.98))) ;
}
potential ( BP | CO TPR )
{
  data = (((0.98 0.01 0.01)(0.98 0.01 0.01)(0.3 0.6 0.1))((0.98 0.01 0.01)
      (0.10 0.85 0.05)(0.05 0.40 0.55))((0.90 0.09 0.01)(0.05 0.20 0.75)(0.01
      0.09 0.90))) ;
}
```

Appendix C

Bayesian Network Tool

C.1 Implementation Notes

A copy of the 'Apollo' Bayesian Network software package has been provided alongside this thesis. This package was developed as both a learning exercise and as a means of exploring the ideas presented in this thesis is in a more technical setting. The development of this package and application of it to problems such as that outlined in Chapter 6 provided a number of insights into the technical challenges associated with the end-to-end development of a BNS.

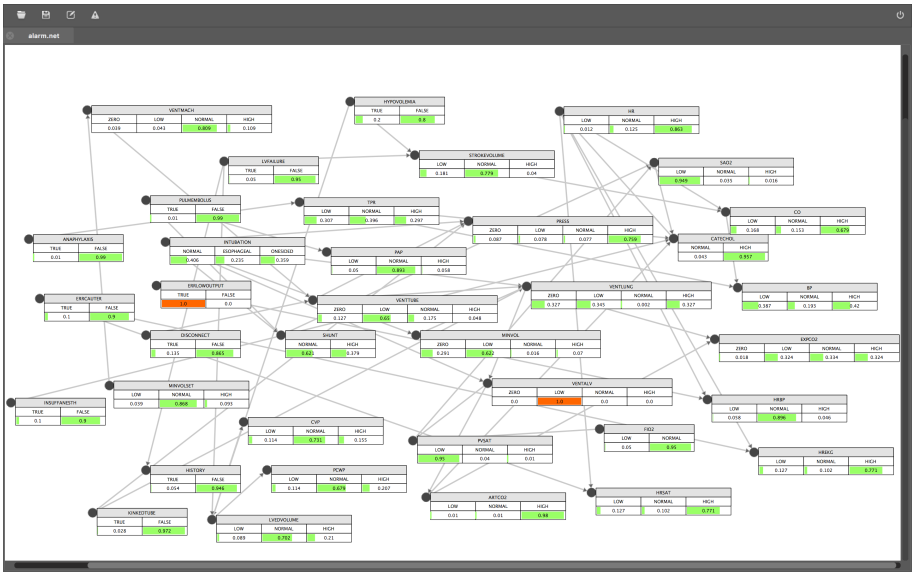


Figure C.1: A screenshot of the Apollo Dashboard app.

The package was written in Python. Experimental parts of the package were written in Cython and C. However, for portability, stability and ease-of-use, the copy supplied

contains only components written in pure Python. It is capable of performing exact and approximate inference over several types of probabilistic graphical models. This includes Markov Networks (and by extension Markov Chains), Hidden Markov Models, Conditional Linear Gaussian Models and of course Bayesian Networks. A user interface ('Apollo Dashboard') was also developed for the Apollo package. This is shown in Figure C.1. The interface was built on top of the Qt User Interface (UI) framework. This has not been included in the provided package due to the terms of the Qt licensing agreement.

Included in the Apollo package is an implementation of the MCA algorithm outlined in Chapter 4. Instructions for the installation of the package can be found in the README.md file (this file is also available in plain text format within the package). Directions for the use of the package are also provided in this file. The accuracy of the inference engine implemented in the package was verified against two commercially available inference engines: HUGIN Lite and SAMIAM. The primary limitation of the package with respect to commercial solutions is speed – it is significantly slower than both HUGIN and SAMIAM. Updates aimed at improving the performance of the package were abandoned due to time constraints, though these could be completed if required.

As discussed in Chapters 4 and 6, the MCA implementation is a simple approach to a complex problem. Therefore, the 'ModelCriticalityAnalysis' has been designed to work with any object that conforms to the interface defined by the 'BaseSensitivityAnalysis' object. This is intended to support the modification of the MCA technique in the event the package were to be used for some of the 'Future Work' avenues suggested in Chapter 7. Finally, the package was developed in Python 3 and utilises a minimal set of dependencies – the only dependency outside of Python 3's standard library is 'NumPy' (Version 1.14.0 or higher).

Abbreviations

BN	Bayesian Network
BNS	Bayesian Network-based System
AI	Artificial Intelligence
AIS	Artificial Intelligence-based System
MCA	Model Criticality Analysis
RM-BNS	Reference Model for Bayesian Network-based Systems
ANN	Artificial Neural Network
SVM	Support Vector Machine
FTA	Fault Tree Analysis
FMEA	Failure Modes and Effects Analysis
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Organization for Standardization
UAV	Unmanned Aerial Vehicle
UAS	Unmanned Aerial System

References

[1] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. Springer Science & Business Media, 2nd ed., 2009.

[2] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT Press, 1st ed., 2009.

[3] F. V. Jensen, *An Introduction to Bayesian Networks*. UCL Press, 1st ed., 1996.

[4] D. Koller and A. Pfeffer, "Object-Oriented Bayesian Networks," pp. 302–313, 1997.

[5] K. P. Baclawski, "Bayesian network development," *New Trends in Software Methodologies, Tools, and Techniques*, pp. 18–48, 2004.

[6] K. Przytula and D. Thompson, "Construction of Bayesian networks for diagnostics," *Proceedings IEEE Aerospace Conference. (Cat. No.00TH8484)*, vol. 5, pp. 193–200, 2000.

[7] V. O. Ogunsanya, *Decision Support using Bayesian Networks for Clinical Decision Making*. Phd thesis, University of London, 2012.

[8] E. Denney, B. Fischer, J. Schumann, and J. Richardson, "Automatic Certification of Kalman Filters for Reliable Code Generation," no. 1207, 2005.

[9] K. P. Murphy, "An introduction to graphical models," no. May, pp. 1–19, 2001.

[10] Y. Chen, G. Liang, and K. K. Lee, "Abnormal Behavior Detection by Multi-SVM-Based Bayesian Network," in *2007 International Conference on Information Acquisition*, 2007.

[11] J. L. Lustgarten, J. B. Balasubramanian, and S. Visweswaran, "Learning Parsimonious Classification Rules from Gene Expression Data Using Bayesian Networks," *Data*, vol. 2.1, 2017.

[12] J. Dean, "Distilling the Knowledge in a Neural Network," pp. 1–9, 2015.

[13] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," pp. 1–27, 2012.

[14] K. Fowler, "Mission-critical and safety-critical development," *IEEE Instrumentation Measurement*, vol. 7, no. 4, pp. 52–59, 2004.

[15] A. Karpathy, "Software 2.0," *Medium*, 2017.

[16] J. Pearl, "Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning," 1985.

[17] J. Pearl, "Fusion, Propagation, and Structuring in Belief Networks," *Artificial Intelligence*, vol. 29, pp. 241–288, 1986.

[18] J. Pearl and S. Russell, "Bayesian Networks," 2000.

[19] M. G. Kendall, *The Advanced Theory of Statistics*. Charles Griffin London, 2nd ed., 1946.

[20] R. Barlow and J. Wiley, *Statistics: A Guide to the Use of Statistical Methods for the Physical Sciences*. 2008.

[21] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete Problems in AI Safety," pp. 1–29, 2016.

[22] J. Pearl, "Causal inference in statistics: An Overview," *Statistics Surveys*, vol. 3, pp. 96–146, 2009.

[23] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, pp. 452–459, 2015.

[24] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," vol. 82, no. Series D, pp. 35–45, 1960.

[25] M. Correa, C. Bielza, and J. Pamies-Teixeira, "Comparison of Bayesian networks and artificial neural networks for quality detection in a machining process," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7270–7279, 2009.

[26] D. J. Spiegelhalter, A. P. Dawid, S. L. Lauritzen, G. Robert, D. J. Spiegelhalter, A. P. Dawid, S. L. Lauritzen, and R. G. Cowell, "Bayesian Analysis in Expert Systems," *Statistical Science*, vol. 8, no. 3, pp. 219–247, 1993.

[27] R. S. Sutton, A. G. Barto, R. S. Sutton, A. G. Barto, and S. Richard, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[28] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[29] J. Thornton, T. Gustafsson, M. Blumenstein, and T. Hine, "Robust Character Recognition using a Hierarchical Bayesian Network," 2006.

[30] Q. Ji, "Interactive Image Segmentation," no. October 2011, 2014.

[31] C. Li, "Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[32] S. L. Lauritzen, *Graphical Models*. Clarendon Press, 1996.

[33] W. N. Mohamed, I. Diamond, and P. W. F. Smith, "The determinants of infant mortality in Malaysia: a graphical chain modelling approach," *Journal of the Royal Statistical Society*, vol. 161, no. 3, pp. 349–366, 1998.

[34] S. L. Lauritzen and T. S. Richardson, "Chain graph models and their causal interpretations," 2002.

[35] M. Velikova, J. Terwisscha van Scheltinga, P. J. Lucas, and M. Spaanderman, "Exploiting causal functional relationships in Bayesian network modelling for personalised healthcare," *Internation Journal of Approximate Reasoning*, vol. 55, no. 12, pp. 59–73, 2014.

[36] B. Cai, H. Liu, and M. Xie, "A real-time fault diagnosis methodology of complex systems using object-oriented Bayesian networks," *Mechanical Systems and Signal Processing*, vol. 80, no. August 2016, pp. 31–44, 2011.

[37] V. Pavlovi, J. M. Rehg, T.-j. Cham, and K. P. Murphy, "A Dynamic Bayesian Network Approach to Figure Tracking Using Learned Dynamic Models," in *International Conference on Computer Vision*, no. Iccv 99, pp. 94–101, 1999.

[38] S. Thrun, "Bayesian Landmark Learning for Mobile Robot Localization," *Machine Learning*, 1997.

[39] H. Zhou and S. Sakane, "Sensor Planning for Mobile Robot Localization Using Bayesian Network Inference," *Advanced Robotics*, 2002.

[40] P. Aloy, A. Pujol, R. Mosca, and J. Farre, "Unveiling the role of network and systems biology in drug discovery," no. February, 2010.

[41] D. D. Lewis, "Naive (Bayes) at Forty : The Independence Assumption in Information Retrieval," in *European conference on machine learning*, 1998.

[42] V. J. Easton and J. H. McColl, "Statistics Glossary," 2002.

[43] S. Park and J. K. Aggarwal, "A hierarchical Bayesian network for event recognition of human actions and interactions," *Multimedia systems*, vol. 10, pp. 164–179, 2004.

[44] N. Friedman and M. Goldszmidt, "Learning Bayesian networks with local structure," *Learning in graphical models*, pp. 421–459, 1998.

[45] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," *Journal of the Royal Statistical Society*, vol. 50, no. 2, pp. 157–224, 1988.

[46] Norsys, "Norsys Net Library," 2018.

[47] E. Cambria and B. White, "Jumping NLP curves: A review of natural language processing research," *IEEE Computational Intelligence Magazine*, vol. 9, no. 2, pp. 48–57, 2014.

[48] S. Park and J. K. Aggarwal, "Recognition of two-person interactions using a hierarchical Bayesian network," *First ACM SIGMM international workshop on Video surveillance - IWVS '03*, p. 65, 2003.

[49] C. Premebida, D. R. Faria, and U. Nunes, "Dynamic Bayesian network for semantic place classification in mobile robotics," *Autonomous Robots*, vol. 41, no. 5, pp. 1161–1172, 2017.

[50] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational Inference: A Review for Statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[51] M. I. Jordan, "Graphical models and approximate posterior inference," in *Learning in Graphical Models*, vol. 1, pp. 4–17, 2012.

[52] S. H. Chen and C. A. Pollino, "Good practice in Bayesian network modelling," *Environmental Modelling and Software*, vol. 37, pp. 134–145, 2012.

[53] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," 2015.

[54] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. 2014.

[55] J. Bergstra JAMESBERGSTRA and U. Yoshua Bengio YOSHUABENGIO, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[56] A. C. Constantinou, N. Fenton, W. Marsh, L. Radlinski, and A. Constantinou, "From complex questionnaire and interviewing data to intelligent Bayesian Network models for medical decision support Europe PMC Funders Group," *Artif Intell Med*, vol. 67, pp. 75–93, 2016.

[57] a. J. Butte and I. S. Kohane, "Unsupervised knowledge discovery in medical databases using relevance networks.," *Proceedings / AMIA ... Annual Symposium. AMIA Symposium*, no. February, pp. 711–715, 1999.

[58] O. Woodberry, A. E. Nicholson, K. B. Korb, and C. Pollino, "Parameterising Bayesian Networks," *Artificial Intelligence*, pp. 1101–1107, 2004.

[59] C. A. Pollino, O. Woodberry, A. Nicholson, K. Korb, and B. T. Hart, "Parameterisation and evaluation of a Bayesian network for use in an ecological risk assessment," *Environmental Modelling and Software*, vol. 22, no. 8, pp. 1140–1152, 2007.

[60] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data," *Machine Learning*, vol. 20, no. 3, pp. 197–243, 1995.

[61] J. Pitchforth and K. Mengersen, "A proposed validation framework for expert elicited Bayesian Networks," *Expert Systems with Applications*, vol. 40, no. 1, pp. 162–167, 2013.

[62] A. Rashwan, H. Zhao, and P. Poupart, "Online and Distributed Bayesian Moment Matching for Parameter Learning in Sum-Product Networks," *Proceedings of*

the 19th International Conference on Artificial Intelligence and Statistics, vol. 51, pp. 1469–1477, 2016.

[63] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, "Artificial intelligence: a modern approach," 2003.

[64] B. J. Taylor, *Methods and Procedures for the Verification and Validation of Artificial Neural Networks*. 2006.

[65] B. G. Marcot, "Metrics for evaluating performance and uncertainty of Bayesian network models," *Ecological Modelling*, vol. 230, pp. 50–62, 2012.

[66] K. W. Przytula, D. Allen, J. Vian, and G. Mansouri, "Health Monitoring For Commercial Aircraft Systems," in *International Congress of the Aeronautical Sciences*, 2008.

[67] SAE, "ARP-4754: Certification considerations for highly-integrated or complex aircraft systems," 2010.

[68] SAE, "ARP-4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," 1996.

[69] RTCA, "DO-178C: Software Considerations in Airborne Systems and Equipment Certification," 2012.

[70] R. A. Weaver, *The Safety of Software – Constructing and Assuring Arguments*. PhD thesis, 2003.

[71] R. Alexander, M. Hall-may, and T. Kelly, "Certification of Autonomous Systems Robert Alexander, Martin Hall-May, Tim Kelly Department of Computer Science, University of York Heslington, York, YO10 5DD," *2nd SEAS DTC Technical Conference*, 2007.

[72] S. A. Jacklin, M. R. Lowry, J. M. Schumann, P. P. Gupta, J. T. Bosworth, E. Zavala, J. W. Kelly, K. J. Hayhurst, C. M. Belcastro, and C. M. Belcastro, "Verification, Validation, and Certification Challenges for Adaptive Flight-Critical Control System Software," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 1–10, 2004.

[73] J. Hatcliff, A. Wassyng, T. Kelly, C. Comar, and P. Jones, "Certifiably safe software-dependent systems: challenges and directions," *Proceedings of the on Future of Software Engineering - FOSE 2014*, pp. 182–200, 2014.

[74] T. Kelly, "Software Certification: Where is Confidence Won and Lost?," in *Safety Critical Systems Symposium 2014*, pp. 1–13, 2014.

[75] J. A. Mcdermid, "Software Safety : Where' s the Evidence ?," *7th International Symposium on Formal Techniques in RealTime and FaultTolerant Systems*, vol. 3, pp. 23–36, 2002.

[76] R. A. Weaver, J. A. McDermid, and T. P. Kelly, "Software Safety Arguments: Towards a Systematic Categorisation of Evidence," in *International System Safety Conference*, 2002.

[77] I. Habli, R. Hawkins, and T. Kelly, "Software safety: Relating software assurance and software integrity," *International Journal of Critical Computer-Based Systems*, vol. 1, no. 4, pp. 364–383, 2010.

[78] M. Squair, "Issues in the application of software safety standards," *... workshop on Safety critical systems and software- ...*, vol. XXX, 2006.

[79] J. McDermid and D. Pumfrey, "Software Safety: Why is there no Consensus?," *Proceedings of the 19th International System Safety Conference*, 2001.

[80] J. Rushby, "New challenges in certification for aircraft software ppt," *2011 Proceedings of the Ninth ACM International Conference on Embedded Software (EMSOFT)*, no. October, pp. 211–218, 2011.

[81] R. Alexander, T. Kelly, and B. Gorry, "Safety Lifecycle Activities for Autonomous Systems Development," *4th SEAS DTC Technical Conference*, 2010.

[82] M. Asplund, P. Gustafsson, T. Nordmark, M. Rantatalo, M. Palo, S. M. Famurewa, and K. Wandt, "Reliability and measurement accuracy of a condition monitoring system in an extreme climate: A case study of automatic laser scanning of wheel profiles," *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 228, no. 6, pp. 695–704, 2014.

[83] D. J. Pumfrey, *The Principled Design of Computer System Safety Analyses*. PhD thesis, 1999.

[84] Z. Kurd, "Developing artificial neural networks for safety critical systems," pp. 11–19, 2007.

[85] C. Culbert, G. Riley, and R. T. Savely, "Approaches to the Verification of Rule-based Expert Systems," 1987.

[86] J. Rushby, "Quality Measures and Assurance for AI Software," Tech. Rep. October 1988, 1988.

[87] M. Holloway, "Making the Implicit Explicit: Towards An Assurance Case for DO-178C," 2013.

[88] B. Brosgol, "Safety and security: Certification issues and technologies," *Crosstalk, The Journal of Defense Software Engineering*, pp. 9–14, 2008.

[89] C. M. Holloway, "Towards understanding the DO-178C / ED-12C assurance case," *System Safety, incorporating the Cyber Security Conference 2012, 7th IET International Conference on*, no. December, pp. 1–6, 2012.

[90] A. Dupuy and N. Leveson, "An empirical evaluation of the MC/DC coverage criterion on the HETE-2 satellite software," *19th DASC. 19th Digital Avionics Systems Conference. Proceedings (Cat. No.00CH37126)*, vol. 1, pp. 1B6/1–1B6/7, 2000.

[91] M. Thomas, "Engineering Judgement," *Proceedings of the 9th Australian Workshop on Safety Related Programmable Systems (SCS'04)*, pp. 43–47, 2004.

[92] P. V. Bhansali, "The MCDC paradox," *ACM SIGSOFT Software Engineering Notes*, vol. 32, no. 3, p. 1, 2007.

[93] A. Rajan, M. W. Whalen, and M. P. E. Heimdahl, "The Effect of Program and Model Structure on MC / DC Test Adequacy Coverage," *Proc. ICSE*, no. January, pp. 161–170, 2008.

[94] IEC, "IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems," 2010.

[95] ISO, "ISO-26262: Road vehicles – Functional safety," 2011.

[96] R. Alexander, H. Hawkins, and D. Rae, "Situation coverage – a coverage criterion for testing autonomous robots," pp. 1–20, 2015.

[97] Ministry of Defence, "Defence Standard 00-056," 2017.

[98] J. A. Asensio, J. M. Barton, L. A. Wonsetler, and N. R. Thomford, "A Systematic Approach to Safety Case Management," tech. rep., 2004.

[99] Ministry of Defence, "Defence Standard 00-055," 2016.

[100] C. Menon, R. D. Hawkins, and J. McDermid, *Defence Standard 00-56 Issue 4: Towards Evidence-Based Safety Standards*. 2009.

[101] I. E. C. (IEC), "International Electrotechnical Vocabulary (IEV) 191-05-24,"

[102] Y. Papadopoulos, D. Parker, and C. Grante, "Automating the failure modes and effects analysis of safety critical systems," *High Assurance Systems Engineering, 2004. Proceedings. Eighth IEEE International Symposium on*, pp. 310–311, 2004.

[103] M. T. Oldenhof, J. F. van Leeuwen, M. J. Nauta, D. de Kaste, Y. M. C. F. Odekerken-Rombouts, M. J. Vredenbregt, M. Weda, and D. M. Barends, "Consistency of FMEA used in the validation of analytical procedures," *Journal of Pharmaceutical and Biomedical Analysis*, vol. 54, no. 3, pp. 592–595, 2011.

[104] C. Spreafico, D. Russo, and C. Rizzi, "A state-of-the-art review of FMEA/FMECA including patents," *Computer Science Review*, vol. 25, pp. 19–28, 2017.

[105] F. Redmill, "Risk analysis - a subjective process," *Engineering Management Journal*, vol. 12, no. 2, p. 91, 2002.

[106] C. a. Ericson, "Fault Tree Analysis – A History," *The 17th International System Safety Conference*, pp. 1–9, 1999.

[107] W. Larsen, "Fault Tree Analysis," 1974.

[108] S. Kabir, "An overview of fault tree analysis and its application in model based dependability analysis," *Expert Systems with Applications*, vol. 77, pp. 114–135, 2017.

[109] R. E. Barlow, "Introduction to Fault Tree Analysis," no. December, 1972.

[110] A. Bobbio, L. Portinale, M. Minichino, and E. Ciancamerla, "Improving the analysis of dependable systems by mapping Fault Trees into Bayesian Networks," *Reliability Engineering and System Safety*, vol. 71, no. 3, pp. 249–260, 2001.

[111] N. Khakzad, F. Khan, and P. Amyotte, "Safety analysis in process facilities: Comparison of fault tree and Bayesian network approaches," *Reliability Engineering and System Safety*, vol. 96, no. 8, pp. 925–932, 2011.

[112] B. Fischhoff, P. Slovic, and S. Lichtenstein, "Fault Trees: Sensitivity of Estimated Failure Probabilities to Problem Representation," *Perception*, vol. 4, no. 2, pp. 330–344, 1978.

[113] K. Pei, Y. Cao, J. Yang, and S. Jana, "DeepXplore : Automated Whitebox Testing of Deep Learning Systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*, 2017.

[114] R. V. Yampolskiy, "Artificial intelligence safety engineering: Why machine ethics is a wrong approach," *Studies in Applied Philosophy, Epistemology and Rational Ethics*, vol. 5, pp. 389–396, 2013.

[115] B. G. Buchanan, "A (Very) Brief History of Artificial Intelligence," *AI Magazine*, vol. 26, no. 4, pp. 53–60, 2006.

[116] M. Z. Bell, "Why Expert Systems Fail," vol. 36, no. 7, pp. 613–619, 1985.

[117] R. M. O'Keefe and D. E. O'Leary, "Expert system verification and validation: a survey and tutorial," *Artificial Intelligence Review*, vol. 7, no. 1, pp. 3–42, 1993.

[118] M. Mehrotra, "Rule Groupings : An Approach towards Verification of Expert Systems," pp. 2–5, 1991.

[119] H. L. Dreyfus and S. E. Dreyfus, "What Artificial Experts Can and Cannot Do," *AI & Society*, no. January 1990, pp. 18–26, 1992.

[120] B. Boehm, "A Spiral model of software development and enhancement," 1988.

[121] J. Rushby, "Validation and Testing of Knowledge-Based Systems How bad can it get?," *AAAI-88 Workshop on Validation and Testing of Knowledge-Based Systems*, pp. 1–8, 1988.

[122] J. Schumann, P. Gupta, and S. Nelson, "On Verification & Validation of Neural Network Based Controllers," in *Proc. of International Conf on Engineering Applications of Neural Networks*, 2003.

[123] IEC, "12207: Systems and software engineering – Software life cycle processes," 2017.

[124] J. F. Kolen and J. B. Pollack, "Back Propagation is Sensitive to Initial Conditions Back Propagation is Sensitive to Initial Conditions," *Advances in Neural Information Processing Systems(3)*, no. 1988, pp. 860–867, 1991.

[125] J. Y. F. Yam and T. W. S. Chow, "A weight initialization method for improving training speed in feedforward neural network," *Neurocomputing*, vol. 30, no. 1-4, pp. 219–232, 2000.

[126] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances In Neural Information Processing Systems*, pp. 1–9, 2012.

[127] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (Almost) from Scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[128] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," pp. 1–9, 2014.

[129] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," *arXiv preprint arXiv:1312.5602*, pp. 1–9, 2013.

[130] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[131] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Ł. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and

J. Dean, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," pp. 1–23, 2016.

[132] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[133] J. Yosinski, J. Clune, A. Nguyen, J. Yosinski, and J. Clune, "Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images," 2014.

[134] J. Leike, M. Martic, V. Krakovna, P. A. Ortega, T. Everitt, A. Lefrancq, L. Orseau, and S. Legg, "AI Safety Gridworlds," 2017.

[135] W. Patrick, "MIT 6.034 Artificial Intelligence," 2010.

[136] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper, "The ALARM Monitoring System: A Case Study with two Probabilistic Inference Techniques for Belief Networks," in *Artificial Intelligence in Medicine*, pp. 247–256, 1989.

[137] Y. Yang and G. I. Webb, "Discretization for naive-Bayes learning: Managing discretization bias and variance," *Machine Learning*, vol. 74, no. 1, pp. 39–74, 2009.

[138] Z. Ghahramani, "Learning Dynamic Bayesian Networks," *Adaptive processing of sequences and data structures.*, vol. 1387, no. 11, pp. 168–197, 1997.

[139] L. A. Davis, "First Stage recovery," pp. 152–153, 2016.

[140] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel, "Object-Oriented Bayesian Networks for Detection of Lane Change Maneuvers," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 3, pp. 19–31, 2012.

[141] M. Neil, N. Fenton, and L. Nielsen, "Building large-scale Bayesian networks," *Knowledge Engineering Review*, vol. 15, no. 3, pp. 257–284, 2000.

[142] F. Ferri, A. Ahondan, G. Colombatti, C. Bettanini, S. Bebei, O. Karatekin, B. Van Hove, N. Gerbal, S. Asmar, S. Lewis, and F. Forget, "Atmospheric mars entry and landing investigations & analysis (AMELIA) by ExoMars 2016 Schiaparelli Entry Descent module: The ExoMars entry, descent and landing science," *4th IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2017 - Proceedings*, pp. 262–265, 2017.

[143] D. Heckerman and J. S. Breese, "Causal independence for probability assessment and inference using Bayesian networks," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, vol. 26, no. 6, pp. 826–831, 1993.

[144] Data Safety Inititiative Working Group, *Data Safety Guidance.* third ed., 2016.

[145] Y. Yang and G. I. Webb, "A Comparative Study of Discretization Methods for Naive-Bayes Classifiers," *Knowledge Acquisition*, vol. 2002, pp. 159–173, 2002.

[146] N. Friedman, D. Geiger, and M. Goldszmit, "Bayesian Network Classifiers," *Machine Learning*, vol. 29, pp. 131–163, 1997.

[147] N. L. Zhang and D. Poole, "Exploiting causal independence in bayesian network inference," *Journal of Artificial Intelligence Research*, vol. 5, pp. 301–328, 1996.

[148] G. F. Cooper, "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks (Research Note)," *Artif. Intell.*, vol. 42, no. 2-3, pp. 393–405, 1990.

[149] K. Baumgartner, S. Ferrari, and G. Palermo, "Constructing Bayesian networks for criminal profiling from limited data," *Knowledge-Based Systems*, vol. 21, no. 7, pp. 563–572, 2008.

[150] M. Ramoni and P. Sebastiani, "Learning Bayesian Networks from Incomplete Databases," no. 13, 1998.

[151] M. J. Beal and Z. Ghahramani, "The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures," *Bayesian statistics 7: proceedings of the seventh Valencia International Meeting, June 2-6, 2002*, p. 453, 2003.

[152] T. C. Fogarty N.S. and Ireson, "Evolving Bayesian classifiers for credit control – a comparison with other machine learning methods.," *IMA Journal of Mathematical Applications in Business and Industry*, vol. 5, no. July, pp. 63–75, 1994.

[153] D. M. Chickering, D. Heckerman, and C. Meek, "A Bayesian approach to learning Bayesian networks with local structure," *UAI'97 Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pp. 80–89, 1997.

[154] D. M. Chickering and D. Heckerman, "A comparison of scientific and engineering criteria for criteria for Bayesian model selection," *Statistics and Computing*, vol. 10, pp. 55–62, 2000.

[155] P. P. Shenoy and G. Shafer, "Axioms for probability and belief function propagation," *Uncertainty in Artificial Intelligence 4*, pp. 169–198, 1990.

[156] P. Shames and J. Skipper, "Toward a framework for modeling space systems architectures," *AIAA 9th International Conference on Space Operations (SpaceOps)*, 2006.

[157] D. Heckerman, A. Mamdani, and M. Wellman, "Real-world applications of Bayesian networks," 1995.

[158] J. Schumann, P. Gupta, and Y. Liu, "Application of Neural Networks in High Assurance Systems : A Survey," vol. 268, pp. 1–19, 2010.

[159] D. Mackall, S. Nelson, and J. Schumann, "Verification Aerospace and Validation of Neural Networks for Aerospace Systems," 2002.

[160] S. A. Jacklin, "Closing the Certification Gaps in Adaptive Flight Control Software," in *Guidance, Navigation and Control Conference*, 2008.

[161] K. W. Przytula, D. Dash, and D. Thompson, "Evaluation of Bayesian networks used for diagnostics," *IEEE Aerospace Conference Proceedings*, vol. 7, pp. 3177–3187, 2003.

[162] R. D. Hawkins, I. Habli, and T. P. Kelly, "The Principles of Software Safety Assurance," in *31th International System Safety Conference*, 2013.

[163] N. T. Nguyen and S. A. Jacklin, "Neural Net Adaptive Flight Control Stability, Verification and Validation Challenges, and Future Research Neural Net Adaptive Flight Control Stability, Verification and Validation Challenges, and Future Research," in *Workshop on Applications of neural networks in high assurance systems-International Joint Conference on Neural networks*, 2007.

[164] E. Yudowsky and E. Yudowsky, "Artificial Intelligence as a Positive and Negative Factor in Global Risk," *Artificial Intelligence*, pp. 1–42, 2006.

[165] V. G. Cerf, "A comprehensive self-driving car test," *Communications of the ACM*, vol. 61, no. 2, pp. 7–7, 2018.

[166] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars," 2017.

[167] D. o. D. (AUS), "MIL-STD-882E Department of Defense Standard Practice System Safety," no. May, 2012.

[168] S. Lee, N. Ybarra, K. Jeyaseelan, J. Seuntjens, I. E. Naqa, S. Faria, P. Brisebois, J. D. Bradley, C. Robinson, S. Lee, N. Ybarra, K. Jeyaseelan, J. Seuntjens, I. E. Naqa, J. D. Bradley, and C. Robinson, "Bayesian network ensemble as a multivariate strategy to predict radiation pneumonitis risk Bayesian network ensemble as a multivariate strategy to predict radiation pneumonitis risk," *Medical Physics*, vol. 42.5, pp. 2421–2430, 2015.

[169] T. G. Dietterich and T. G. Dietterich, "Ensemble Methods in Machine Learning," *Proceedings of the First International Workshop on Multiple Classifier Systems*, pp. 1–15, 2000.

[170] K. B. Laskey, "Sensitivity Analysis for Probability Assessments in Bayesian Networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 6, pp. 901–909, 1995.

[171] M. Pradhan, M. Henrion, G. Provan, B. Del Favero, and K. Huang, "The sensitivity of belief networks to imprecise probabilities: an experimental investigation," *Artificial Intelligence*, vol. 85, no. 1-2, pp. 363–397, 1996.

[172] R. Gray, *Entropy and Information Theory*. 2009.

[173] H. Chan and A. Darwiche, "Sensitivity Analysis in Bayesian Networks: From Single to Multiple Parameters," in *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, 2004.

[174] H. Chan, *Sensitivity Analysis of Probabilistic Graphical Models by*. PhD thesis, 2005.

[175] M. A. Gómez-Villegas, P. Main, and R. Susi, "Sensitivity of Gaussian Bayesian networks to inaccuracies in their parameters," *Proceedings of the 4th European Workshop on Probabilistic Graphical Models, PGM 2008*, pp. 145–152, 2008.

[176] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," pp. 1–10, 2013.

[177] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial Attacks on Classification Models for Graphs," 2018.

[178] S. Geman, R. Doursat, and E. Bienenstock, "Neural Networks and the Bias Variance Dilemma," *Neural Computation*, vol. 4, pp. 1–58, 1992.

[179] R. R. Lutz, "Software engineering for safety," *Proceedings of the conference on The future of Software engineering - ICSE '00*, vol. 1, no. 212, pp. 213–226, 2000.

[180] S. Bhattacharyya, D. Cofer, D. J. Musliner, J. Mueller, and E. Engstrom, "Certification Considerations for Adaptive Systems NASA STI Program . . . in Profile," no. March 2015, 2015.

[181] J. Hill and S. Tilley, "Creating safety requirements traceability for assuring and recertifying legacy safety-critical systems," *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference, RE2010*, pp. 297–302, 2010.

[182] D. Hadfield-menell, S. Milli, P. Abbeel, S. Russell, and A. D. Dragan, "Inverse Reward Design," no. Nips, 2017.

[183] N. V. Chawla, N. Japkowicz, and P. Drive, "Editorial : Special Issue on Learning from Imbalanced Data Sets," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, 2004.

[184] R. Hawkins and T. Kelly, "A structured approach to selecting and justifying software safety evidence," *5th IET International Conference on System Safety 2010*, pp. 3A1–3A1, 2010.

[185] P. G. Bishop and R. E. Bloomfield, "The SHIP Safety Case Approach," no. October 1995, pp. 437–451.

[186] A. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[187] B. Efron and R. Tibshirani, "Improvements on cross-validation: The .632+ bootstrap method," *Journal of the American Statistical Association*, vol. 92, no. 438, pp. 548–560, 1997.

[188] L. Shi, "The MicroArray Quality Control (MAQC)-II study of common practices for the development and validation of microarray- based predictive models," *Nature Biotechnology*, vol. 28, no. 8, pp. 827–838, 2010.

[189] K. R. MURPHY, "Fooling Yourself With Cross-Validation: Single Sample Designs," *Personnel Psychology*, vol. 36, no. 1, pp. 111–118, 1983.

[190] H. Chan and A. Darwiche, "When Do Numbers Really Matter?," *Journal of Artificial Intelligence Research*, vol. 17, pp. 65–74, 2002.

[191] C. Lacave and F. J. Díez, "A Review of Explanation Methods for Bayesian Networks," *The Knowledge Engineering Review*, 2002.

[192] M. Pradhan, G. M. Provan, B. Middleton, and M. Henrion, "Knowledge Engineering for Large Belief Networks," *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI1994)*, pp. 484–490, 1994.

[193] W.-T. Tsai, R. Vishnuvajjala, and D. Zhang, "Verification and validation of knowledge-based systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 1, pp. 202–212, 1999.

[194] P. Bishop, R. Bloomfield, T. Clement, and S. Guerra, *Software Criticality Analysis of COTS/SOUP*. 2002.

[195] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," *Proceedings - Real-Time Systems Symposium*, pp. 239–243, 2007.

[196] A. Burns and R. Davis, "Mixed Criticality Systems - A Review," *Department of Computer Science, University of York, Tech. Rep*, no. July 2016, pp. 1–56, 2013.

[197] A. Rae, R. Alexander, and J. McDermid, "The Science and Superstition of Quantitative Risk assessment," *Journal of Systems Safety*, vol. 48, no. 4, p. 28, 2012.

[198] H. Chan and A. Darwiche, "Reasoning about bayesian network classifiers," *Proceedings of the Nineteenth conference on ...*, 2002.

[199] E. B. Wilson, "Probable Inference , the Law of Succession , and Statistical Inference," *Journal of the American Statistical Association*, vol. 22, no. 158, pp. 209–212, 1927.

[200] B. E. Hanson, C. William III and Marshall, "Artificial intelligence applications in the intensive care unit," *Critical Care Medicine*, vol. 29, no. 2, pp. 427–435, 2001.

[201] B. Höll, S. Spat, J. Plank, L. Schaupp, K. Neubauer, P. Beck, F. Chiarugi, V. Kontogiannis, T. R. Pieber, and A. Holzinger, "Design of a mobile, safety-critical inpatient glucose management system," *Studies in Health Technology and Informatics*, vol. 169, no. January, pp. 950–954, 2011.

[202] I. Lee, G. J. Pappas, R. Cleaveland, J. Hatcliff, B. H. Krogh, P. Lee, H. Rubin, and L. Sha, "High-Confidence Medical Device Software and Systems Recommended Citation High-Confidence Medical Device Software and Systems," *Reprinted from IEEE Computer*, vol. 39, no. 4, pp. 33–38, 2006.

[203] L. Snipen, T. Almøy, and D. W. Ussery, "Mixture Models," vol. 8, no. 1976, pp. 1–8, 2009.

[204] a. G. Stephenson, D. R. Mulville, F. H. Bauer, G. a. Dukeman, P. Norvig, L. S. LaPiana, and R. Sackheim, "Mars Climate Orbiter Mishap Investigation Board Phase I Report," p. 44, 1999.

[205] L. C. V. D. Gaag and S. Renooij, "Analysing Sensitivity Data from Probabilistic Networks Analysing Sensitivity Data from Probabilistic Networks," in *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence*, pp. 530–537, 2001.

[206] M. Douthwaite and T. Kelly, "Establishing verification and validation objectives for safety-critical Bayesian networks," *Proceedings - 2017 IEEE 28th International Symposium on Software Reliability Engineering Workshops, ISSREW 2017*, pp. 302–309, 2017.

[207] M. Douthwaite and T. Kelly, "Safety-Critical Software and Safety-Critical Artificial Intelligence : Integrating New Practices and New Safety Concerns for AI systems," in *Safety Critical Systems Symposium 2018*, Safety Critical Systems Club (SCSC), 2018.

[208] S. Andreason, M. Woldbye, and B. Falck, "MUNIN - A Causal Probabilistic Network for Interpretation of Electromyographic Findings," pp. 366–372, 1986.