Safety Assurance of Aviation Systems

Derek Wade Reinhardt

Doctor of Philosophy

University of York
Computer Science

October 2013

Abstract

From review of historical projects, there is evidence that limitations in contemporary safety assurance approaches for software-dependent systems contribute to programmatic and certification difficulties, e.g. delays and risk retention. These difficulties arise particularly in relation to evaluating risk of systematic behavioural anomalies and evidence shortfalls or deficiencies. These findings question the effectiveness of current safety assurance approaches. Although these problems are general, this thesis is grounded in the context of Australian Defence Force aviation projects.

Through analysing the purpose of safety assurance standards, this thesis establishes principles and guidelines for defining effective safety assurance frameworks for aviation systems. The principles and guidelines are used to define a novel integrated framework which is responsive to the specific challenges of military aviation systems acquisition.

The framework qualifies knowledge of risks and uncertainty, focusing on product behaviour in the architectural context. It is based on evaluation of properties of architecture, including the prevention and tolerance of faults. Knowledge of product behaviours is informed by attributes of supporting evidence, and the tolerability of limitations in evidence. A key factor in the success of safety assurance standards, in an acquisition context, relates to their effectiveness for reducing uncertainty for supplier delivery of safety evidence across contracting processes. Thus this thesis also provides a method for contracting for the novel integrated framework.

Evaluation of the principles, guidelines and framework has been conducted through peer review via workshop and survey questionnaire, analysis against real world aircraft architectures, analysis with respect to historical project data, a constructed example, anti-hypothesis analysis, and evaluation as an audit tool and contract evaluation aid on several projects. Evaluation on an actual project was not possible. A major factor identified in the effectiveness of safety assurance standards is how stakeholders are incentivised (or conversely discouraged) in decision making pertaining to product risk and evidence. This thesis shows that the novel integrated framework, through implementation of the principles and guidelines, could help to avoid the classes of project issues observed historically by enabling developers and assessors to focus on reasoning about the risks of behavioural properties of products, and in the production of evidence used to inform product behaviours. Further evaluation via application to actual projects is required to provide more definitive evidence of benefits and limitations.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisor John McDermid for his help, guidance and encouragement. You've provided me focus and perspective when I've found difficulty wrestling with complexity and ambiguity. I'd also like to thank the staff members at the University of York who have provided guidance.

I would like to thank the Australian Defence Force and Royal Australian Air Force for providing me exposure to numerous opportunities to be dismayed by circumstances brought about by current safety certification regimes, and the motivation this has given me to look to find better approaches. Of course these thanks go equally to the numerous contractors who exercise these certification frameworks.

I would like to thank the Royal Australian Air Force for providing me a professional development posting and ensure the work is published so that its benefits, however modest, can be realised by those suppliers, acquirers and regulators working in the industry.

Finally, I would like to thank my family and friends for their love and support always.

Author’s Declaration

Some of the material presented within this thesis has been previously published in the following papers or reports:

- D.W. Reinhardt “Assurance of Evidence for Software Safety Cases – Qualifying Dissertation”, Department of Computer Science, University of York, 05 Nov 2008.
- D.W. Reinhardt and J.A. McDermid, “Assuring Against Systematic Faults Using Architecture and Fault Tolerance in Aviation Systems”, Improving Systems and Software Engineering Conference (ISSEC) 23-25 Aug 2010.
- D.W. Reinhardt and J.A. McDermid, “Assurance of Claims and Evidence in Aviation Systems”, IET System Safety Conference Oct 2010.
- D.W. Reinhardt and J.A. McDermid, “Contracting for Assurance of Military Aviation Software Systems”, Australian System Safety Conference May 2012.

The work contained within this thesis represents the original contribution of the author.

1 Introduction

Despite the application of contemporary safety and software assurance standards, there is evidence of limitations to these approaches contributing to programmatic and certification difficulties. This evidence of limitations is drawn from Australian Defence Force (ADF) certification activities of aircraft avionics systems and software for the Australian Defence Force (ADF). Historic ADF projects reveal that contemporary approaches do not seem to routinely result in completion of aviation system developments within cost and schedule constraints, nor do they achieve difficulty-free certification by airworthiness regulators. Evidence supporting these observations exists in the form of:

- failed project approvals due to concerns with limitations in evidence,
- cost and schedule increases within projects due to emergence of safety issues, or
- the retention of elevated safety risks by relevant authorities at release to service.

1.1 Certification Challenges in Australian Defence Force Aircraft Avionics Acquisitions and Modifications

The following sub-sections describe two examples from ADF experience where certification challenges have occurred due to evidence shortfalls or late emergence of safety risks.

1.1.1 Flight Control System Example

Several years ago, the ADF was forced to ground a fleet of their aircraft after a series of flight control events during test flying (Australian National Audit Office, 2009). At the centre of the problem was the aircraft's Automatic Flight Control System (AFCS), which was intended to reduce the workload of the crew. The AFCS is a single channel design which is intended to provide stability augmentation and control of aircraft pitch attitude, roll attitude and heading, including autopilot. The AFCS was not intended for 'hands off' operation because it only has limited control authority and the pilot should be able to overcome erroneous AFCS behaviours.

The ADF undertook an investigation to identify the causes of the flight control anomalies, which revealed the following:

- Erroneous data from faulty air data sensors was being processed by the control law computations as valid information, the result being rapid changes to aircraft commanded motion.
- Actuator position sensor or sensor wiring failures caused loss of closed loop control reference signals to the AFCS. The result could be control system runaway, leading to control actuators moving to their full permissible authority.
- Limitations in fault tolerance resulted in non-benign responses to the aforementioned sources of failures. Further investigation revealed that other credible failure scenarios would result in similar non-benign AFCS responses.
- Changes to the cockpit configuration resulted in anthropometric limitations to the range of control input movement, and thus limiting the available manual control authority in parts of the flight envelope. As a result, there are parts of the flight envelope where the crew may not have sufficient control authority to overcome erroneous AFCS commands.

These factors prompted the ADF to undertake a broader investigation of design practices, system safety program and software assurance to seek understanding of how these vulnerabilities were introduced. The investigation revealed the following:

- Aspects of the system safety program for the AFCS design had been conducted retrospective to design activities. Safety arguments sought to justify the established architectural design and implementation, rather than to influence design and architecture via safety design requirements.
- The design solution was based on a digital computer rather than the analogue control system used in former designs to which the historical service history related safety arguments applied.
- Although a simplex architecture, the design solution did not capitalise on opportunities for the implementation of additional fault prevention or tolerance.
- Safety analysis lacked the fidelity and systematic completeness to properly draw conclusions regarding the appropriateness of the behaviours of the software under identified fault conditions.

- Safety evidence associated with external system components such as sensors and some actuators did not systematically identify credible failure modes of these components.
- Software development had a process focus. There was limited product focus in safety arguments based on evidence produced from software development activities.
- Verification evidence of safety-related behaviours of the system relied heavily on the wrong types of verification evidence.
- The program had already suffered cost and schedule overruns, and thus there was commercial pressure to resolve the safety issues swiftly with minimum rework to the design.
- The contract behaved more as an inhibitor to the resolution of safety issues than it did as an enabler for resolution. Contractual dispute featured often in discussions.
- Many of these issues may have been visible, and potentially could have been averted either programmatically or technically, had pre-contract processes sought the delivery of appropriate evidence pre-contract signature and pre-design review milestones. Critical examination of safety arguments earlier in the lifecycle may have also revealed those arguments which were later revealed to be inferior.

Clearly there were many contributing factors in these events. However, an inspection of them reveals that several notable themes do emerge. These themes are suitability of evidence, argument, architecture, and contractual mechanisms. Take note of these, as another example is considered.

1.1.2 Flight Management System Example

Recently, the ADF incurred a delay to the release to service of aircraft with an upgraded Flight Management Systems (FMS) due to the emergence of safety issues during flight test evaluation. These safety issues have emerged in an environment of on-going disputes between supplier and acquirer over limitations to the suitability of the evidence supporting safety arguments for flight systems for this aircraft.

The upgrade primarily includes the introduction of the FMS, developed from an existing civil aircraft FMS, and modified to provide additional military capabilities and interface translation to the existing aircraft mission and flight systems.

Investigation was undertaken into the safety issues (International Program Office, 2010), which revealed the following:

- Waypoint and leg sequencing errors would potentially result in a deviation from flight routes and cause the aircraft to be commanded to fly routes where minimum safe altitude clearance may not be preserved.
- Errors with vertical navigation cues could potentially cause the aircraft to descend below minimum safe altitudes during performance based navigation approaches.
- Inconsistent menu layouts, including the operation of cancel and enter functions, between different operating modes and sub-systems, could cause confusion to operators when commanding critical flight functions.
- Incorrect translation and display of flight parameters on other flight systems with which the flight management system communicates.
- The flight management system would routinely degrade to a non-operational state.

Many of these faults were correlated to safety assurance limitations, as follows:

- Evidence of requirements analysis and decomposition had weaknesses. Requirements, including safety requirements, had not been refined and decomposed to abstractions that could be correlated to their implementation. Neither could they be verified or validated.
- Evidence of consideration of sources of software faults lacked the fidelity and systematic completeness to properly draw conclusions regarding the appropriateness of the behaviours of the software under these fault conditions.
- Fault handling strategies favoured failing the system to a non-operational state rather than provide some resilience against credibly routine sources of faults. The strategy for fault avoidance and fault tolerance was inconsistent with functional dependability objectives.
- Software development had a process focus, with the software level assignment viewed as the mitigation to sources of software faults. There was minimal product focus in safety arguments based on evidence produced from software development activities.
- The trustworthiness of some software evidence was undermined by limitations in review and inspection practices.
- Verification evidence of safety-related behaviours of the system was predominantly an implicit bi-product of functional ground and flight testing. Clear traceability of evidence of safety verification and validation from software development through to flight test was not evident.

- The program had already suffered cost and schedule overruns, and thus there was commercial pressure to resolve the identified issues rather than re-questioning the safety argument and evidence across the design and implementation.
- Contractual arrangements and software planning artefacts behaved more as inhibitors to the safety issues than as enablers for resolution.

As with the flight control system example, an inspection of the factors relating to the flight management system example reveals consistency in the notable themes that emerge. Once again we see themes relating to suitability of evidence, argument, architecture, and contractual mechanisms.

1.1.3 Other Examples

A host of other ADF and international programs have revealed similar themes. This includes a range of different projects including unmanned systems, weapons, navigation system upgrades, mission system upgrades, etc. Analysis of a range of historical ADF programs and their evidence is undertaken within this thesis (refer Chapters 2, 3 and 10) to provide confirmation of the themes highlighted in the two aforementioned examples. As some of these programs are also cooperative with nations such as the United States of America, Great Britain, Canada, Germany, France, Italy and New Zealand; it is reasonable to speculate that these issues are not unique to the ADF.

1.1.4 Identifying the Limitations and Challenges

The themes that have emerged in the motivating examples relate to limitations in the suitability of evidence, argument, architecture, and contractual mechanisms. Elaborating these themes provides insight into the limitations from which they originate, and the motivations for the challenges to resolve them, as follows:

- **Evidence.** Supplier capacity to produce evidence will always be limited because there is never unlimited time or money. Therefore it is important they provide evidence which materially contributes to assurance of safety. However, rationale on how evidence contributes to safety often differs between suppliers, acquirers and regulators. Supplier, acquirer, and regulator assessments of the suitability and sufficiency of evidence supporting safety arguments will also often differ. Therefore, it would be beneficial if there were approaches for achieving consensus regarding suitability and sufficiency of evidence.

- **Argument.** Operational authorities use safety arguments from safety cases to inform decisions about risk treatment or retention (Defence Aviation Safety Authority, 2011). Decisions about risk treatment or retention are product focused, and relate to hazard mitigations provided by design features, controls or guards, workarounds and operator intervention. But safety arguments relating to software intensive systems often have a greater process focus, rather than product focus. This makes operational decisions about risk treatments or retention difficult. While it is possible to make software safety arguments product focused, as shown by (Weaver, 2003), this approach is not routinely encountered in real project aviation system safety arguments outside of the UK. The argument also influences the production of evidence; hence an inferior argument can lead to inferior evidence production.
- **Architecture.** Design mitigations to hazards are achieved from deliberately designed behaviours of a system, many of which emerge from architectural properties of a system. However when fault avoidance and fault tolerance architectural properties are overlooked, the systems in the motivating examples have become prone to hazardous behaviours. For more serious hazards, the motivating examples indicate that single defences were inadequate. Hence approaches are required for achieving consensus between suppliers, acquirers and regulators regarding the suitability of architecture in the presence of faults.
- **Contracts.** For many military aviation system developments, the relationships between supplier, acquirer and regulator are articulated through a contract. However, the evidence suggests that contractual arrangements don't seem to be helping with the resolution of safety issues; in fact the evidence is that it may be inhibiting effective resolution within cost and schedule constraints. Ambiguity and uncertainty seem to be significant factors. However, there are many opportunities within contractual establishment processes that could be exploited to improve this circumstance.

1.1.5 Addressing the Limitations and Challenges

Despite the on-going application of contemporary assurance standards, safety issues and limitations in safety evidence are still the causes of cost and schedule overruns for projects. Likewise they continue to contribute to difficulties in airworthiness regulators completing certification free from risk retention. It is these issues that have provided motivation for the research presented within this thesis. The thesis proposition provides insight into how the author proposes to address the issues relating to evidence, argument, architecture and contracts in the context of military aviation systems.

1.2 Defining the Key Concepts

Great importance is placed upon the terminology used. Terms like evidence, argument and architecture have emerged, as have concepts of decisions on risk treatments and retention, safety cases, and assurance. Therefore in order to ensure consistent interpretation, it is important that the terminology and concepts are defined. It is also important to note that some of these concepts and terms may take on more specific meaning than those attributed under general English language interpretation. Furthermore, there is a proliferation of sometimes conflicting definitions for some of these terms throughout the safety community. Hence this section also serves the purpose of articulating which interpretation applies to the work described by this thesis. Every effort has been made to adopt terms defined by pre-existing and recognised literature, however where necessary this thesis provides a 'local' definition of terms.

A goal of this thesis is to assist with the achievement of safety.

Safety

The expectation that a system does not, under defined conditions, lead to a state in which human life is endangered.

Defence Standard 00-56 Iss 2 (Ministry of Defence, 1996)

When safety is not achieved, accidents may occur. Hence a goal to achieve safety is a goal to prevent accidents or reduce their impact on human life.

Accident

An unintended event, or sequence of events, that causes harm. Where harm is defined as death, physical injury or damage to the health of people, or damage to property or the environment.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

In contemporary practice the achievement of safety is articulated and justified through a safety case.

Safety Case

A structured argument, supported by a body of evidence that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given operating environment.[1]

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

The Safety Case is a key artefact used for the certification of systems.

Certification

The end result, which is usually recorded in a certificate, of a process that formally examines and documents compliance of a product against pre-defined requirements to the satisfaction of the certifying authority.

Local Definition adapted from (Wade, 2009)

Core to the safety case is the concept of the safety argument.

Safety Argument

A logically stated and convincingly demonstrated reason why safety requirements are met.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

Arguments can be defined in more general terms.

Argument

An argument is a rationale in which the reason presents evidence in support of a claim made in the conclusion. Its purpose is to provide a basis for believing the conclusion to be true.

(Mayes, 2013)

Arguments are one of two kinds of rationale.

[1] This definition implies that all safety cases are compelling, comprehensible and valid. This may not always be the case, but should be true of 'acceptable' safety cases.

Rationale

Rationales are models used to reveal the logical relationships underlying our reasoning There are two types of rationale: argument and explanation.

(Mayes, 2013)

In this thesis rationale is used where both argument and explanation are implied.

Explanation

An explanation is a rationale in which the reason presents a cause of some fact represented by the conclusion. Its purpose is to help us understand how or why that fact occurs.

(Mayes, 2013)

Confidence is achieved through the concept of assurance.

Assurance

Adequate confidence and evidence, through due process, that safety requirements have been met.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

The other core aspect of the safety case is evidence which supports both the safety argument and the assurance of the safety argument.

Evidence

Information indicating whether a belief or proposition is true or valid.

Oxford English Dictionary (Oxford University Press, 2010)

A widely used strategy for arguing safety is to argue that all risks have been reduced to As Low As Reasonably Practicable (ALARP).

Risk

Combination of the likelihood of harm and the severity of that harm.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

As Low As Reasonably Practicable (ALARP)

A risk is ALARP when it has been demonstrated that the cost of any further Risk Reduction, where the cost includes the loss of Defence capability as well as financial or other resource costs, is grossly disproportionate to the benefit obtained from that Risk Reduction.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

Risks are reduced by treating either likelihood or severity, or both. It is common practice to use Probabilistic Risk Assessment (PRA) to assess risk, although further discussion on the merits and drawbacks of PRA are discussed later in this thesis.

Systematic identification of risks implies a systematic identification of hazards from which risks emerge.

Hazard

A physical situation or state of a system, often following from some initiating event, that may lead to an accident.

Defence Standard 00-56 Iss 4 (Ministry of Defence, 2007)

The initiating event leading to a hazardous state is a failure.

Failure

An event that occurs when the delivered service deviates from correct service. A service fails either because it does not comply with the functional specification, or because this specification did not adequately describe the system function. A service failure is a transition from correct service to incorrect service, i.e., to not implementing the system function.

(Avizienis, et al., 2004)

The deviation that causes the failure is called an error.

Error

A deviation of the external state of the system from the correct service state.

(Avizienis, et al., 2004)

The cause of an error is a fault.

Fault

The adjudged or hypothesized cause of an error. Faults can be internal or external to a system. The prior presence of a vulnerability, i.e., an internal fault that enables an external fault to harm the system, is necessary for an external fault to cause an error and possibly subsequent failure(s).

(Avizienis, et al., 2004)

Succinctly, (Avizienis, et al., 2004) depicts the relationship between fault, error and failure as shown in Figure 1.

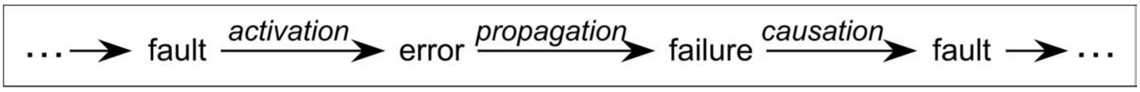


Figure 1: Fault, Error and Failure relationships

Since the development of the digital computer, software has played an increasingly important role in the control and operation of safety-related functions.

Software

Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

IEEE 610.12 (IEEE, 1991)

Faults can be characterized as either random or systematic. Faults in software are systematic (Weaver, 2003), and thus can’t be characterised by random process models.

Systematic Fault

Faults caused by design and implementation errors made by developers (i.e. humans or tools) during system design, development or manufacture, or by human error during operation or maintenance.

Local Definition adapted from (Weaver, 2003)

Hence assessing the likelihood of a systematic software fault cannot use probabilistic methods. Historically safety cases have argued the safety of software based on an appeal to the development processes. However (Weaver, 2003) and the examples at the start of this introduction showed why this may not be sufficient.

1.3 Thesis Proposition

This thesis investigates the following proposition:

It is feasible to establish principles for defining effective safety assurance frameworks. These principles enable frameworks to be developed to satisfy safety objectives for military aviation systems in typical acquisition contexts.

This proposition is supported:

- by showing that the principles are based on concepts that preserve the benefits and reduce the limitations of existing assurance paradigms for the military certification situation;
- by examining the principles in the practical context of historical military aviation system projects; and
- through the development and application of a novel integrated framework for assurance of aviation software systems, addressing identified deficiencies in existing assurance frameworks.

1.4 Research Paradigm, Activities and Criterion

The research presented within this thesis is based on a form of engaged scholarship defined by (Van de Ven, 2007) as *Design and Evaluation Research.* This form of research examines questions dealing with design and evaluation of models for solving practical problems. It includes an undertaking to provide a plausible explanation of the problem, as well as the collection and analysis of evidence-based knowledge of the feasibility and usefulness of the proposed approach to applied problems. While this approach differs to more traditional forms of basic science which are used to describe, explain or predict a phenomenon, it is necessary because the problems motivating this research are practical rather than theoretical problems.

Contributions from theoretical research are amalgamated into this research, and are thus described by this thesis. However the focus is with respect to the practical problem and means of addressing the problem. This leads to an important assumption behind this work. The approach described by this thesis provides a framework focused at providing a pragmatic solution within the constraints imposed by stakeholders to the problem. To achieve this, the approach may make compromises of theoretical perspectives in exchange for pragmatism and practicality where there is benefit to either execution or outcome. Where deviations from theoretical perspective exist, this thesis identifies and provides rationale for them.

This research provides a practical contribution to knowledge by addressing the criterion and questions posed by the study activities of (Van de Ven, 2007) model for engaged scholarship, summarised as follows:

- ***Relevance*** of the research is addressed by demonstrating that the problem formulation is based on historical experience of the industrial application of contemporary safety assurance paradigms by suppliers, acquirers and regulators.
- ***Validity*** of the research is addressed by demonstrating that the general principles of the approach are based on deduction, induction or abduction of strengths and weaknesses of the practical application of contemporary safety assurance paradigms.
- ***Truth (Verisimilitude)*** of the research is addressed by demonstrating that it is plausible that the approach proposed by this thesis provides a better result than contemporary safety assurance paradigms through examination of the approach against historical project evidence, a constructed example based on real system designs, survey review by stakeholders familiar with the problems, and application to a current project as an evaluation tool. This thesis doesn't strive to show that the approach developed here is the mature end point for this research; instead it shows that the approach offers benefits over existing approaches. This is an acceptable research goal according to (Van de Ven, 2007)'s model.
- ***Impact*** of the research is addressed by engaging stakeholders to interpret the feasibility and usefulness through communication, interpretation and negotiation of the findings through participative evaluation activities, such as surveys, and through the findings presented within this thesis.

1.5 Thesis Scope

Due to their holistic nature, the research question and thesis proposition posed in Sections 1.1.4, 1.1.5 and 1.3 could be potentially interpreted as a very encompassing, leading to a lack of bounds on the work. This is not the intent, and the research question and thesis proposition presented within this introduction are intended only to give readers a general introduction to the topic of research. In order to provide a more tangible and thus measureable body of work within this thesis, Chapter 2 uses the literature survey to provide a more extensive explanation of the limitations and challenges introduced in Section 1.1 and uses these to define a detailed research question and thesis proposition (refer Chapter 2).

1.6 Thesis Structure and Layout

This thesis is structured into the following chapters:

Chapter 2 presents a survey of literature on the certification of safety-related aviation systems and the associated contemporary approaches for safety assurance. Limitations with the current approaches are identified and explained, and potential treatments to the limitations are described and compared. Based on the limitations a detailed research question is stated and an elaborated thesis proposition presented.

Chapter 3 establishes general principles for safety assurance frameworks that are used throughout the thesis for the development of assurance frameworks for architectural, claims and evidence assurance. These general principles are also used in the evaluation to evaluate the proposed assurance frameworks against motivating issues. A constructed example is introduced that is used throughout the remaining chapters of this thesis for explaining and evaluating the proposed assurance framework.

Chapter 4 focuses on architectural assurance. The chapter summarises the concepts of fail-safe design, fault avoidance and tolerance. An examination of the architectures and fault avoidance/tolerance behaviours of real world aviation software systems is presented, and correlated to the principles of the fail safe design from the systematic failure perspective. Observations regarding the handling of systematic faults by the real world systems and the consideration of the fail-safe design criteria in this context lead to the development and explanation of meta-arguments for architectural assurance, and the definition of the Architectural Safety Assurance Level (ASAL) concept.

Chapter 5 focuses on product behavioural knowledge. The chapter summarises the role of knowledge of product behaviours in safety arguments and examines contemporary approaches to providing assurance of safety arguments. Using principles derived from contemporary approaches for safety arguments and software assurance, meta-arguments for assurance of product behaviours are defined, and the Claims Safety Assurance Level (CSAL) is developed and explained. The relationship between ASALs and CSALs is also explained.

Chapter 6 focuses on evidence assurance. The chapter describes a categorisation of evidence types and discusses the roles of differing evidence types. The chapter proposes the concept of 'Tolerability of Limitations' and defines the Evidence Safety Assurance Level (ESAL) based on this concept. The impact on evidence assurance is described with respect to properties of evidence including relevance, trustworthiness and results.

Chapter 7 describes the challenges of the certification environment of aviation systems in both the civil and military contexts. Unique circumstances of the military context are identified as challenges for achieving assurance objectives, most specifically being the need to articulate and enforce the assurance requirements via contract rather than legislation. Contracting paradigms are examined, as are acquisition paradigms. An approach is proposed for contracting for the assurance of military aviation systems, for which a specific instantiation for the ASAL/ESAL/CSAL frameworks is described. Guidance on the conduct of tender and contract execution processes is also provided.

Chapter 8 details the historical problems of relating assurance to risk evaluation. The chapter examines alternatives to probabilities in risk matrices and re-defines risk based on a strength of defences paradigm.

Chapter 9 recognises that there are a number of assumptions that may impact the feasibility of the proposed framework These include imperfect hazard analysis, the suitability of architectural factors, independence, managing change, and systems of systems.

Chapter 10 describes how the proposed approaches have been evaluated. The evaluation of the work is based on peer and survey review, review of historical project evidence, anti-hypothesis analysis, application of the framework to a constructed example based on real system designs, application of the framework to a current project as an audit/evaluation tool.

Chapter 11 presents the conclusions established from this body of research work. It describes the extent to which the work presented in the previous chapters supports the thesis proposition, and identifies topics for future work.

The body of this thesis is supplemented with additional supporting material presented in several Appendixes, as follows:

Appendix A presents the technical description of architectural fault avoidance and fault tolerance mechanisms of actual aviation systems including the flight control systems of the Boeing 777, Airbus A330, C-17A, and F/A-18A/B; and the flight management systems or mission computers of the Boeing 777, Airbus A330 / KC-30A, F/A-18A/B, and C-130J.

Appendix B presents the taxonomy of attributes of software lifecycle products referenced by Chapter 5.

Appendix C provides an example Tender/Contract Statement of Requirement (SOR), Statement of Work (SOW), Data Requirements List (TDRL/CDRL) and the associated Data Item Descriptors (DIDs) for the contracting framework described in Chapter 7.

Appendix D provides the survey evaluation forms and results

Appendix E summarises the results of the review of historical projects.

2 Survey of Assurance of Evidence for Safety Cases

2.1 Introduction

Chapter 1 introduced the notion that contemporary approaches to safety assurance are not routinely resulting in completion of aviation system developments within cost and schedule constraints. Nor are they achieving difficulty-free certification by airworthiness regulators. Safety issues and limitations in safety evidence are often the causes of cost and schedule overruns for projects. Experience in military aviation systems suggests that these outcomes are the result of limitations in evidence, argument, and architecture; and the articulation of requirements for these in contracts between suppliers, acquirers and regulators.

This chapter expands the introduction by presenting a survey and analysis of current standards, literature and applicable research on the safety assurance of aviation systems. The intent is to summarise the current approaches for the purposes of analysing and emphasising the benefits and limitations with these approaches. The chapter also provides the context for the contribution made by this research. Additional survey material is also introduced within later chapters as required to establish and contextualise the explanation of the author's contributions.

This chapter is divided is divided into the following sections:

- **Background on Certification for Safety-related Aviation Systems** – The process by which regulators undertake safety certification of aviation systems, and the opportunities and constraints of these processes.
- **Communication and Enforcement of Certification Requirements using Contracts** – Outlines the issues for contracts communicating certification requirements.
- **Current Approaches for Safety-related Aviation Systems** – The current industrial approaches, including standards, for developing and evaluating safety-related aviation systems.
- **Background on the Safety / Risk Case** – The use of the safety case or risk case in certification evaluations and for informing decisions on operational risk treatment or retention. The structure of the safety case and the techniques for presenting safety arguments and reasoning about evidence.
- **A Discussion of Current Approaches for Safety-related Aviation Systems** – A comparison of the benefits and limitations of the current industrial approaches.

- **Potential Approaches to Addressing the Limitations with Current Approaches** – Emphasises the focus for research that offers solutions to the problems with the current approaches.
- **Thesis Contribution** – Based on the survey presented, the research questions and contribution of this thesis is described.

2.2 Background on Certification of Safety-related Aviation Systems

2.2.1 The Concept of Certification

A local definition for certification was introduced in Section 1.2 which was adapted from (Wade, 2009). For aircraft designs this means that certification is the process that examines and documents compliance of the aircraft or aircraft modification (i.e. the product) against pre-defined 'airworthiness' requirements and standards to the satisfaction of the certifying authority.

Similar interpretations of certification also exist for products within the domains of military equipment, railways, power generation, manufacturing and processing plants, and medical devices. Although it should be emphasised that the processes and certificates by which this is achieved, and the frameworks and regulations governing such activities may be notably different to aviation. There are also differences between civil and military aircraft certification approaches, but these shall be elaborated in forthcoming paragraphs.

Other areas of the aviation sector (e.g. parts manufacturing and maintenance venues), and also entirely different domains (e.g. quality assurance, insurance and finance) may also use a broader definition of certification that encompasses process and/or organisational compliance, perhaps in addition to product compliance.

These product, process and organisational certification themes are reflected in the following definitions of certification identified from the literature:

- *"To attest by a certificate"*, where a certificate is *"a writing on paper certifying to the truth of something"* (The Macquarie Library, 2002)
- *"An official document attesting a fact, in particular"* (Oxford University Press, 2010)

- *"The process of assuring that a product or process has certain stated properties, which are then recorded in a certificate"* (Committee on Certifiably Dependable Software Systems, 2007)
- *"The end result of a process that formally examines and documents compliance of a product, process or organisation against pre-defined requirements to the satisfaction of the certifying authority."* (Wade, 2009)
- *"Legal recognition by a certifying authority that a product, service or organisation complies with applicable requirements. Such certification comprises the activity of checking the product, service, organisation or person and the formal recognition of compliance with the applicable requirements by issue of certificate, license, approval or other document as required by national law or procedures. In particular, certification of a product involves:*
  - *(a) the process of assuring the design of a product to ensure that it complies with a set of standards applicable to that type of product so as to demonstrate an acceptable level of safety, (acceptable risk);*
  - *(b) the process of assessing an individual product to ensure that it conforms to the certified type design;*
  - *(c) the issue of any certificate required by national laws to declare that compliance or conformity has been found with applicable standards in accordance with item(a).* (Aviation Glossary, 2012)

A number of important points should be made about these definitions.

Firstly, while the emphasis in the dictionary definitions pertains to the issuance of a certificate, the industrial engineering definitions of certification emphasise it as a process of assessing compliance/achievement, not just the act of issuing the certificate. The act of recording the results in a certificate is simply the final step in the process.

Secondly, certification involves at least two parties – an applicant and a certifying authority, and sometimes three if independent assessors are involved. The role of the certifying authority is to evaluate the evidence presented by the applicant against the pre-defined requirements and standards, and determine compliance/achievement. This differs somewhat from the concept of 'self-certification', which implies emphasis on attestation rather than on evaluation by an independent certifying authority. However, the certifying authority may require the applicant to self-certify their compliance/achievement as a component of the evidence that the certifying authority examines.

Thirdly, certification requires that pre-defined requirements and standards be specified as the benchmark for certification. The applicant and the certifying authority require a common benchmark against which evidence can be produced by the applicant, and the evidence evaluated by the certifying authority. The role of standards is discussed further in Section 2.2.2.

Finally, certifications of products that can cause harm inevitably involve the determination of the level of risk. Hence safety is a fundamental element of certification for aviation systems.

As this thesis is concerned with the product safety aspects (i.e. acceptable/tolerable safety risk) of aviation systems, discussion regarding process and organisational compliance will be limited to where they form part of existing standards, contemporary approaches or sources of evidence. The emphasis throughout this thesis will be product safety achievement. Hence, in the context of the product focus of the certification definitions provided, this thesis examines ways that the supplier and certifying authority relationship can effectively and practically achieve product safety with acceptable risk.

2.2.2 Role of Standards in Certification

Standards are fundamental to certification because they:

- are often used as a tool for communicating certification requirements;
- may also be used as a preferred means or guidance for demonstrating or assessing compliance for certification requirements;
- provide a way of packaging requirements for a specific topic or technology;
- reduce subjectivity of general principles they embody;
- reduce variability of acceptable solutions to specific problems;
- provide a way of providing re-use of certification benchmarks; and
- are developed by a process of consensus amongst relevant stakeholders, which may include input from both applicant and certifying authority stakeholders.

However, the effectiveness of standards achieving these fundamentals depends on the type of standard. The following sub-section examines the different types of standards, and how the type of standard may benefit or limit its effectiveness.

2.2.3 Types of Standards

(McDermid & Rae, 2012) propose that there are different types of standards:

- those that relate to technical aspects of products,
- those that relate to whole products, and
- those that relate to process.

Standards for Technical Aspects of Products

An example of a standard relating to technical aspects of a product is G88-05 *"Standard Guide for Designing Systems for Oxygen Service"* (ASTM International, 2005). This standard defines specific performance and safety requirements for oxygen systems including material selection, design methods, causal factors to fire risks, test methods, acceptability criteria, and approaches than minimise risk of a fire. The standard is flexible enough to permit design of oxygen systems for applications such as medical devices, aircraft oxygen systems, air separation plants, and spacecraft, while capturing the fundamental product features and safety devices that should be incorporated to produce an acceptably safe oxygen system design. Many other such standards exist covering technological aspects of aviation systems products including:

- environmental requirements (temperature, humidity, dust, vibration, shock, etc.),
- electromagnetic compatibility,
- electronic circuit board design,
- electrical wiring,
- pneumatic systems,
- hydraulic systems,
- structural integrity, etc.

Standards for Whole Products

Somewhat more encompassing are those standards that relate to a whole product (e.g. complete aircraft, ship, railway; or whole system). Some examples are:

- Aircraft certification standards such as Title 14 code of Federal Regulations Part 25 *Airworthiness Standards for Transport Category Airplanes* (National Archives and Records Administration, 2012) or the equivalent European standard.
- Aircraft systems standards such as Technical Standard Order (TSO) TSO-C129a *Airborne Supplemental Navigation Equipment Using GPS* (FAA, 1996).

Standards for Processes

Finally, there are the process standards. Some examples of these standards are:

- System Safety Standards such as Aerospace Recommended Practice (ARP) 4754 (SAE Aerospace, 2010), Defence Standard 00-56 (Ministry of Defence, 2007), MIL-STD-882 (US DoD, 2000), and Def (Aust) 5679 (Australian Department of Defence, 2006).
- Software Lifecycle Standards such as ISO/IEC 12207 (ISO/IEC, 2008)

Additional Classifications of Standards

Further to the product and process distinction identified in the paragraphs above, standards maybe classified as *prescriptive* or *goal-based*, although some standards may incorporate elements of both paradigms. Prescriptive standards are used by certifying authorities to tell applicants what to achieve (i.e. outcomes or objectives), and how to go about it (i.e. methods and techniques). Goal-based standards, on the other hand, set objectives saying what has to be achieved (i.e. outcomes or objectives), but don't saying how to go about it (although supplemental guidance may provide examples of how to comply).

There is substantial academic and industrial debate regarding the preference for prescriptive or goal-based standards. Table 1 provides a summarising model produced by the author of the benefits and limitations of prescriptive versus goal-based standards incorporating arguments made by:

- (McDermid & Rae, 2012)
- (Committee on Certifiably Dependable Software Systems, 2007)
- (Kelly, 2008)
- (Kelly, et al., 2005)

Table 1 highlights that there is both a symmetry (italicised) and asymmetry (underlined italicised) between the benefits and limitations of the approaches. The collective symmetric properties for each approach define the paradigm, and this influences perceptions regarding the utility of the approaches depending on the 'world-view' of the supplier and certifying authority (McDermid & Rae, 2012). This leads to the perspectives (variability and subjectivity) reflected in the asymmetric properties of each approach. The effects of these benefits and limitations will be examined in more detail in Section 2.3 as they apply to safety assurance standards.

	Goal-based	Prescriptive
Benefit	*Flexible:* Provides greater flexibility for suppliers' solutions, encouraging novelty and technology innovation in solutions. *Methodisible.* Selection of techniques and methods can be based on a specific system, problem or design solution. *Enduring:* Standards do not require updating in response to changes in technology or knowledge of techniques/methods. *Deductive:* The intent of the standard will follow directly from the outcomes and objectives it specifies.	*Adjuring:* Requires suppliers to do enough of the things (i.e. techniques, methods, solutions) the regulator views as right. *Educating:* Provides a means of educating potential suppliers on the right approaches. *Uniformity:* Minimises variation in approaches used to conform, and thus simplifies the regulator evaluation. *Assessable-invariability:* Strong prescriptions may reduce variability of assessor assessments.
Limitations	*Abjuring:* May not provide a clear benchmark of the things the regulator views as right. May provide too much flexibility for suppliers, leading to confusion. *Non-educating:* Does not provide a means of educating potential suppliers on the right approaches, only acceptable outcomes. *Non-uniformity:* May result in unnecessary variation in approaches used to conform, and thus may complicate the regulator evaluation. *Assessable-subjectivity:* Assessments of the extent to which achievement of goals is compelling may differ between assessors.	*Inflexible:* Limits supplier choice, and thus potentially inhibits novelty and innovation in design. *Non-methodisible:* Selection of techniques and methods is based on prescriptions rather than being based on a specific system, problem or design solution. *Non-enduring:* Standards may require frequent updating in response to changes in technology or knowledge of techniques/methods, or the standards may end 'out of date'. *Inductive:* The desired rationale for the prescriptions achieving the outcomes may not be compelling nor absolute.

Table 1: Benefits/Limitations of Goal-based and Prescriptive Standards

2.2.4 Empowerment of Certification Environments

Another factor affecting certification and the usage of standards as certification requirements or benchmarks is the way the certification authority is empowered, and thus how enforcements of certification requirements is achieved. This is best illustrated by examining the differences between the civilian and military aviation certification environments.

Civil Aviation Certification

Consider the civil aviation certification environment. The Federal Aviation Administration (FAA), the civil aviation airworthiness regulator in the United States of America, issues certificates for new and modified aircraft and aircraft equipment. This certification is relied upon by the customers (owners and operators) who purchase and operate the aircraft. The FAA approach is also common to other civil aviation National

Airworthiness Authorities (NAAs) around the world (e.g. Australia – Civil Aviation Safety Authority (CASA), UK – Civil Aviation Authority (CAA), Europe – European Aviation Safety Agency (EASA)).

In this environment the roles of the developer, manufacturer, owner, operator and regulator are typically separated amongst different organisations or entities. This separation affords each of these entities some opportunity for independence in their function. For example, the owner and operator might be the same organisation (e.g. Qantas), whereas the supplier/developer/manufacturer might be an aircraft developer/manufacturer (e.g. Airbus or Boeing), and the regulator is a government agency (e.g. FAA, CAA, CASA, EASA). In addition, the prime developer and manufacturer are supported by a suite of sub-contractors that develop and manufacture aircraft systems and subcomponents.

It is important to note that the civil regulator is supported by regulations that are indoctrinated in law, and are therefore legally enforceable by the regulator onto those to which they apply (developer and manufacturer). For example, in Australia the Air Navigation Act 1920 (Commonwealth of Australia, 1920), and the Civil Aviation Act 1988 (Commonwealth of Australia, 1988) define 'Australian' aircraft, for which CASA are responsible for promulgating and enforcing Civil Aviation Safety Regulations (CASRs). The regulations effectively become an extension of the law. The CASRs then communicate the certification requirements and CASA performs compliance assurance against them. A similar arrangement exists for the United States (United States of America, 2012) and UK/European environments.

Because they are enforceable by law, then most civil aviation suppliers factor the costs of undertaking this certification into their underlying business model and project costing. Those that don't, find themselves with a non-viable business model. The existence of consultancy businesses that specialise in guiding and recovering aircraft system developments with respect to civil certification requirements, is evidence of the seriousness with which developers are required to comply with certification requirements. Some examples of such business are listed at (Airsearch, 2008), with a specific example being (Certification Services, Inc., 2012)). However, it also suggests that naivety of certification requirements by prospective suppliers is also commonplace. While all new aircraft developments represent a business gamble by the developer, their gamble is with the airline market buying their product, not with the certification authority on the production of certification evidence and compliance with certification requirements. The supplier is incentivised to comply with certification requirements; else they can’t sell their product when it is completed. After all, no airline would buy an aircraft or aircraft system which didn’t achieve certification in the civil context.

Military Aviation Certification

However, in the military aviation certification environment the regulator is not so overtly independent of the acquirer. Unlike civilian aviation arrangements, many militaries around the world are owners, operators and regulators; and to some extent developers and manufacturers. The militaries are their own regulators or airworthiness authorities because they require flexibility to do things civilian operators would never need, such as: low flying, combat, close proximity flying, special modifications, stores clearances, contingency maintenance, battle damage repair, and operational imperatives involving safety versus capability trade-offs; none of which are regulated by the civil authorities (Wade, 2009). This situation is reflected in the way militaries are empowered by laws to perform this regulation. For example laws pertaining to empowering the military (e.g. Defence acts, etc.), and workplace health and safety legislation (refer to Table 2) are the legal mechanisms used to delegate the responsibility for airworthiness.

These military airworthiness authorities typically define regulations that govern the conduct of their activities, however unlike the civil regulations, these regulations are open to discretion by the military regulator/authority to allow trade-offs between providing capability and safety based on the current military climate (e.g. war operations, peace support, counter terrorism, humanitarian assistance, peacetime training, etc. (Royal Australian Air Force, 2007)). For example, in Australia the Air Navigation Act of 1920 (Commonwealth of Australia, 1920) defines ‘State’ aircraft and designates the Chief of the Air Force (CAF) as the Defence Aviation Authority for Air Force, Army and Navy aircraft. Through internal Defence Instructions (DI(G) OPS 2-2 *Defence Aviation Safety Program* (Defence Aviation Safety Authority, 2011)), airworthiness management is separated into technical and operational responsibilities. The instruction also distinguishes the functions of the regulators (i.e. the entities that write the technical and operational regulations) versus the authorities (i.e. the entities that are responsible for interpreting the regulations and making the discretionary trade-offs (via risk treatment or retention) between capability and safety).

To illustrate this point, Table 2 identifies the military airworthiness laws, orders, regulations and publications applicable to airworthiness certification that were reviewed in the conduct of this survey for the ADF, United Kingdom (UK) Ministry of Defence

(MoD), and United States Department of Defense (United States Air Force (USAF), United States Navy (USN), and United States Army).

	Australian Defence Force	United Kingdom Ministry of Defence	United States Air Force	United States Navy	United States Army
Legal Origins	Air Navigation Act Defence Act	Army and Air Force Act Health and Safety at Work Act	National Defense Authorization Act		
Responsibilities	DI(G) OPS-2-2 Defence Aviation Safety Program	MAA Charter from Secretary of State for Defence MAA01: MAA Regulatory Policy	Air Force Policy Directive 62-6 USAF Airworthiness	NAVAIRINST 13034.1 – Flight Clearance Policy for Air Vehicles and Aircraft Systems	Army Regulation (AR) 70-62 – Airworthiness Qualification of U.S. Army Aircraft Systems
Regulations / Instructions	AAP7001.048 – ADF Aviation Safety Program AAP7001.053 – Technical Airworthiness Management Manual AAP8000.010 – ADF Operational Airworthiness Manual	Regulatory Articles (RA): 1000 Series - General Regulations 2000 Series - Flying Regulations 3000 Series - Air Traffic Management Regulations. 4000 Series - Continuing Airworthiness Engineering Regulations 5000 Series - Design and Modification Engineering Regulations	Air Force Instruction 62-601 - USAF Airworthiness	NAVAIRINST 13034.1 – Flight Clearance Policy for Air Vehicles and Aircraft Systems OPNAVINST 3710.7 – NATOPS General Flight and Operating Instructions	Army Regulation (AR) 70-62 – Airworthiness Qualification of U.S. Army Aircraft Systems AR 95-1 Flight Regulations AR 385-16 System Safety Engineering and Management
Certification Requirements	AAP7001.054 – Airworthiness Design Requirements Manual	Defence Standard 00-970 – Design and Airworthiness Requirements for Service Aircraft	MIL-HDBK-516B – Airworthiness Certification Criteria		

Table 2: Military Airworthiness Certification Laws, Orders and Publications

Unlike the civil airworthiness regulations, the military airworthiness regulations are typically described in military orders, instructions and publications which constitute lawful orders to those military and civilian government staff applying them (i.e. the regulator, acquirer, and operator). However, they are not necessarily legally binding to those developers and manufactures (i.e. suppliers) supplying equipment to the military. Instead the contract between supplier and acquirer is the primary means by which requirements are set for suppliers and by which compliance and enforcement of these requirements is achieved. Military contracts typically achieve this by ensuring that relevant contract clauses between their suppliers and government reference the applicable regulations and safety standards. However the earlier discussion highlights that referencing regulations and safety standard may not be all that's required by the contact, and that the contract may also need to establish the roles and relationships between supplier and certifying authority.

Contracts are instruments which provide a legally binding agreement for the purchase/exchange of goods or services. A contract normally consists of terms and conditions, and is supported by technical annexes to define the requirements for goods/services and scope of work. For aviation systems, contracts are used for the acquisition and/or modification of these systems between the developer/manufacturer (i.e. supplier) and the owner or operator (i.e. acquirer). While there is a branch of legal studies associated with contract law, this law pertains to the lawful execution of contracts, and not the enforcement of certification requirements within contracts.

Comparison of Military and Civil Certification Environments

In the civil case, suppliers were incentivised by the laws and by a motivation to sell their product. They also desire to not become bankrupt in doing so. However in the military context, the role of the contract changes the sources of incentivisation. The laws and motivation to sell the product are no longer the key incentive; instead the incentive is contractual compliance while preserving profit margins. Motivation to sell their product is limited because the contract already guarantees payment if they achieve contractual compliance. The legal responsibility for airworthiness and safety mostly falls onto the acquirer because of the way the responsibilities are empowered by law. Health and safety laws do provide some incentive for suppliers to develop products with safety in mind; however because prosecution under these laws tends to only occur retrospective to an accident, they are not in isolation effective certification incentives.

Hence it can be seen that the means for communicating, incentivising and enforcement of certification requirements differs between the civilian (i.e. laws and regulations) and military (i.e. contracts) aviation system cases. This variation occurs even though the role of regulators in these two domains is holistically similar, as are some of their practices in the conduct of certification processes. Therefore, it is apparent that there are differences between the role of the civil airworthiness authorities, and some military regulators and airworthiness authorities. The differences are particularly notable with respect to the level of independence of the regulator from the other entities in the certification environment (owner, and operator), and the potential for incentives and legal enforcement of their requirements.

Other Certification Environments

Outside the scope of airworthiness, separate provisions apply to safety in society, as well as the regulation of technologies that are not aircraft. Parallels can be drawn with regards to the regulation of these other technologies (e.g. Ships, Vehicles, Weapons, etc.) and the aviation case.

There are also other industries outside the aviation industry where the regulator plays a very much more passive role than in the aviation case. Consider consumer product safety. In this industry regulations and standards are empowered by consumer product, environmental and health and safety laws. However, the regulator is much more passive than in the aviation case. Responsibility for complying rests solely with the supplier for the demonstration and assessment of requirements and standard, including safety, with regulators focusing on recall of products and legal prosecution of suppliers who don’t comply. The ‘CE’ marking used on consumer products is an example of such suppler ‘self-certification’. Such an environment changes significantly the behaviours of suppliers from those in the more active regulator environment. Instead decisions regarding safety and production of safety evidence will often be treated as commercial decisions based on trade-offs between benefits and business risk (Docker, 2011).

The Contract is Important!

The focus of this thesis is on military aviation systems, and as such the role of the contract in communicating and enforcing certification requirements is important. The contract must communicate certification requirements, provide incentives for suppliers to comply and provide mechanisms for enforcement when suppliers don’t comply. Section 2.3 will examine the way military contract authorities communicate and manage enforcement of certification requirements.

2.3 Communication and Enforcement of Certification Requirements using Contracts

In Section 2.2 the importance of the contract was highlighted for the military airworthiness certification environment. To understand how certification requirements are communicated and enforced by the contract, it is important to understand the constraints on contracts for military programs. The following sub-sections describe the principal constraints, and provide background on why it is important that an assurance framework can work within these constraints.

2.3.1 Government Preference for Fixed Price Contracts

(Defense Contract Management Agency, 2012) summarises that there are numerous different contracting paradigms, which can be generally categorised as follows[2]:

- fixed-price contracts (fixed price, fixed price with economic adjustment, fixed price incentive),
- cost-plus contracts (cost plus award, cost plus fixed fee, cost plus incentive),
- time and materials contracts,
- performance or outcome based frameworks,
- cost and schedule risk sharing arrangements such as accords, alliances, and cooperative agreements, etc.

Contracts may also incorporate elements of several of these paradigms into one single contract. For example, the performance-based paradigm may apply to those elements of a contracting specifying service delivery requirements, whereas compliance with airworthiness requirements and product delivery is often achieved using fixed-price contract arrangements.

<u>Reasons for Fixed-price Contract Preference</u>

(Commonwealth of Australia, 2012) defines a fixed-price contract as *"a contract in which the price remains unchanged for the period of the contract except for agreed contract scope changes or variations in escalation and exchange rates if applicable."* The fixed-price contract relies on the premise that a supplier is able to estimate the cost of producing and supplying the goods or services with reasonable accuracy.

[2] The differences often relate to how the acquirer wishes to incentivise the supplier's behaviours or manage contract risk.

(Defense Contract Management Agency, 2012) states that fixed-price contracts are "*preferred to all others because it encourages the contractor to contain costs.*" The Australian Defence Material Organisation also prefers fixed-price contracts (Defence Materiel Organisation, 2010), (Commonwealth of Australia, 2012), as do the UK Ministry of Defence (MoD) (Think Defence, 2010). The preference by military acquisition organisations for fixed-price contracts exists due to the following factors:

- Government funding approvals tend to favour firm costing estimates, due to the fixed periods of political budgeting.
- Fixed-price contracts facilitate straightforward project budgeting arrangements and cost management due to their cost certainty.
- Fixed-price contracts transfer perceived cost and schedule management responsibility to the contractor.
- Fixed-price contracts appear to reduce the opportunity for scope creep on the supplier and acquirer sides, offering Governments greater assurance that budgets are being used appropriately.

Difficulties of Fixed-price Contracts

Despite the benefits of fixed-price contracts, this type of contract is not without its difficulties. When it comes to the acquisition of largely non-developmental aircraft or aircraft systems, where reasonable costs can be established at the outset, fixed-price contracts are generally suitable. Highly developmental systems are less cost effective to contract for under fixed priced arrangements, as the unknowns affecting the developmental aspects usually translate into significant cost and schedule risk margins appearing in supplier cost estimates. Open competition and fixed price bidding also encourage supplier under-bidding (Think Defence, 2010). When a contact is under-bid, cost and time overruns are common, and delivery is only achieved by de-specifying, delaying and reducing quantities; all of which undermine the intended capability outcome. From the safety perspective, the result may be that the responsible authority has to retain undesirable risks, because they can't be treated within the resources available.

In cases where these problems are prevalent, cost plus, risk sharing arrangements such as accords or alliances, or just a really well managed time and materials contract can be more cost effective in the long run. However, there are still numerous examples of developmental systems being acquired under fixed price arrangements for the ADF, US DoD and UK MoD.

Working Within the Fixed-price Paradigm

A value for money and on-time/on-budget fixed price contract will only be possible when both the acquirer's and supplier's expectations resulting from their 'world views' are aligned. This in turn implies that they undertake a series of actions to align their expectations of the product and evidence requirements prior to contract signature. The better the supplier understands the requirements before contract signature, and the better they understand how shortfalls in product and evidence are to be resolved within the contract, the better the likelihood of a favourable contractual outcome. A favourable contractual outcome is generally a pre-requisite for a favourable capability and safety outcome also.

In the literature referenced in Section 2.3, 2.5, and 2.6, the constraints of fixed-priced contracting arrangements on achieving safety assurance for military systems have largely been ignored. Hence, many of the limitations of contemporary practices for safety assurance discussed in Section 2.6 may be a bi-product of the lack of recognition of the role the contracting paradigm.

For the purposes of relevance to the Australian Defence acquisition environment and the types of contracts preferred, this thesis focuses on fixed price contract arrangements. There may also be read across to other contract paradigms.

2.3.2 Evidence Delivery or Access

Evidence produced by suppliers under a contract can typically be classified as either deliverable or non-deliverable. Deliverable evidence is supplied to the acquirer by the supplier, whereas non-deliverable evidence will tend to be held at the supplier facility, or their sub-contractors. Contracts provide acquirers the ability to assess evidence in one of two ways, depending on if the evidence is deliverable or non-deliverable. Delivery versus access to evidence is usually dictated by intellectual property considerations.

Deliverable Evidence

Access for assessors to deliverable evidence is usually straightforward, as is typically stipulated through the Statement of Requirement (SOR), Statement of Work (SOW) and Contract Data Requirements List (CDRL) which references applicable Data Item Descriptors (DIDs) for each piece of evidence the acquirer and certifying authority require to conduct acceptance and certification.

Non-deliverable Evidence

For non-deliverable evidence, assessor access may be achieved through on-site access provisions. If an element of non-deliverable evidence is not stipulated through the contract, or through a supplier plan for which acquirer approval is required under the contract, then it is unlikely that the supplier will make this evidence available. In the event that there are evidence shortfalls, and the contract does not cater for this circumstance, the acquirer may be forced to seek an amendment to the contract, which will usually incur an associated cost and schedule impact.

Limitations with Data Item Descriptors

For deliverable evidence, although DIDs define the structural content requirements for evidence, they do not necessarily define the quality of the information that underlies the required content (e.g. forms of argument, defensibility of the argument, quality of evidence, etc.). An acquirer review and acceptance cycle is usually the means of assuring the quality of the content of artefacts delivered against DIDs. The acceptance and rejection criteria are often constrained by the activities described in acquirer approved plans from earlier in the lifecycle. This illustrates that it is vital that the supplier plans are meaningful to the goals for safety assurance, and also that the appropriate assessors to review the deliverables. Both are difficult propositions for projects (Docker, 2011), (Kinnersly, 2011), if they are not properly coordinated with the certifying authority.

On this basis, it is possible to infer that contract DIDs need to be accompanied by material on the quality of argument and evidence required to comply with software system safety assurance objectives. This thesis also examines the requirements for integration between assurance requirements and contracting mechanisms for evidence delivery to determine criteria for assurance paradigms to enable successful integration with contracts.

2.4 Current Approaches for Safety-related Systems

Let's now examine the certification of safety-related systems including the standards employed across the civil and military aviation domains and draw comparisons between them. Table 3 proves a summarised list of the standards related to software and safety-critical systems. Note that the columns used to group the standard do not necessarily imply that all listed standards are applied to a development in that domain.

	Military			Civil Aviation	Other
	US DoD	UK MoD	Other		
System Safety	MIL-STD-882C/D	DEF STAN 00-56	DEF (AUST) 5679	AC/AMC 25.1309 SAE ARP 4754/A SAE ARP 4761	IEC61508 Part 1 (Funct. Safety EEPE) ISO 26262 (Road Vehicles) EN 50126 (Rail) ISO15026 (IT)
Software Safety	JSSSC SSSH (Guidebook)	DEF STAN 00-58 (Obsolete) DEF STAN 00-56 SSEI-TR-0000041[3]	DEF (AUST) 5679 H ProgSäkE (Handbook)	CAP670 SW01 (Air Traffic Control)	IEEE 1228 IEC61508 Parts 3 and 7 (Functional Safety EEPE) NASA-STD-8719.13 (Space)
Software Assurance	No dedicated standard[4]	DEF STAN 00-55 (Obsolete)[5] SSEI-TR-0000041	DEF (AUST) 5679[6] H ProgSäkE (Handbook)	RTCA/DO-178B/C DO-248B/C DO-278A (CNS/ATM) FAA Order 8110.49 Job Aid	IEC61508 Parts 3 & 7 (Funct. Safety EEPE) ISO 26262 (Road Vehicles) ISO15026 (IT) EN50128 (Rail)
Software Development	DoD-Std-2167A MIL-STD-498	No dedicated standard	No dedicated standard	DO-330 (Tools) DO-331 (Modelling) DO-332 (Object Oriented), DO-333 (Formal Methods)	J-STD-016 IEEE12207 (Software Life Cycle IEEE829 (Test Docs) IEEE830 (Req Specs) IEEE1012 (V&V) IEEE1028 (Reviews) IEEE1042 (CM) IEEE1044 (SPRs)

Table 3: Standards Pertaining to Safety and Software Assurance

The approaches adopted by these standards can be broadly classified as one of three approaches:

- The Assurance Level Approach (prescriptive)
- The Evidence Assurance Level Approach (semi-prescriptive)
- The Safety Argument Approach (goal-based)

The following sub-sections discuss each of these approaches.

[3] The Software Systems Engineering Initiative (SSEI) developed a technical report on Software Safety Evidence Selection and Assurance for guidance for compliance with Defence Standard 00-56.

[4] MIL-HDBK-516B references RTCA/DO-178B, but is often not used by suppliers to the US military.

[5] Defence Standard 00-55 was made obsolete by the issue of Defence Standard 00-56 Issue 4. Defence Standard 00-55 is currently undergoing redevelopment.

[6] DEF (AUST) 5679 is predominantly used by the Royal Australian Navy. The Royal Australian Air Force tends to use the civil aviation standards where possible.

2.4.1 The Assurance Level Approach

The assurance level approach has historically been the most widespread way of providing assurance of software for safety-related systems. The term assurance level is a generic label for those standards employing a predominantly development process based assurance level framework. Examples of their specific labels and the standards from which they are derived are shown in Table 4.

Software Level	**Levels**$^{\#\%}$	**Source**
Safety Integrity Level (SIL)[7]	(0) 1-4	IEC61508 Edition 1 (IEC, 1998) IEC561508 Edition 2 (IEC, 2010) Defence Standard 00-55 Issue 2 (Ministry of Defence, 1997) ISO15026 (ISO/IEC, 1998) EN50128 (CENELEC, 2001)
Automotive SIL (ASIL)	A-D	ISO26262 (ISO, 2011)
Design Assurance Level (DAL)[8]	(E) D-A	RTCA/DO-178B (RTCA Inc., 1992) RTCA/DO-178C (RTCA Inc., 2011)
Safety Assurance Level (SAL)	(S_0) S_1-S_6	Def (Aust) 5679 Issue 2[9] (Australian Department of Defence, 2006)
Software Hazard Risk Index (SHRI)	5-1	MIL-STD-882C (US DoD, 1993)
# Levels shown in brackets indicates the lowest level for which the standard defines no requirements. % Levels are presented from least assurance to most assurance.		

Table 4: Examples of Software Level Approaches

The following sub-sections consider the assignment and application of assurance levels.

Assignment of an Assurance Level

There are two factors that must be considered in the assignment of an assurance level:

- what the assurance level is being assigned to, and
- how the assignment is performed.

Depending on the specific standard, assurance levels may be assigned to either a safety function (or the safety requirements associated with a safety function), or to a configuration item. The former allows the assurance level to be fully contextualized by

[7] Note that the SILs used in Defence Standard 00-55, IEC 61508 and ISO 15026 are not equivalent.

[8] ARP4754A introduces the concept of the Functional DAL (FDAL) and Item DAL (IDAL), although strictly speaking, assurance is only applied to IDALs in the way it was formerly applied to DALs under ARP4754, with FDALs used to model DAL assignment/reduction for functions and architecture.

[9] Note that SALs were previously defined as Safety Integrity Levels SILs in Def (Aust) 5679 Issue 1 with levels S_1-S_6

the importance of the safety function or safety requirement for a specific system. The latter allows visibility of the safety importance of the specific configuration item for achieving its allocated functions. The latter also suggests easier portability and re-use of assurance evidence as the evidence is traceable to a specific configuration item, and not a safety function contextualized by a specific system implementation. However system specific context is often unavoidable and thus portability is rarely straightforward.

Because faults in software are systematic (Weaver, 2003), and thus can't be characterised directly by random process models, assurance level assignment can't be based on probability. Thus the traditional dimensions of risk (i.e. consequence and probability) can't be used directly for assignment of an assurance level. Most standards recognise this, and thus have developed alternative ways for assigning assurance levels that don't involve establishing probabilities directly. It should also be observed that development of software to an assurance level does not imply the assignment of a failure rate for that software. Thus, assurance levels or reliability targets based on assurance levels cannot be used by the system safety process as hardware failure rates are (RTCA Inc., 1992).

There are two most dominant approaches for assigning assurance levels. The most widely used is to assign the assurance level proportional to the severity or consequence of the failure condition, hazard or accident of either the physical item or of the function it implements. A related approach is to assign the level based upon the acceptable probability of failure of the function. In this second case, it should be noted that the acceptable probability of failure of the function is established proportionally to the severity or consequence of the failure of that function, and thus in many respects it mirrors the severity proportional approach.

One less widely used approach (ISO/IEC, 1998) proposes that the assurance level should be proportional to risk, being a function of both consequence and frequency. However no guidance is provided on how the frequency of software failure is determined, noting the limitations with this approach discussed earlier. A further variation on this approach is to assign the assurance level based on the proportional combination of the severity/consequence and a control category established from the degree of autonomous control the software has over the hardware function (US DoD, 1993).

Some standards permit a reduction of the assurance level (usually only one level) based on architectural mitigations or mitigation external to the system. Others limit the claimed assurance level based on architectural configuration. Table 5 summarises the approaches adopted by the different standards.

Standard	**Level Assigned to**	**Assignment Methodology**	**Level Reduction / Claim Limits**
IEC61508	Safety Function	Average Probability of Failure (Functional Failure)	Architectural (Claim Limit – Safe Failure Fraction and Hardware Fault Tolerance)
Def Stan 00-55 Iss 2	Configuration Item	Severity Proportional (Accident)	Architectural
ISO15026	Configuration Item	Risk Proportional (Threat)	Architectural
EN50128	Safety Function	Tolerable Hazard Rate	Architectural
ISO 26262	Safety Function / Hazard	Severity, Controllability, Exposure Time Proportional	Architectural
RTCA/DO-178B/C	Configuration Item	Severity Proportional (Functional Failure)	Architectural
Def (Aust) 5679 Issue 1 & 2	Safety Requirement	Severity Proportional (Accident)	External Accident Mitigation
MIL-STD-882C	Configuration Item	Severity (Hazard) and Control Category Proportional	Control Category

Table 5: Assurance Level Assignment Approaches

Application of Assurance Levels

Based on the assigned assurance level most of the standards present in tabular form, or equivalent, mandated or recommended lists of activities, processes and methods/techniques that should be applied at an applicable phase of the development lifecycle or process. Some standards also include architectural design feature prescriptions (e.g. IEC 61508 (IEC, 2010)). The more rigorous the assurance level, the greater the number or more thorough the prescription of activities, processes and methods/techniques. Some standards also allow for flexibility by accepting alternative techniques or methods provided there is justification that they are as effective as those they are replacing.

A variation on this approach involves the prescription of objectives of the software lifecycle (e.g. RTCA/DO-178B (RTCA Inc., 1992), rather than activity, technique or method prescriptions. This approach offers greater flexibility to the developer on selection of activities, processes and methods/techniques, but they are still bound by certification authority approval and the software lifecycle processes to which the objectives apply. (Kelly, 2008) argues that this approach is still synonymous to the activity, technique and method prescriptions which characterise the prescriptive approach because supporting the objectives are descriptions of activities and design considerations for achieving those objectives and descriptions of the evidence that indicates objective satisfaction. (McDermid & Rae, 2012) emphasise though that only two of the objectives in (RTCA Inc., 1992) stray into direct prescription, and that the perception of it being a process standard may be because there are so many objectives.

At the level of techniques, activities and methods there are some commonalities between the standards, but also some significant differences. For example, Defence Standard 00-55 (Ministry of Defence, 1997) and Def(Aust) 5679 (Australian Department of Defence, 2006) place significant emphasis on formal methods in the demonstration that requirements are consistent with each other, and that the requirements translate correctly to implementation. In addition, Defence Standard 00-55 (Ministry of Defence, 1997) placed significant emphasis on static code analysis. IEC61508 (IEC, 2010), on the other hand, identifies a very large range of techniques and methods, to almost encyclopaedic proportions, including formal methods. In significant contrast, RTCA/DO178B (RTCA Inc., 1992) implicitly stresses human centric reviews and rigorous testing to assure that requirements are adequately specified and that they translate correctly into implementation (McDermid & Kelly, 2006), (McDermid, 2001). Note though, that RTCA/DO-178C (RTCA Inc., 2011) has now been supplemented with RTCA/DO-333 (RTCA Inc., 2011) which provides guidance on the application of formal methods within aviation software developments.

(McDermid, 2001) points out that the rationale for recommending or prescribing development processes and methods is complex, but is based upon two key assumptions:

- the processes for higher assurance levels produce apparently “better” software; and
- the processes for the higher assurance levels are more expensive, hence it is inappropriate to use them unless the consequences of failure are severe.

The inductive argument (and presumption) is that the more rigorous the activities, processes and methods/techniques, the greater the assurance that the software does not contain errors, and can be relied on to function safely within its operating context.

Benefits of the Assurance Level Approach

The assurance level approach is widely used, and thus there must be perceived benefits. A review of relevant literature and examination of industrial practice reveals the following benefits:

- **Track Record.** Despite a relatively small number of high-profile accidents being attributed to software, software developed using this approach has a remarkably good track record (McDermid & Pumfrey, 2001). Albeit, it is not possible to conclude that the good track record resulted from the assurance level approach, or was achieved due to other factors.
- **Process to Product Paradigm.** There are circumstances under which the assurance level approach could be considered to satisfy the core requirements of a product-based approach, provided it can be shown that the requirements subject to such prescriptions include all safety requirements (Kelly, 2008).
- **Minimal Subjectivity and Variability.** The prescription of processes sets clear expectations for the supplier's scope of work and evidence delivery (Kelly, 2008), leading to consistent understanding between suppliers and acquirers. The prescription of processes sets expectations for a minimum baseline which should deter those suppliers who don't have the requisite organisation, people and processes and tools (Kelly, 2008). The use of a defined process reduces variability; thus also improving planning and costing estimates for both suppliers and assessors (Kelly, 2008).
- **Trustworthy Evidence.** They provide guidance on how to develop and implement requirements in a trustworthy manner (Kelly, 2008).
- **Certifying Authority Compatible.** Certification authorities have developed entire certification frameworks and the associated environments around standards adopting the assurance level approach, including whole of product standards (refer Section 2.2.4), related standards, assessor delegations and authorisations, assessor guidance and training material.

Common themes that emerge are with respect to the reduction in subjectivity and variability between suppliers, acquirers and certifying authorities; and the confidence established from trustworthy evidence. Both of these benefits are relevant to the military certification environment. Therefore, it is desirable that improvements in safety assurance frameworks strive to preserve such benefits. This is examined further in Section 2.7.

Limitations of the Assurance Level Approach

However despite the benefits listed above, there are a significant number of criticisms of the process-based assurance approaches that use assurance levels. These are as follows:

- **Realised Risks.** Avoidable software failures have occurred where the assurance level approaches have been applied, leading to loss of life and for major economic losses (Committee on Certifiably Dependable Software Systems, 2007), (Marks, 2008). The good track record is often disputed by the difficulty of attributing accident causal factors to software (McDermid & Pumfrey, 2001).
- **Process isn't Product.** The assurance level approaches have concentrated on the process aspects, and product aspects are mostly implicit (Lindsay & McDermid, 1997). This has led to the certification of software systems using the assurance level approach to rely more on assessments of the process used to develop the system rather than on the properties of the system itself (Committee on Certifiably Dependable Software Systems, 2007). There is a lack of evidence that adherence to the prescribed process leads to a specific level of integrity (Lindsay & McDermid, 1997), (Redmill, 2000). There is also a lack of evidence that software of differing levels does have failure rates of "integrity level order" (McDermid, 2001), due in part to poor correlation between the techniques and methods prescribed and the failure rate implicitly defined by the assurance level (Kelly, 2008). The assurance level is also often wrongly interpreted as achieving the target rate of dangerous failures of the product (Redmill, 2000). Finally, the product argument inherent behind the assurance level approach is not explicit.
- **Questionable Level Definition.** Specific assumptions underlying assurance level definition are questionable (McDermid, 2001). Assurance level definitions differ between standards and are derived in different ways; thus making it difficult to transfer assurance levels from one standard domain to another (Redmill, 2000). Some methods for assigning assurance levels (e.g. using control categories) assume the risk of software faults leading to an accident is decreased by giving the human more control; but this is incompatible with the (US DoD, 1993) design order of precedence (Lindsay & McDermid, 1997).
- **Differences in Methods.** Some standards overemphasise testing and human reviews as verification methods (Kelly, 2008). Other standards prescribe formal methods, which may be ineffective for some safety-critical control systems, where it is necessary to assess control stability, jitter, timing, etc. of discrete

approximations to continuous control problems (McDermid & Rae, 2012). Compliance is not easily portable from one domain to another due to differences in the requirements (methods, techniques, processed, documentation) between standards (Redmill, 2000), (Kelly, 2008).

- **Questionable Value for Money.** Some prescribed methods do not provide a material contribution to safety in certain circumstances, and thus maybe wasteful in terms of cost and schedule for projects (McDermid & Pumfrey, 2001).
- **Inflexible.** The prescription of processes can hinder the adoption of new process approaches that could improve flexibility and predictability of software development (Redmill, 2000), (Kelly, 2008). The prescription inhibits freedom to choose arguments and evidence that address the specific circumstances of the software safety requirements (Kelly, 2008).
- **Used Out of Context.** The assurance level is contextualised by the safety assessment that assigned it, and the safety requirements applicable to the specific system. The assurance aspects of these approaches are sometimes applied independent of the system safety approaches to which they are dependent, leading to confusion of the term (Redmill, 2000).

While the list of criticisms is certainty long, several key themes are evident. Most significant is the limitations in an explicit product behavioural focus with respect to safety. Of similar importance also are the somewhat arbitrary prescription of techniques and methods that do not have clear rationale for their risk reducing role. In terms of practical implementation in the military certification environment, assurance level approaches do not include a means for assessing the impact on safety risk when there is a shortfall in evidence against one of the requirements of the standard. While the civil certification frameworks tend to take a black and white compliance/non-compliance view, and this somewhat avoids the problem, military programs and their associated cost, schedule and capability constraints mean that establishing the risk in such cases is vital. However the assurance level approach is a target and does not address the implications for shortfalls with respect to risk. Therefore, it is desirable that improvements in assurance frameworks strive to avoid such limitations. This is examined further in Section 2.7.

2.4.2 The Evidence Assurance Level Approach

In recent years several standards and literature have emerged that are based on the concept of evidence assurance rather than assurance levels. The evidence assurance level approach sets benchmarks for the suitability and sufficiency of evidence used to achieve compliance with safety requirements or safety goals. The term evidence assurance level is a generic label for those approaches employing this concept. Examples of their specific labels and the standards from which they are derived are shown in Table 6.

Evidence Assurance Level	**Levels#%**	**Source**
Assurance Evidence Level (AEL)	1-5	CAP670 (Civil Aviation Authority, 2003)
Evaluation Assurance Level (EAL)	1-7	ISO/IEC 15408 (ISO/IEC, 2009)
Safety Evidence Assurance Level (SEAL)[10]	1-4	(Fenn & Jepson, 2005)
# Levels shown in brackets indicates the lowest level for which the standard defines no requirements. % Levels are presented from least assurance to most assurance.		

Table 6: Examples of Evidence Assurance Level Approaches

The following sub-sections consider both the assignment of evidence assurance levels and the application of the evidence assurance levels.

Assignment of the Evidence Assurance Level

As for the assurance level, there are two factors that must be considered in the assignment of a evidence assurance level:

- what the evidence assurance level is being assigned to, and
- how the assignment is performed.

Depending on the specific example, evidence assurance levels can either be assigned to safety objectives or configuration items. The former allows the evidence assurance level to be fully contextualized by the importance of the safety objectives or safety requirement it pertains to for a specific system. In the latter case, this implies that the evidence assurance level applies to the collective set of the safety requirements applicable to the configuration item. As in the assurance level case, the latter also suggests easier portability and re-use of assurance evidence as the evidence is traceable

[10] Note that the SEAL referred to here is not the implementation of the SEAL for the F-35 JSF Program (Eccles, 2007), where, despite intent, the SEAL is more akin to the software level assurance paradigm, rather than the evidence assurance paradigm.

to a specific configuration item, and not a safety objective contextualized by a specific system implementation. However since the evidence is with respect to safety objectives of the configuration item, portability of assigned levels is rarely straightforward. However the evidence assurance level approach does offer the advantage that the evidence may be portable for assessment against differing safety objectives in another system context.

There are two most dominant approaches for assigning evidence assurance levels. There is a severity proportional method similar to that used for assigning assurance levels, and there is a combined severity proportional and failure probability of mitigating factors approach. The only difference is that the latter permits consideration of mitigating factors outside the context of the specific safety objective. Where assurance levels are assigned to low level safety objectives (or claims if applied in conjunction with an argument structure – refer Section 2.4.3) then some approaches suggest refactoring methods for reducing evidence assurance levels based on the way safety sub-objectives combine to achieve the overall safety objective.

Some approaches permit a reduction of the evidence assurance level based on the number and strength of defensive layers or other architectural mitigations external to the system. Table 7 summarised the different approaches.

Source	**Level Assigned to**	**Assignment Methodology**	**Level Reduction / Claim Limits**
CAP670	Configuration Item	Severity Proportional	Number and strength of defensive layers
ISO/IEC 15408 (Security)	Configuration Item	Severity Proportional	Nil
SEAL	Each Safety Objective (or Goals if applied to a Safety Argument)	Top Goal: Severity and Failure Probability of Mitigating Factors Proportional Sub-goals: Relevance, Coverage and Trustworthiness Proportional	Layered Protection or Defence In Depth

Table 7: Evidence Assurance Level Assignment Approaches

<u>Application of the Evidence Assurance Level</u>

Based on the assigned evidence assurance level most of these approaches present in tabular form, or equivalent, mandated or recommended lists of evidence types that should be provided in support of the safety objective. The philosophy is that the developer should not be required to follow any particular process or use any particular method or technology, provided evidence is produced in support of the safety objective and the production of evidence is sufficiently rigorous. Evidence assurance level approaches are sometimes proposed to be used in conjunction with the safety argument approach which will be summarised in Section 2.4.3.

Evidence assurance level approaches typically present categories of evidence contextualised by attributes of the product that they relate to. For example, CAP670 (Civil Aviation Authority, 2003) identifies three general evidence types: analytic evidence, test evidence and field experience across attributes of the product including functional properties, timing properties, robustness, reliability, accuracy, resource usage and overload tolerance. Other approaches vary the degree of formality in the production of evidence types (ISO/IEC, 2009). The more onerous the evidence assurance level, the more diversity and formality required in the provision of evidence.

Benefits and Limitations of the Evidence Assurance Level Approaches

The application of evidence assurance level approaches is not yet widespread, and is constrained to several limited domains, as illustrated by Section 2.4.2. Therefore, there is limited literature discussing the specific benefits and limitations of such approaches. Evidence assurance level approaches appear to have been developed to bring some of the benefits from the assurance level approach into the product-focused assurance paradigm. The approach provides a more explicit product focus than the assurance level approach by setting product safety requirements and providing a framework for determining evidence to support the safety requirements. The focus moves away from development lifecycles, and specific techniques and methods. However, in practice the application of evidence assurance levels has been found to be difficult (Weaver, 2003). This is predominantly due to:

- Evidence assurance levels tend to apply to configuration items, or at best specific requirements assigned to configuration items. The measures of evidence set by the evidence assurance level approach are not based upon the type of requirement and the failure modes or hazards to which it relates (Weaver, 2003).
- The focus is on setting rules for the types of evidence that should be provided, but not on how the evidence is combined. Limited elaboration is provided by these methods for how the evidence combines to provide a strong case for safety (Weaver, 2003).

- Limitations in the guidance for the selection of evidence to overcome the aforementioned limitations (Weaver, 2003).
- Software components assume the more rigorous evidence assurance level equivalent to the highest level of the individual software safety requirements implemented in the software component. This can lead to depth of evidence and evidence types being recommended at a component level without adequate consideration of the specific software hazardous failure modes that the requirements address.

The key themes evident with respect to evidence assurance levels are that evidence needs to be appropriate to the requirements and failure modes to which it relates, and that the properties established from the way differing evidence combines is more important than simply the type of evidence presented. Therefore, it is desirable that improvements in assurance frameworks strive to incorporate such factors relating to evidence. This is examined further in Section 2.7.

2.4.3 The Safety Argument and Evidence Approach

Most contemporary of the approaches summarised within this thesis is the safety argument and evidence approach to software safety assurance. The approach has come to provenance as a means of addressing some of the limitations with the assurance level approach (Weaver, 2003). The safety argument and evidence approach adopts a product-centric perspective for assurance (Kelly, 2008). In this paradigm, arguments are required to:

- justify the determination and adequacy of product behavioural safety objectives derived from hazard analysis;
- present the case for the satisfaction of the product behavioural safety objectives based on relevant and trustworthy evidence;
- justify the selection of evidence used for specific claims within the case;
- justify why the presence of counter evidence does not undermine the case for safety; and
- state why the case supports an acceptable level of safety in the identified usage context.

The argument and evidence required to justify safety collectively form the safety case, which is often summarised in a safety case report (Kelly, 1998).

Safety case arguments are rarely provable deductive arguments (Kelly, 2008). Because they essentially catalogue a set of beliefs about the evidence, by their nature, they are often subjective. The arguments are mostly inductive and carry with them a degree of uncertainty as to their truth. On this basis, (Kelly, 2008) indicates that the objective of safety case development is not only the presentation of the subjective case, but also the process of obtaining mutual agreement between the supplier and certifying authority of the validity of the subjective case. The acceptance of the case is a social process (Bloomfield & Bishop, 2010).

There are several different types of arguments that are commonly included within safety arguments (Kelly, 1998). Central to the thrust of the safety arguments and evidence approach is the product safety argument, sometimes referred to as the direct argument. This argument should focus on product behaviours and the validity and satisfaction of safety objectives. Supporting the direct argument should be backing arguments, which provide additional information that informs confidence about the direct argument and the evidence used to support the direct argument.

The safety argument and evidence approach is generic, and thus can be applied to all elements of a system, including software. Software is best dealt with within this paradigm by ensuring that the focus of the software-related arguments (direct claims) is on demonstrating the product safety resulting from software product behaviours, rather than demonstrating the development of the software to any specific process. The safety argument and evidence should address both normal operating and failure circumstances of the software and the system, and their mutual behavioural interactions. Arguments and evidence about the development process are still important, but they have a different role in the safety argument. They are used in supporting subordinate and backing claims about the trustworthiness of evidence used for supporting product arguments.

Software Product Arguments

(University of York, 2004) summarises several historic safety case patterns for software aspects. Some example patterns include arguments over:

- product and process,
- hazards and safety requirements,
- functional versus non-functional properties.

However, there are problems with these approaches. The main problem stems from these arguments creating a discontinuity in the safety case (Weaver, 2003). This is because the safety argument has a product focus, and the software argument historically had a process focus. In many respects, this is unsurprising because historic arguments originated from the principles underlying the assurance level approaches described in Section 2.4.1. Furthermore, software cannot be safe or unsafe, and the safety of software can only be judged in the system context (Leveson, 1995). Hence the software safety argument cannot be disjoint of the system safety argument. Because of this problem, more product oriented patterns have emerged for dealing with software arguments in the safety case, such as those described by (Weaver, 2003) and those inferred by (Civil Aviation Authority, 2003).

Examining (Weaver, 2003)'s patterns provides a good illustration of the principles important to safety arguments for software. (Weaver, 2003) proposes arguments for showing that causes of hazardous software failure mode are either absent or are detected and handled. Absence arguments are relevant to the software component under consideration whereas detection and handling arguments may be relevant to both the software component under consideration and other elements of the system. The framework is based upon (Pumfrey, 1999)'s taxonomy of software failures, which is based around the provision of services.

Service

The communication of a piece of information, with a specific value, at a particular time.

(Pumfrey, 1999)

There are five categories of software failure in (Pumfrey, 1999)'s taxonomy, as follows:

- *Omission*: the service is never delivered;
- *Commission*: the service is provided when it is not required;
- *Early*: the service occurs earlier than intended (either absolute real time, or relative to some other action);
- *Late*: the service occurs later than intended (either absolute real time, or relative to some other action); and
- *Value*: the information (data) delivered has the wrong value.

(Pumfrey, 1999)'s classification was chosen by (Weaver, 2003) over other classifications, as it addresses some of the discrepancies between other such classifications (i.e. those defined by (Ezhilchelvan & Shrivastava, 1989) and

(Bondavalli & Simoncini, 1990)). It is also consistent with (Avizienis, et al., 2004)'s taxonomy of service failure modes, and the definitions therein provide some consensus on the topic.

(Weaver, 2003) (and also (Civil Aviation Authority, 2003)) identify that the three principal types of evidence needed within the safety argument for hazard directed requirements are as follows:

- *Requirements Validation*: Evidence that the behaviour specified by the requirement is complete and accurate (e.g. real world operation, simulation or analytic evidence that confirms the appropriateness of the behaviour under relevant operating and failure conditions).
- *Requirements Satisfaction*: Evidence that the behaviour specified by the requirement is achieved by the system and software implementation (e.g. a combination of analytic and test based verification evidence, potentially supplemented with field service experience).
- *Requirements Traceability*: Evidence that the hazard directed safety requirement has been decomposed or refined through the system design into the system and software implementation (e.g. using matrices or methods as described by (Palmer, 1997) or (Praxis Critical Systems, 2001)).

(Weaver, 2003) also defines a categorisation of evidence at the system requirements level, software requirements level, and software functional unit level to assure that evidence is appropriately contextualised. The reader is directed to (Weaver, 2003) for further information on the framework.

Safety Argument Assurance

Because it is possible to build both strong and weak safety arguments, including those for software, approaches to safety argument assurance have been proposed by both (Weaver, et al., 2003) and (Fenn & Jepson, 2005). Both these approaches define qualitative levels of argument assurance and means for factoring the levels across the argument. (Weaver, et al., 2003) reasons that by expressing the assurance of an argument it is possible to identify what confidence can be placed in that argument. The safety argument assurance is intended to assist in development of the safety case and in assessor review by making explicit the confidence in the safety argument and the evidence presented.

(Weaver, et al., 2003) proposes a qualitative approach defining Safety Assurance Levels (SAL) and states that the SAL is *"the level of confidence that a safety argument element (GSN goal or solution) meets its objective"*. There are four SALs in the framework (SAL 1 lowest confidence – SAL 4 highest confidence). (Weaver, 2003) describes a top level process for applying SALs as follows:

- Determine the top level SAL to set the target assurance of the argument.
- Analyse the decomposition of the argument and re-factor the argument as necessary to fit a single support pattern, linked support pattern or convergent support pattern.
- Determine SAL decomposition across child elements using the SAL decomposition tables relevant to the support pattern type:
    - Single Support – direct assignment,
    - Linked Support – assignment based on relevance,
    - Convergent Support – assignment based on independence.
- Determine SALs for evidence.

(Fenn & Jepson, 2005) propose an alternative approach using SEALs, for which the evidence assurance aspects were introduced in Section 2.4.2. SEALs attempt to overcome the difficulties in practice of decomposing safety arguments into the support patters described by (Weaver, et al., 2003). The SEAL is *"a qualitative statement of requirement for a degree of confidence in the evidence that a specific safety goal has been achieved"* (Fenn & Jepson, 2005). As for (Weaver, et al., 2003)'s SALs, SEALs are decomposed from the top level goal across child elements, although the rules proposed for decomposition differ somewhat from SALs. Decomposition strategies are suggested based on the adequacy of evidence (inadequate, adequate, more than adequate) and taking into account apportionment strategies, independence and claims related to direct versus backing evidence (Fenn & Jepson, 2005).

Separation of the Safety and Confidence Arguments

(Kelly, 2008) and (Hawkins, et al., 2011) both identify that failure to recognise the differences in role between the product-based elements and the process assurance-based elements of the safety argument can lead to safety arguments being difficult to interpret. In practice, this failure also leads to safety cases that lack sufficient product focus and that are not compelling to assessors. To address this, (Hawkins, et al., 2011) introduces the concept of *assured safety arguments*. An assured safety argument provides a structure for arguing safety in which the product safety argument is accompanied by a confidence argument that documents *"the confidence in the structure and bases of the product safety argument."*

(Hawkins, et al., 2011) propose that any product safety argument includes assertions related to the sufficiency and appropriateness of the inferences declared in the argument, the context and assumptions used, and the evidence cited. Therefore, to be compelling the safety argument should justify the confidence in the assertions made, and the degree of uncertainty in the truth.

(Hawkins, et al., 2011) describes that the confidence arguments should be tied to three types of Assurance Claim Points (ACP), which correspond to the three different types of assertions in safety arguments, as follows:

- *Asserted Inferences* – the link between the parent claim and its strategy or sub-claims; the confidence argument should document why the assessor should believe that the premises are sufficient to establish the probable truth of the conclusion.
- *Asserted Context* – the link between contextual information (represented by context or assumption elements) and the argument elements to which it applies; the confidence argument should document why the assessor should believe that the asserted context is appropriate and trustworthy.
- *Asserted Solution* – the link between a solution and the argument; the confidence argument should document why the assessor should believe that the evidence is appropriate to support the claim, and the evidence is trustworthy.

Each confidence argument should propose that the probable truth of the assertion is believable, residual uncertainties in the assertion have been identified, and residual uncertainties in the assertion are insufficient to undermine the probable truth (Hawkins, et al., 2011).

(Hawkins, et al., 2011) also propose that the individual fragments of confidence arguments applicable to each assertion across the product safety argument, should also be assembled together into an overall confidence argument. The proposal is that the overall confidence argument requires that all assertions of the safety argument have an accompanying confidence sub-argument that argues confidence for all inferences, all context and all evidence used in the safety argument (Hawkins, et al., 2011). (Hawkins, et al., 2011) also identify some concerns in relation to the overall confidence argument. Arguing the sufficiency of the overall confidence in the safety argument is probably more complex than the simple composition of arguments of sufficient confidence for each argument assertion. This is because it is necessary to examine whether the multiple branches of argument in the safety argument share common underlying shortfalls in confidence or believability.

(Hawkins, et al., 2011) claim that by exercising discipline over the permissible claims and evidence of the safety argument, and encouraging a systematic approach to the construction of a confidence argument, the suitability and sufficiency of the arguments can begin to be addressed.

Benefits of the Safety Argument and Evidence Approach

The safety argument and evidence approach is the most contemporary of the approaches examined. It has emerged primarily in response to the limitations of the assurance level approaches. A review of relevant literature and examination of industrial practice reveals the following anticipated benefits:

- **Product Focussed.** The approach emphasises the product-based assurance aspects, and thus is conceptually more straight-forward to relate to product safety risk than the process-based approaches.
- **Flexible Rationale and Evidence.** The approach permits evidence to be chosen that is specifically relevant to the safety arguments being made (Kelly, 1998). This inherently provides flexibility for selection and methods based on specific system, problem or design solution (Kelly, 2008).
- **Standards Endure.** Standards do not require updating in response to changes in the rapid progression of software and related technologies or the establishment of new techniques/methods (McDermid & Rae, 2012).
- **Pattern Reuse.** Patterns provide guidance on acceptable types of arguments and evidence for software aspects of systems (Weaver, 2003). Likewise anti-patterns have also been proposed to describe unsuitable approaches (Kelly, 1998).

It is important to note that these benefits may be realised for the safety argument and evidence approach, irrespective of the specific situation. The key theme that is evident from the benefits of the safety argument and evidence approach is the product focus and opportunity to reason about product behavioural properties with respect to safety and risk. The concept of patterns is useful because it provides a means to reduce the inherent variability of this approach. Therefore, it is desirable that improvements in assurance frameworks strive to utilise such benefits. This is examined further in Section 2.7.

Limitations of the Safety Argument and Evidence Approach

However, despite the benefits, a review of the literature indicates that there are a number of criticisms of the safety argument based approaches. These are as follows:

- **Compromised Objectivity.** Safety arguments tend to converge on the answer the supplier wants (i.e. the "system is acceptable safe") and thus tend to be self-fulfilling prophesises (Kinnersly, 2011). Such safety cases don't truthfully represent counter evidence because the counter evidence doesn't support the positive claims being made. Further, the safety arguments within safety cases are rarely challenged by the assessment process (Kinnersly, 2011). There are issues of how such challenges and rebuttals are accommodated within the safety case (Bloomfield & Bishop, 2010). Conventional contracting processes related to supplier delivery of safety case documents (milestone drafts, and final), and acquirer review of the safety case don't encourage safety arguments to be challenged, or assessors to search for counter evidence (Kinnersly, 2011).
- **Incomprehensible.** Compelling arguments may be difficult to construct for non-experts (Hawkins, et al., 2011). Conversely weaknesses of the argument are often not evident and so are easily overlooked by assessors. Arguments are often indirect and unfocused, and the link between elements of the argument and risk is often lost. This causes safety arguments tend to become large and incomprehensible; there is too much information in the argument, leading to lead to *"voluminous, rambling, ad infinitum arguments"* (Hawkins, et al., 2011). Arguments often suffer from the following:
  - Necessary elements of the argument are sometimes omitted, because the need for the specific elements is lost in the volume of the argument (Hawkins, et al., 2011).
  - Necessary evidence is sometimes omitted, because the need for the specific evidence is lost in the obscurity of the argument.
- **Blurring of Product Focus.** Both the safety argument and the confidence argument tend to be poorly prepared, because the lack of distinction between the two makes it more difficult to spot incompleteness or poor structure in either (Hawkins, et al., 2011). While separation of the product argument and the confidence argument helps, for large safety arguments it may simply not be practical to provide arguments of confidence for every assertion in the safety argument (Hawkins, et al., 2011).

- **Avoid Acknowledging Real Risk.** The approach of arguing that the "system is acceptably safe" fails to adequately inform acquirer decisions regarding risk treatment or retention. In turn this fails to adequately inform supplier decisions regarding necessary changes to their design and additional evidence generation. The emphasis on arguments contributes to a culture of arguing away inadequacies in system design and shortfalls in the evidence of safety, rather than supporting the adequate determination of system design treatments and provision of evidence of safety. Further, safety arguments often give equal attention to hazards regardless of their severity, and thus failing to emphasise where the greatest risks in the system lie (Kinnersly, 2011).
- **Subjectivity Complicates Acceptance.** The subjectivity of arguments in safety cases makes achieving mutual acceptance between suppliers and certifying authorities difficult (Kelly, 2008).
- **Trustworthiness Undermined.** Uncertainty in the provenance of evidence in safety arguments can undermine trustworthiness in the evidence and in the safety arguments (Habli & Kelly, 2007).
- **Difficult to Interpret.** Decomposing the abstract objectives set forth within the standards to practice can be difficult for some suppliers (Kelly, 2008). The additional guidance required to help suppliers understand acceptable means of compliance had lagged release of the standards significantly.
- **Flexibility Not Exercised.** In practice top level safety arguments tend to follow the same repetitive, mechanical format (Kinnersly, 2011), leading to doubt that substantial flexibility is required in this part of the safety case.
- **Difficult Maintenance.** Despite intentions that safety cases are 'living' documents (Kelly, 1998), and methods having been suggested for safety case maintenance (Kelly, 2008), most safety cases languish on shelves after their initial development (Kinnersly, 2011).

Many of the above problems with current practice in the application of safety cases were highlighted by (Haddon-Cave, 2009). It is important to note though that these limitations might not all apply to each specific safety cases because the circumstances of each safety case are different. The key themes evident from the limitations of the safety argument and evidence approach centre around subjectivity and the impacts on supplier, acquirers and certifying authorities in resolving this subjectivity and managing variability. Subjectivity and variability will complicate the enforcement of certification requirements within contracts used for military system. Further, the approach needs to ensure that systems that present intolerable risks are clearly identified as requiring treatment. It should not only be feasible to build a compelling safety argument and evidence for a good design, it is necessary to identify unsafe designs as non-compliant against the safety objectives, either because of inferior product argument or evidence. Therefore, it is desirable that improvements in assurance frameworks strive to reduce the impact of such limitations. This is examined further in Section 2.7

2.4.4 Presenting Safety Arguments

Because knowledge will be assumed in later chapters of this thesis, it is relevant to review methods of presenting safety arguments. There are numerous means of presenting a safety argument within a safety case. For example, (Department of Computer Science, 2004) describes safety arguments in the following forms:

- Textual Narrative
- Tabular Format / Traceability Matrices
- Argument Notations:
    - Claim Structures,
    - Toulmin Structures (Toulmin, 1958),
    - Adelard's Claims-Argument-Evidence (CAE) notation (Adelard, 2008),
    - Bayesian Belief Networks (Littlewood, et al., 1998), and
    - Goal Structuring Notation (GSN) (Kelly, 1998), (Origin Consulting Limited, 2011).

Additionally, (Object Management Group, Inc., 2012) proposes a Structured Assurance Case Meta-model (SACM), which is comprised of two specifications:

- Argumentation Meta-model (ARM) (Object Management Group, Inc., 2010), and
- Software Assurance Evidence Meta-model (SAEM). (Object Management Group, Inc., 2010)

The effectiveness, however, of communicating the safety argument varies depending on the means of expression. For example (Ankrum & Kromholz, 2005) identifies that the existing frameworks for constructing (e.g. textual / tabular forms) and evaluating (e.g. human narrative review) assurance cases often provide excruciating detail about the final table of contents but offer little about how to identify, collect, merge, and analyse technical evidence. Some *"generate large volumes of data without offering guidance for navigation and analysis"* (Ankrum & Kromholz, 2005). Deriving a single judgment of safety or risk from this is often an informal process of "expert judgment", which may be unreliable and is difficult to analyse and verify (Littlewood, et al., 1998). The development of argument notations for the safety domain has provided a means to improve the level of expression and clarity in presenting the safety argument over textual methods. Although representing an argument graphically clearly disambiguates the structure and elements of the argument, it cannot ensure that the argument itself is 'good', or sufficient for its purpose (Hawkins, et al., 2011). While methods such as BBNs offer a formal mathematical language for providing quantitative analysis and reasoning in uncertain situations (Littlewood, et al., 1998), the underlying quantitative values are nothing more than an encoding of confidence. Therefore such notations do not in themselves assure a compelling argument.

2.5 Background on the Safety / Risk Case

Section 1.2 described that within contemporary practice the achievement of safety is articulated and justified through a safety case. Section 1.2 provided a definition for a safety case. This is reinforced by (Kelly, 1998) who states that a safety case *"communicates a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context"*.

2.5.1 Structure of the Safety Case

A safety case normally consists of two elements:

- *Safety Argument* which presents the principles on which the safety is based and identifies the safety objectives and requirements (Department of Computer Science, 2004). The safety argument is "a *logically stated and convincingly demonstrated reason why safety requirements are met"* (Ministry of Defence, 2007). It *"communicates the relationship between the evidence and objectives"* (Kelly, 1998).
- *Evidence* supporting the safety argument.

(Department of Computer Science, 2004) makes an important distinction between the safety case and safety case report. The safety case is the totality of the safety justification and all of the supporting material. Supporting material might include testing reports, validation reports, relevant design information, modelling, simulation, analysis, etc. The safety case report is the document that presents all of the key components of the safety case and references all supporting documentation (Department of Computer Science, 2004).

While these definitions are broadly reflective of contemporary practice, there are variations on the approach to documenting and communicating the safety case and variations to the principle inclusions to a safety case. Table 8 summarises the approaches used for documenting and communicating the safety justification by several military and civil airworthiness authorities.

Authority	**Achievement of Safety articulated and justified by:**	**Source**
ADF	Safety Case Report: MIL-STD-882C programs: the Safety Case Report is a summary report consisting of the Task 301 Safety Assessment, Task 401 Safety Verification and Task 402 Safety Compliance Assessment ARP4754 programs: the Safety Case Report is a summary report summarising the results of the Aircraft and Systems Functional Hazard Assessment, System Safety Assessment, Common Cause Analysis and Health Hazard Assessment.	DI (G) OPS 2-2 (Defence Aviation Safety Authority, 2011) AAP7001.053 (Directorate General Technical Airworthiness, 2010) AAP7001.054 (Directorate General Technical Airworthiness, 2010) Sect 2 Chap 1
UK MoD	Safety Case Report: a report that summarises the arguments and evidence of the safety case, and documents progress against the safety programme.	Defence Standard 00-56 (Ministry of Defence, 2007)
USAF, USN, US Army	DI-SAFT-80102 Safety Assessment Report DI-SAFT-81300 Mishap Risk Assessment Report	MIL-STD-882C (US DoD, 1993) MIL-STD-882D (US DoD, 2000)
FAA, EASA, CAA, CASA	Reports summarising the results from Aircraft and Systems Functional Hazard Assessment, System Safety Assessment, and Common Cause Analysis.	ARP5754 (SAE International, 1996)

Table 8: Safety Justification Artefacts of Airworthiness Authorities

Despite the differences in report types and contributing assessments, the various reports provide the justification as to why the system is acceptably safe or why the risk is acceptable or tolerable. This thesis will primarily assume the domain of the safety case and the associated arguments and evidence, although it will be evident that the ideas developed within this thesis have read across to the other approaches used to justify safety.

2.5.2 The Emergence of the Risk Case

In the report into the loss of the RAF Nimrod in 2006 (Haddon-Cave, 2009), Haddon-Cave recommended that safety cases be renamed and be made more focused, proportionate, and relevant. The official recommendation is as follows:

> *Recommendation 21.E.1: The Regulator shall set the requirements for a single, concise, through-life "Risk Case" for each platform in a format which stimulates effective analysis, encourages focus on key risks and can easily be assimilated and understood by the intended user.*
>
> (Haddon-Cave, 2009)

(Haddon-Cave, 2009) proposes a simple definition of risk case as *"reasonable confirmation that risks are managed to ALARP."*. (Haddon-Cave, 2009) is not the only source to criticise the safety case approach. Criticisms of safety cases are:

- They shouldn't argue that the system is safe; they should argue why the risks are controlled and indicate those areas where remedial action is needed to achieve an acceptable level of safety.
- The focus should be on decision-making: for both decisions as to the acceptance of risk and decisions as to the deployment of resources to reduce risk.
- They should not be a 'snapshot' report or an 'archaeological' collection of documents; they should be actively informing decisions on risk treatment or retention.
- The definition of safety case in (Ministry of Defence, 2007), *tends to encourage a laborious, discursive, document-heavy 'argument' aimed at justifying a self-fulfilling prophesy* (Haddon-Cave, 2009).
- They suffer from factors outlined by (Kelly, 2008) including: the apologetic safety case, the documents-centric view, the approximation to the truth, prescriptive safety cases, safety case shelf-ware, imbalance of skills, and the illusion of pictures.

Based on these criticisms Haddon-Cave proposes a *"paradigm shift is required away from the current verbose, voluminous and unwieldy collections of text, documents and GSN diagrams to risk cases which comprise succinct, focused and meaningful hazard analysis which stimulate thought and action."*

To some extent the concept of the risk case is already evident in the safety case approaches of the ADF and US military. For example the Mishap Risk Assessment Report often used in US programs focuses on risk, although these reports are not immune from the criticisms of (Haddon-Cave, 2009) either. While it is yet to be clear if the risk case recommendation will be fully adopted by stakeholders, the evidence supporting the criticisms of the use of safety cases is largely indisputable, and should prompt further examination into the role and usage of safety cases. In the context of this thesis, it is important to understand the impacts of these criticisms on certification of aviation systems. Section 2.6 will discuss this further.

2.5.3 Software Safety Cases

So far this section has discussed safety cases in general, and so it is worthwhile clarifying the term *software safety case*. A software safety case is the element of a safety case that argues the safety of the software component of the system. The software safety case should recognise that the software does not exist in isolation to the remainder of the system and that the software's interaction with the hardware and other elements of the system are crucial. The software safety case records the software viewpoint in terms of the safety case. Views are very common in engineering and computer science (Clements, et al., 2010).

2.6 A Discussion of Current Approaches

Section 2.4 provided an overview of the contemporary approaches used for assurance of safety-related software systems. From these overviews it is evident that the practice of safety assurance varies substantially between the assurance level, evidence assurance level, and safety argument and evidence approaches. There is also variation between domains (i.e. between military and civil aviation, rail, etc.), and there is variation between the specific requirements of standards. Because there is variation, it follows that there will also be variation in the benefits and limitation of using these approaches.

The specific benefits and limitations of each approach were also analysed, key themes which should be factored into the application of safety assurance frameworks identified. Section 2.7 will look at how these themes can be used to derive general principles for an assurance framework in this context. However, before doing so, it is worthwhile to examine the literature that provides holistic analysis of these approaches and sets a broader direction for principles for an assurance framework.

2.6.1 Disputing the Assurance Level Approach

The assurance level approach is in widespread use, and may be viewed as the accepted norm, at least in some domains (McDermid, 2001). Even the most recent revisions to standards of this type (e.g. (RTCA Inc., 2011), (IEC, 2010)) have continued to use the assurance level approach.

However, despite the benefits, the assurance level approach is heavily criticised. Taken individually, none of the difficulties are sufficient to suggest that the current approaches should be abandoned, but taken together they suggest a more systematic and defensible approach is needed (Lindsay & McDermid, 1997). (Committee on Certifiably Dependable Software Systems, 2007) notes that in part the problems are due to inadequate oversight, inconsistent application of the standards, and processes established without regards for principles of standards. However this is not the whole answer, and the answers lie in the lack of product focus and relevance to risk assessments.

2.6.2 The Assurance Level Approach Has Value

Despite the criticisms, (McDermid, 2001) stresses that the assurance level approaches have value, and the criticisms and need for change should not be interpreted as a ‘free for all’ in development. The assurance level based standards do contain a lot of sensible requirements, advice and guidance on development. Their emphasis on requirements traceability also assists with ensuring requirements satisfaction is achieved. However, the approach doesn’t focus on providing information about the properties of the system that contributes to safety. What is needed is more focus on the product and the validity of the safety requirements for that product. Therefore this thesis asks if it possible to give the assurance level approaches a greater product focus while still preserving their benefits. Likewise is it possible to impart some of the assurance level’s benefits into the safety argument and evidence paradigm. If principles to this effect can be established, they might make the existing approaches more complementary.

2.6.3 The Need for Product and Evidence-based Approaches

The emergence of safety argument and evidence assurance based approaches is indicative of a growing concern about validity of the previously accepted wisdom. There is widespread support within the literature that the approach should be to seek explicit evidence of safety, rather than making a general appeal to the development processes (Committee on Certifiably Dependable Software Systems, 2007).

2.6.4 Safety Argumentation Needs Enhancement

The safety argument and evidence approach can provide a more product-assurance focused framework that provides an opportunity for addressing the recommendations for explicit claims and evidence. However a problem is that it gets misused because of its permissive subjectivity leading to unconstrained variation in judgements. The prominent reasons for this are the relative immaturity of safety argument methods, and the lack of experience in using product evidence as the main thrust of safety assurance. (Bloomfield & Bishop, 2010) suggest that contemporary safety argumentation and associated methods need to be enhanced to achieve this. Recent research literature also reflects a disproportionate focus on questions of argument, rather than questions of evidence, and thus more research is required regarding the problems of evidence sufficiency. A system should be regarded as dependable only if sufficient evidence of its explicitly-articulated properties is presented to substantiate the dependability objectives (Committee on Certifiably Dependable Software Systems, 2007). (Committee on Certifiably Dependable Software Systems, 2007) suggests that in practice, certification will be based on inspection, analysis and challenging of the dependability claim and the evidence offered in its support. Where over-sight by regulators is less prevalent, the approaches should provide more transparency, so that users can make informed judgements about dependability.

However, as illustrated by Section 2.2, the approach adopted has to be complementary to the context of the specific regulatory environment and overall approaches to safety in that domain (Bloomfield & Bishop, 2010). This is a significant weakness in the literature pertaining to safety assurance. This thesis is one of the few works which addresses the contextual issue of the certification environment and the application of a product based approach.

2.6.5 Never Enough Evidence

There will never be unlimited evidence, because there is never unlimited time and money. So there will always be limitations in evidence, and it is important that an assurance framework recognises this. (Littlewood, 2007) contests that *"it still remains impossible to show, before using it, that a system will be extremely dependable in operation"*, on the basis of what (Littlewood, 2007) describes as the *"unforgiving law about the extensiveness of evidence needed to make very strong dependability claims"*. However despite these reservations, the world's demand for systems will prevent any halt on the supply of such systems. The burden on those supplying and certifying these systems is to establish ‘reasonability’ in the eyes of the public. Therefore, it is important that in addressing the problems with the current approaches, some consensus on benchmarks is sought on how much evidence, and what sort, is required to make ‘reasonable’ claims about dependability and safety?

2.6.6 Informing Risk in Real Time

Safety (assurance) cases should be living documents, to reflect changes to the system and operational context over time (Kelly, 1998). Hence the case should not be constrained to initial development, and should evolve and continue to inform risk assessments throughout the life of the system. For example, (Ankrum & Kromholz, 2005) states that *“assurance case frameworks address new software development but rarely consider the larger lifecycle, including how to maintain confidence as the software evolves”.* Section 2.4.3 has also identified that many safety cases become large and incomprehensible, which make them difficult to maintain. This is important because certifying authorities and operators need to continually identify, analyse and evaluate risks during operation. Thus the assurance framework needs to be useable during operation. It should help users make on-going risk treatment decisions.

2.6.7 Understanding the Lack of Consensus

Safety assurance is only effective if it is not only possible to produce a compelling safety case, but it is probable. Approaches that don’t result in probable production of a compelling safety case are potentially not effective. For any approach (new or existing) to be effective, it must be possible to not only identify examples where the approach is successful, but also ensure counter-examples don't indicate ineffectiveness. Where there is ineffectiveness, it must be possible to provide explanation for the ineffectiveness.

As can been seen from the presentation of benefits and limitations, the variation in the approaches is indicative of differing philosophies of safety assurance standards between domains and regulatory contexts. The variation reflects the extent to which prescriptive and goal-based approaches are favoured, and thus the commensurate emphasis on process-assurance and product-assurance. The variation is also indicative of a lack of consensus in practice. What can be concluded from this lack of consensus is that current approaches to providing safety assurance have limitations. Thus, as neither paradigm is without its limitations in this context, it is possible that the more effective approach may be a compromise between both paradigms. Certainly, the symmetry between prescriptive and goal-based approaches identified in Section 2.2.3 suggests this.

This should be investigated because the context of suppliers, acquirers and certifying authorities is something that has been largely absent from literature criticising assurance frameworks. While this does not invalidate the criticisms it does mean that there has been an assumed context to the criticisms which warrant better understanding.

2.7 Providing an Approach for Addressing the Limitations with Current Approaches

This chapter has examined the certification of safety-related aviation systems and summarised the current and contemporary approaches. Section 2.4 has examined the benefits and limitation of these approaches, and identified themes amongst the benefits and limitations that should be addressed if better approaches are to emerge. For the assurance level approach the themes are summarised as follows:

- Reducing variability in presentation of evidence is beneficial for relationships between suppliers, acquirers and certifying authorities.
- Consensus on benchmarks for trustworthy evidence has benefits for suppliers, acquirers and certifying authorities
- Aspects of the prescriptive approach are beneficial where contracts are used to enforce certification requirements relating to safety assurance.
- Limitations in explicit product behavioural focus complicate the assessment of risk and the achievement of safety objectives.
- Lack of clear rationale for prescription of techniques and methods with respect their risk reducing role leads to confusion.
- Absence of a means for assessing the impact on safety risk when there is a shortfall in evidence against requirements complicates usage.

For the safety argument and evidence-based approaches, the themes are as follows:

- Evidence needs to be appropriate to the requirements and failure modes to which it relates, and the properties established from the way differing evidence combines are more important than simply the type of evidence presented.
- Product focus and opportunity to reason about product behavioural properties with respect to safety and risk is beneficial.
- The concept of patterns is useful because it provides a means to reduce the inherent variability.
- The major drawbacks are subjectivity and the impacts on supplier, acquirers and certifying authorities in resolving this subjectivity and managing variability.

- Subjectivity and variability complicate the enforcement of certification requirements within contracts used for military system.
- Intolerable risks are clearly identified as requiring treatment. It should not only be feasible to build a compelling safety argument and evidence for a good design, it is necessary to identify unsafe designs as non-compliant against the safety objectives, either because of inferior argument or evidence.

In the military certification environment the contract is important and integration between the contract and assurance is vital if approaches are to be successful.

This thesis discusses how these themes can be used to derive general principles for an assurance framework. The intent is to find the appropriate balance between goal-based and prescriptive elements. It will also provide a way to understand how the limitations of the current approaches may be resolved without introducing further limitations.

There are benefits if the assurance framework can accurately disclose the risk, in the presence of an evidence set that has been benchmarked against benchmarks established by consensus of regulators and industry. For this goal, it will be important to be able to:

- reason about the impact on risk of limitations,
- make informed decisions before entering into contract, and
- have clear expectation regarding resolving safety shortfalls and evidence shortfalls within the scope of the contract (in a fixed price paradigm).

If these things can be achieved more programs should be completed within cost and schedule constraints.

2.8 Thesis Contribution

2.8.1 Research Questions

The review of literature has motivated the following research questions:

- ***General Principles.*** *Is it possible to establish general criteria for safety assurance based on compromise between benefits and limitations of the contemporary approaches surveyed?*
- ***Informing Risk Decisions.*** *Is it possible to identify criteria for safety assurance to enable stakeholders to make informed judgements about risk?*

- ***Consensus on Argument and Evidence.*** *Is it possible to identify criteria for safety assurance that assist with achieving consensus between suppliers and certifying authorities regarding suitability of argument and evidence?*

- ***Contractual Enforcement.*** *Is it possible to identify criteria for contracts to permit communication and enforcement of safety assurance through contracts between suppliers and assessors in the military domain?*

- ***Architectural Properties.*** *Is it possible to use the properties of aviation systems (e.g. aircraft flight control systems, flight instruments, navigation systems) to identify additional criteria for safety assurance?*

- ***Practice.*** *Is it possible to develop a practical safety assurance framework that adheres to these criteria?*

2.8.2 Thesis Proposition

From these research questions, the author presents the following thesis hypothesis:

This thesis demonstrates that it is feasible to establish principles and usability criteria for defining effective safety assurance frameworks for aviation systems in typical acquisition contexts. This thesis provides meta-arguments that can be used as the basis for defining a novel integrated framework for the assurance of aviation systems. The thesis demonstrates how this approach can be used to address the identified limitations and challenges of the certification of aviation systems. Further, by reducing uncertainty for supplier delivery of safety evidence across contracting processes, the framework is intended to help limit emergence of safety evidence issues, the resultant cost and schedule implications, and reduce the likelihood of retaining intolerable safety risks.

2.9 Summary

This chapter has reviewed the current approaches for assurance of safety related systems, with an emphasis on the treatment of systematic failures. The certification environment and contractual instruments with which these approaches are used was also reviewed. The chapter has identified that approaches can be categorised as prescriptive or goal-based, albeit some approaches inherit properties from both. Neither approach is without limitations; however notably the limitations differ between approaches. Both approaches also have benefits for safety assurance and for the relevant certification frameworks. The review identified that there is a lack of consensus on which approach or which combination of approaches is more effective. There was also a lack of consensus on the purpose of safety assurance, and how this differs from the existence of a safety argument as part of a safety case. There was, however, general consensus that a greater product focus is required and that improvements are needed achieve this. Research questions and a thesis proposition have been defined.

3 Establishing General Principles for Safety Assurance Frameworks

Safety assurance frameworks are only effective if it is not only possible to produce a compelling safety case, but it is probable. To establish which properties from prescriptive and goal-based approaches are most beneficial to safety assurance, this chapter establishes general principles and usability criteria for effective safety assurance frameworks. These general principles and usability criteria form the basis for the definition of a specific framework. The intent of the framework is to demonstrate that it is feasible to inherit properties from both the prescriptive and goal-based approaches to achieve a product focus and compatibility with certification environments and contracting methods.

3.1 Clarifying the Terminology

Substantial importance is placed on the terminology used within this thesis. Section 1.2 provided definitions for key terms that are inherited from existing standards and literature. Relationships between terminologies are also important. In order to ensure consistent understanding, this section clarifies terminology to be used, and defines meta-models based on relationships between terminology.

3.1.1 Parts of the System

In any complex system, it is important to be able to refer to parts of the system accurately. In this thesis the following parts hierarchy is used (refer Figure 2). This hierarchy has been derived from terminology used in (SAE International, 1996) and (SAE Aerospace, 2010).

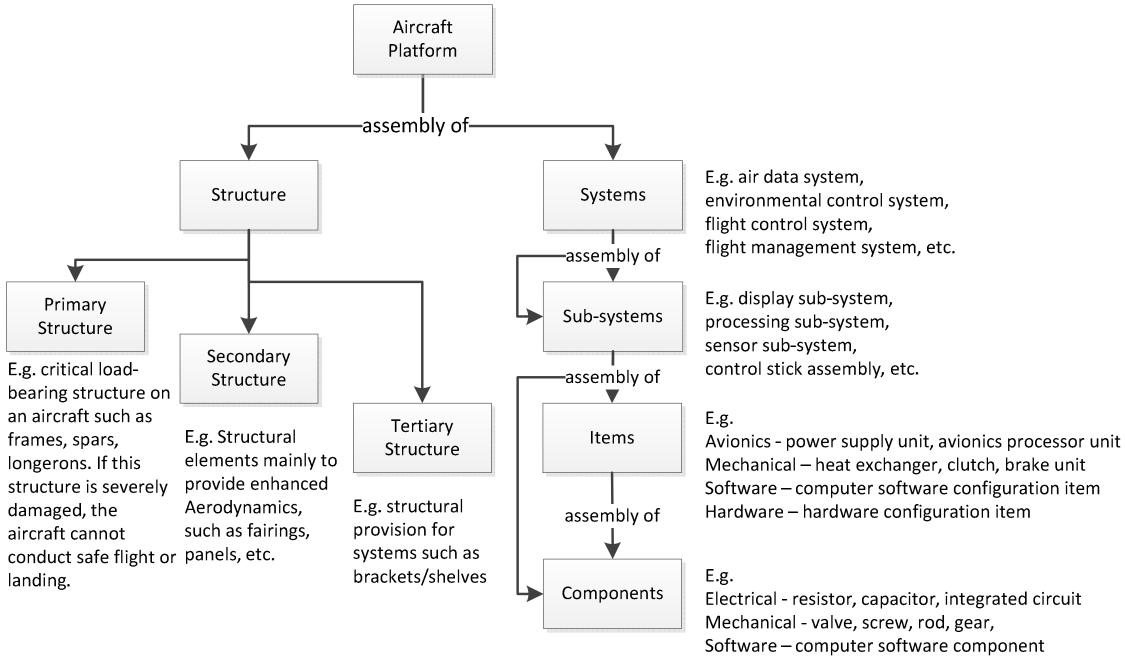

Aircraft Platform
assembly of
Structure
Systems
E.g. air data system, environmental control system, flight control system, flight management system, etc.
assembly of
Primary Structure
Sub-systems
E.g. display sub-system, processing sub-system, sensor sub-system, control stick assembly, etc.
Secondary Structure
E.g. critical load-bearing structure on an aircraft such as frames, spars, longerons. If this structure is severely damaged, the aircraft cannot conduct safe flight or landing.
assembly of
Tertiary Structure
Items
E.g.
Avionics - power supply unit, avionics processor unit
Mechanical – heat exchanger, clutch, brake unit
Software – computer software configuration item
Hardware – hardware configuration item
E.g. Structural elements mainly to provide enhanced Aerodynamics, such as fairings, panels, etc.
E.g. structural provision for systems such as brackets/shelves
assembly of
Components
E.g.
Electrical - resistor, capacitor, integrated circuit
Mechanical - valve, screw, rod, gear,
Software – computer software component

Figure 2: Hierarchy of Parts and Systems

3.1.2 Evidence, Behaviours, Hazards, and Consequences

The terms evidence, hazards and consequences are widely used in the safety assurance literature, as is evident from Chapter 2. The definitions of these terms (from Section 1.2) imply certain relationships which are important for defining safety assurance. Figure 3 summarises these relationships, such that the purpose of safety assurance can be further developed. This meta-model will be assumed vocabulary.

3.1.3 Behaviours

In defining the relationships within Figure 3, it has been necessary to introduce terminology pertaining to systems producing hazards. This thesis supposes that systems exhibit behaviours, and that these behaviours may be desirable or undesirable with respect to safety. Those behaviours which are undesirable may produce hazards. For example, consider an aircraft flight control system which can produce a hard-over under certain conditions, which unrecovered would result in loss of continued safe flight and landing (i.e. a crash). Such behaviour is both undesirable and of catastrophic consequences. Clearly a goal of safety assurance is to provide confidence that the designers have prevented these consequences, by controlling hard-over related hazards, by constraining behaviours of the system that could produce such hazards.

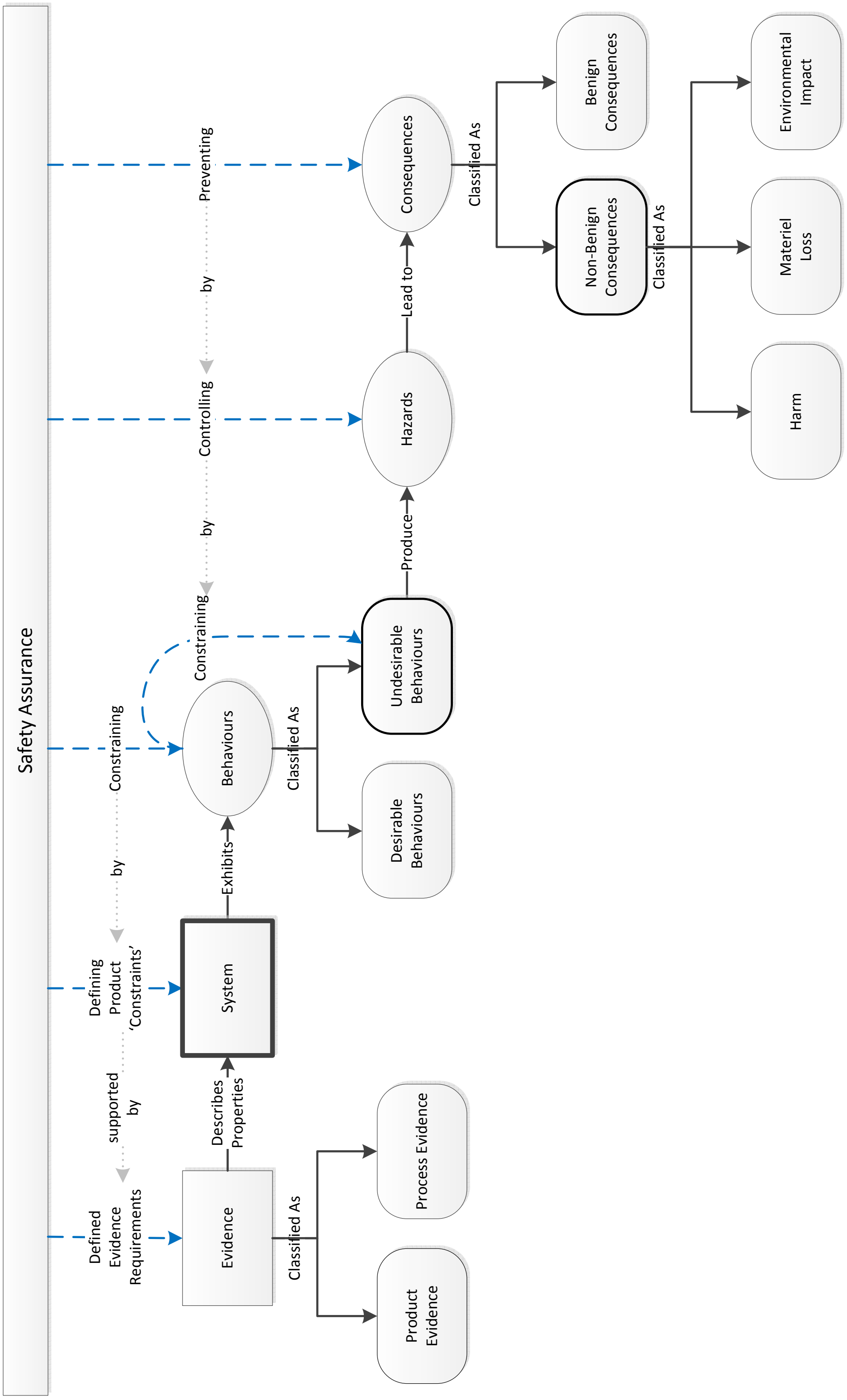


Figure 3: Evidence, Behaviours, Hazards and Consequences

3.1.4 Constraining Behaviours with 'Constraints'

Figure 3 annotates the concept of Product 'Constraints'.

Constraint

A requirement on the system to constrain one or more of the systems behaviours, such that the behaviours exhibited by the system are desirable with respect to safety. Constraint of behaviours may be by means such as prevention or tolerance.

Working Definition

The concept of a 'Constraint' will be elaborated further in Chapter 5. However, notionally, the 'Constraint' is, for example, intended to be consistent with (Weaver, 2003)'s usage of (Pumfrey, 1999)'s classification of software functional failures modes (Omission, Commission, Early, Late, Value), and the requirements (Weaver, 2003) identified on the system and its evidence for treating them (Absence or Detection/Handling). The 'Constraint' is a more generic representation of this concept that can be applied at the system perspective, and not just for software. The reader is referred to (Weaver, 2003) for a description of these concepts.

To illustrate this concept, consider the following. A constraint which uses prevention (or absence) to constrain undesirable behaviours implies there are evidence requirements to demonstrate the prevention. Whereas a constraint which uses tolerance, may need to define both product behavioural requirements, such as a detection/handling capability and the evidence to show that these are correctly implemented. Such implications are important for safety assurance.

3.2 Purpose of Safety Assurance

A purist might argue that the only concern of safety assurance is achieving safety through minimising risks to a level for which there is societal consensus. However even the most cursory inspection of any number of safety assurance standards reveals that these standards seem to concern themselves with a much greater range of factors than just safety. Consider the following description of the purpose of safety assurance standards by (McDermid & Rae, 2012), which states that at least one view of the purpose of safety assurance is to:

- require a minimum standard of safety to be achieved and demonstrated;
- where further safety can be achieved above the minimum, require safety improvement to be balanced with the cost of safety improvement; and
- minimise the cost required to achieve and demonstrate safety.

While it is clear that the purpose includes safety, there are additional factors and relationships expressed, as follows:

- *standard of safety achieved* which is about the safety of the product,
- *demonstration of safety* which is about the way the safety is shown, and
- *cost of safety* which relates to the cost of both achievement and demonstration.

Relationships are also expressed between these factors, as follows:

- *safety / cost relationship* – achieve the minimum standard of safety while minimising cost of achievement,
- *demonstration / cost relationship* – demonstrate the minimum standard of safety while minimising cost of demonstration, and
- *balanced safety / cost improvement* – balance further improvement of safety with the cost of safety improvements.

Of note, the cost factor is prominent in each of these relationships, revealing a practical element of safety assurance. The following sub-section examines this further.

3.2.1 Role of Cost of Achievement and Demonstration

The cost factor and its relationships are prominent in (McDermid & Rae, 2012)'s purpose statement. The notion of cost is important because it emphasises a practical 'real-world' aspect of the achievement and demonstration of safety. It recognises that doing these things takes resources, time and money; and how well they are done is inseparable from the cost of doing them. Minimising cost is intuitively sensible as it is credible commercial goal to inspire efficiency improvement in the achievement and demonstration of safety. However from the very outset it emphasises there will always be limitations to achievement and demonstration based on cost drivers. Hence it implies that a significant aspect of safety assurance is actually the measurement and management of these cost relationships, both for achievement and demonstration.

The contemporary approaches (refer Section 2.4) implicitly acknowledge the role of the cost factor, but have typically excluded, avoided or struggled to comprehend how to express these relationships. Some make assumptions about the cost of safety, and this becomes implicitly encoded in their requirements; while others ignore it and struggle with relevance to practicality. Therefore, it is important to have a clear understanding about the cost relationships for both achievement and demonstration to ensure standards appropriately express safety assurance.

3.2.2 Understanding Achievement and Demonstration

Achievement and demonstration are different concepts, but they are not independent. If achievement of safety is about minimising accidents and reducing their severity through treating risks (refer to definitions of safety, accidents and risk from Section 1.2), then informing the design and operation of these systems to treat or manage those risks also forms part of the evidence used in the demonstration of safety achievement. It implies that sufficient (minimal) information is necessary to understand the risks and make well-informed judgements about the costs and benefits of design options and operational risk treatments used for achievement. Because of this information dependency, achievement is inseparable from enabling elements of demonstration.

Therefore, an additional qualifier to the purpose of safety assurance is the relationship between achievement and demonstration to enable cost minimisation. The purpose should recognise that certain minimum information associated with the demonstration is required to adequately inform achievement. The following paragraphs examine specific measures of achievement and demonstration that will further assist with defining this.

Measures of Achievement

When it comes to the cost of achievement of the minimum standard of safety, there are several well established measures governing this. For example some approaches qualify acceptable, tolerable (or acceptable with higher authority approval) and unacceptable levels of risk. Concepts such as ALARP[11] are then also applied to provide a measure of adequacy of risk reduction (and thus safety achievement) versus cost obligations.

However, because of the information dependency between achievement and demonstration outlined above, there are problems when the ‘traditional’ interpretation of ALARP assumes that this information is without cost (McDermid, 2012). In other words, the only cost considered in ALARP is that of taking action to mitigate risk, not of the work required to establish the risks, determine potential action options, and the potential cost and benefit of such action. In situations where the cost of information is high, such as for aviation systems, this becomes a significant problem. In practice this leads to the intent of ALARP being undermined by the cost of getting the information needed to make ALARP decisions. Hence an additional purpose of safety assurance is

[11] As described by references such as (United Kingdom Goverment, 1974), (UK Health and Safety Executive, 2013) and (Ministry of Defence, 2007)

to set a minimum standard for the information required to inform risks and risk reduction decisions. This information includes the identification, analysis and evaluation of risks, and the identification and analysis of possible treatment options.

Measures of Demonstration

However, for the cost of demonstration, the measures are less clear than for achievement. This is due to the lack of consensus on exactly what should be shown to demonstrate achievement of safety. Some 'traditional' approaches try to avoid this problem by simply prescribing evidence and/or argument for demonstration, but often without any explicit rationale. They effectively make an assumption about what demonstration should be, rather than making explicit the rationale for it.

Other approaches emphasise the concept of confidence. For example, (McDermid, 2008) has proposed the concept of As Confident as Reasonably Practicable (ACARP). ACARP proposes that there is, in effect, a scale of confidence in the evidence and argument available to demonstrate safety, and that this scale of confidence can be treated similarly to the way risk is considered under ALARP. Thus for each limitation in confidence, treatment options should be identified, and decisions made regarding which treatment options provide justifiable benefits to confidence. However, ACARP is little more than a concept at this point, and there aren't any frameworks which formalise this concept as yet. Because of this, there is also limited literature as to whether it actually resolves the demonstration issue. At first glance, it seems to have a comparable limitation to ALARP in that it doesn't acknowledge the minimum information requirement to inform demonstration treatment options. However, there is benefit to having a classification of criteria for measuring when confidence is sufficient.

Demonstration of safety, in essence, is a measure of what knowledge there is of risks, and what opportunity there is for this knowledge to be undermined by uncertainty. Only when the knowledge of risks outweighs the impact of potential uncertainty of risks, can it be possible to reason that demonstration is achieved. Otherwise, there remains the opportunity for the knowledge to be fundamentally undermined by uncertainty.

To articulate this, Figure 4 revises Figure 3 to also show the concept of establishing knowledge and uncertainty of risks.

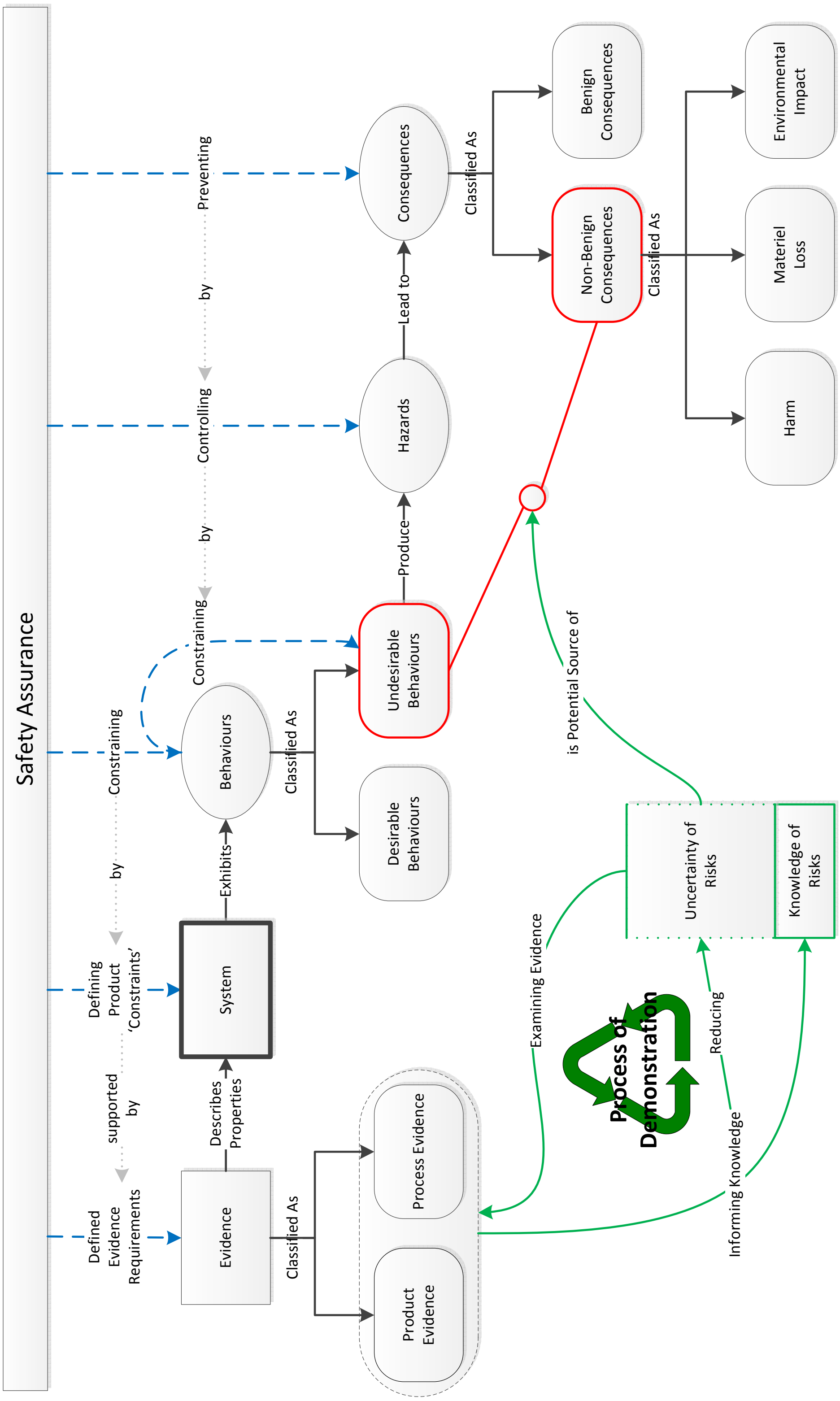


Figure 4: Knowledge and Uncertainty in Safety Assurance

Figure 4 represents that there will always be gaps in knowledge of risks, and thus it is important to be able to determine when knowledge sufficiently outweighs uncertainty when measuring demonstration of safety. It will be important to characterise to what extent the overall risk is based on the following:

- *Known Knowns*[12]*:* known product behaviours which result in risks,
- *Known Unknowns:* known limitations in the extensiveness of the evidence and unknowns about product behaviours;
- *Unknown Unknowns:* unknown product behaviours because of limitations in the evidence, and thus limitations in knowledge.

In Figure 4, uncertainty of risks is represented by a dashed box to illustrate uncertainty is not directly measureable. Instead the goal is to examine how knowledge might be undermined by uncertainty and reduce such uncertainty until it is tolerable.

With respect to demonstration, there is also a notable exclusion in (McDermid & Rae, 2012)'s purpose statement. While the purpose mentions a minimum standard of safety (achieved), and thus implies that the minimum cost of achieving safety is limited by achievement of the minimum standard of safety, there is no equivalent statement for a minimum standard of demonstration of safety. Thus it is not possible to automatically imply a minimum cost of demonstration, and it is not explicit within the purpose statement. Hence an additional purpose of safety assurance is to also set a minimum standard for the demonstration.

3.2.3 An Improved Purpose of Safety Assurance

In sub-section 3.2.2 additional factors and relationships have been established that require inclusion within the purpose of safety assurance. These relate to minimum information required to inform achievement and the minimum standard of demonstration. Restating the purpose of safety assurance results in the following purpose statement (with enhancements shown in italics):

- require a minimum standard of safety to be achieved and demonstrated;
- *require the achievement of a minimum standard of safety to be informed by a minimum set of information*
- *require a minimum standard of demonstration;*
- where further safety can be achieved *or demonstrated* above the minimum, require improvement to be balanced with the cost of improvement; and
- minimise the cost required to achieve and demonstrate safety.

[12] There is a fourth category of Unknown Knowns; however in this context, the notion of communicating the knowledge and proposing an evaluation of the risk effectively shifts any knowledge from this category into the Known Knowns.

Safety assurance is thus inclusive of these factors and the relationships they imply. It should be unsurprising then that safety assurance frameworks have to find a way to express and measure these relationships. However, how to do that in a way that ensures the purpose of safety assurance remains explicit is something the current approaches struggle with because of the difficulties of addressing the cost of demonstration aspects.

3.3 Purpose of Assurance Standards

Standards typically have a wider role than just achieving the purpose of safety assurance. The following subsections describe what assurance standards are used for.

3.3.1 Standardisation of Acceptable Practice

One important role for standards is to standardise acceptable practice. The word 'standard' in general English language definition can imply the following:

- *"anything taken by general consent as a basis of comparison"* (The Macquarie Library, 2002)
- *"a level of quality which is regarded as normal, adequate, or acceptable"* (The Macquarie Library, 2002)
- *"a level of quality or attainment"* (Oxford University Press, 2010)

The key points here are:

- the basis of comparison (usually expressed as a set of criteria)
- against a measure of acceptability or attainment (i.e. the pass-mark)

Interpreting these key points for a safety assurance standard implies that a safety assurance standard should provide a basis of comparison between a product and its evidence, and the desired outcomes of the standard. In the case of a safety assurance standard the goal is achievement and demonstration of safety. Posing the question rhetorically, what are the structured set of properties of the product and its evidence that permits a conclusion to be established that the product's behaviours are appropriate with respect to safety? To answer this question it is necessary to have criteria for measuring how evidence informs product behaviours and product behaviours inform knowledge of risks, risk assessment and safety.

(Weaver, 2003) describes a two properties of evidence that are useful for establishing criteria for suitability of evidence. These are as follows:

Relevance

The extent to which an item of evidence entails[13] *the requirements for evidence.*

(Weaver, 2003)

Trustworthiness

The perceived ability to rely on the character, ability, strength or truth of the evidence.

(Weaver, 2003)

Unfortunately many of the frameworks (refer Section 2.4.1) underpinning 'traditional' assurance standards confuse premises for conclusions or outcomes. Thus they prescribe a 'basis of comparison' focussed around the methods of development or assessment, rather than about the suitability (relevance and trustworthiness) of the behaviours and evidence with respect to safety. In general, this thesis doesn't raise objection to the many valid premises that underpin these standards, and to which (Weaver, 2003) refers to as 'best practice' or 'good practice'. However the inference that they lead to the right conclusions is usually implicit, if not missing altogether. There are also some instances where it is questionable that a standard's premises even link to an appropriate conclusion, or are there for other reasons otherwise not made explicit. While, this is certainly a limitation to existing frameworks, the developers of these frameworks were not entirely at fault for this circumstance.

When acceptable practice is established based on premises (things practitioners become familiar with through practical experience), then the acceptable practice will focus on the methods (e.g. what test method should be used, how should requirements be written, etc.). This may be acceptable where premises lead directly to conclusions. However for safety assurance standards involving technologies whose failures are dominated by systematic failures, rarely does a premise lead directly to a conclusion. Rarely does a claim from one single piece of evidence relate directly to satisfying a safety objective. This is because the product safety objectives are abstracted from the methods of evidence generation. Furthermore, because of the technologies involved, the plethora of techniques and methods, architectural options and implementation possibilities, there are numerous approaches to any one design problem. This creates a challenging conundrum. Should the assurance standard define the preferred combination of the

[13] To involve, or logically necessitate.

methods (or those premises with which practitioners are most familiar)? or should the assurance standard focus on how the premises link to inferred premises and conclusions? This thesis proposes that the focus should be on the latter[14], rather than the former; although the role of the former should be explicitly recognised. To achieve this focus, examine the chain of definitions from Section 1.2 and Figure 3:

- safety is defined in terms of risk,
- risk is defined in terms of product behaviour contributions,
- product behaviours are defined in terms of information provided by evidence.

It emphasises that these things are the measures of safety. Hence it can be inferred that: ***safety assurance should set outcomes and a basis for comparison for those things most important to safety: risk, product behaviours, rationale, and evidence.*** The principles and usability criteria developed in this chapter will use these factors as the basis for defining safety assurance.

3.3.2 Contractual or Regulatory Compliance

A related role of standards which is applicable to safety assurance standards is providing consistent benchmarks for contractual or regulatory compliance. This is because these standards often form part of commercial and/or legally binding relationships between suppliers, acquirers, and regulators. When a standard is part of such a relationship, it must be possible for the stakeholders to consistently distinguish compliance from non-compliance.

While the concept of benchmarks is relatively straightforward, which benchmarks are suitable, and how best to articulate them, is a substantially more challenging question. The purpose statement from Section 3.2.3 has provided three minimum benchmarks that require articulation:

- the minimum standard of safety achievement,
- the minimum standard of demonstration, and
- the minimum information required to inform achievement.

Hence, a safety assurance standard should concern itself with how to measure these from a compliance perspective. This implies a level of prescription, as necessary to measure the benchmarks identified by the purpose statement. It is also important to note

[14] Because only the latter provides the explicit product argument for safety, whereas the former leaves it implicit. There are other pros and cons though, as outlined in Chapter 2.

that there are practical constraints on assessors. The benefits and limitations discussed in Section 2.6 indicate that variability and subjectivity hinder the basis of comparison. Thus minimisation of unnecessary variability and subjectivity is also a goal.

Hence it can be inferred that: ***it must be possible to distinguish compliance / non-compliance or acceptable/unacceptable; and that reducing variability and subjectivity may assist this.*** The principles and usability criteria developed in this chapter will use these factors as the basis for defining safety assurance.

3.3.3 Compliance Assurance and Managing Risk

Inevitably the compliance assurance programs of regulators will eventually find non-compliances with respect to the criteria of any standard, and the same will apply for safety assurance standards. Suppliers are driven by commercial motivators, and despite good intentions things do get missed or avoided. For safety assurance standards, the important thing is that the meaning in terms of safety achievement can be determined from such non-compliances. For product standards, the meaning is usually fairly straightforward – i.e. the product poses a risk because it doesn't have a particular property or safety feature that would normally be expected for this product. However, for safety assurance standards, the impact of the non-compliance might be less certain, particularly when it pertains to shortfalls in demonstration, or the minimum information necessary to inform decisions fundamental to achievement.

The definition of requirements, objectives and outcomes in an assurance standard should be explicit in product meaning, so the safety impact of any non-compliance can be determined. This provides the regulator with a basis for managing the tolerability of any risk associated with non-compliance, rather than being uncertain as to the specific risk. There are also benefits to this approach if shortfalls are learned about retrospectively, and the regulator is faced with reassessing risk and promulgating interim risk treatments until the non-compliance can be properly resolved.

Hence it can be inferred that: ***the impact in terms of risk of limitations in safety assurance on the outcomes of safety assurance should be explainable.*** The principles and usability criteria developed in this chapter will use these factors as the basis for defining safety assurance.

3.4 Key Principles for Safety Assurance

Having established the purposes of safety assurance based on both the intentions of safety assurance and how it is applied in practice, it is possible to establish principles and usability criteria that the safety assurance frameworks should adhere to. This section proposes a set of general principles for safety assurance frameworks based on the purpose of safety assurance established within this chapter. The intent of these principles is to guide the development of safety assurance frameworks within a specific domain or context. These principles will be used through the remainder of this thesis to develop a safety assurance framework for the specific context of military aviation systems and their associated certification environment.

Section 3.3.1 identified that risk, product behaviours, rationale and evidence are important. Thus a model on which principles are based must define the goals of each of these elements and the relationships between them. The relationships between them must address both the rationale for the relationship, as well as how any limitations in one element affect other elements higher in the hierarchy. Figure 5 provides a model of the relationship between these entities.

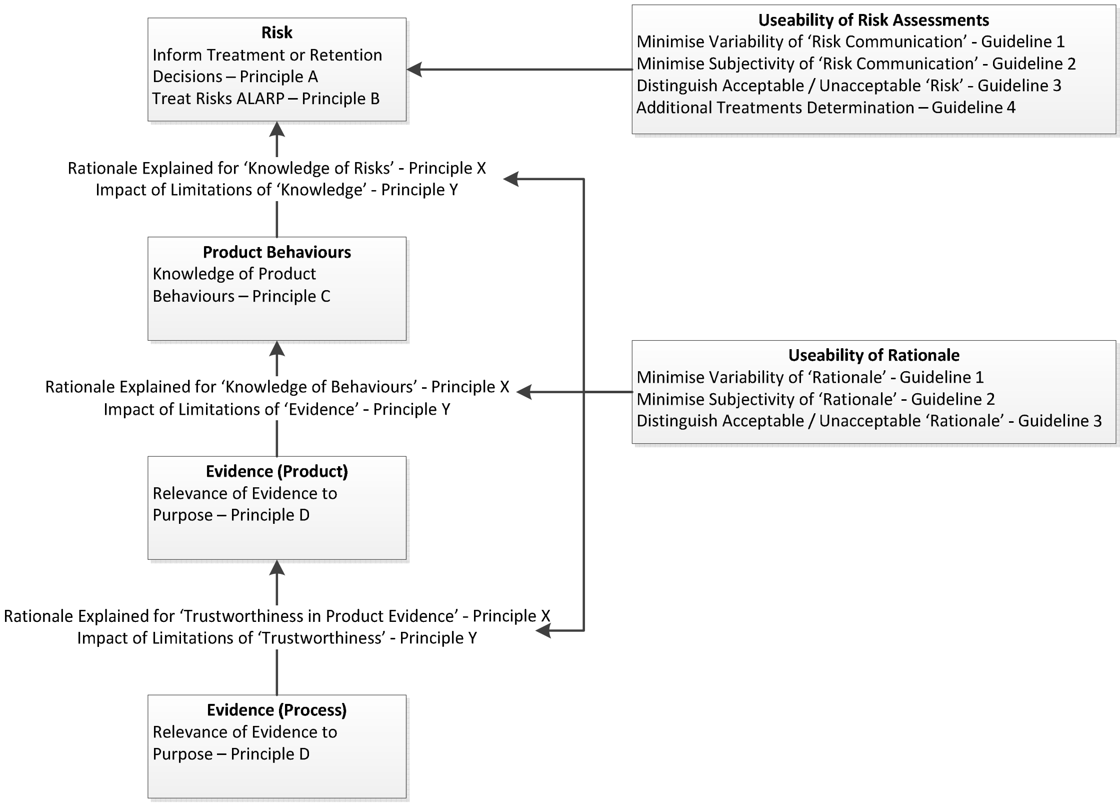

Risk
Inform Treatment or Retention Decisions – Principle A
Treat Risks ALARP – Principle B
Useability of Risk Assessments
Minimise Variability of 'Risk Communication' - Guideline 1
Minimise Subjectivity of 'Risk Communication' - Guideline 2
Distinguish Acceptable / Unacceptable 'Risk' - Guideline 3
Additional Treatments Determination – Guideline 4
Rationale Explained for 'Knowledge of Risks' - Principle X
Impact of Limitations of 'Knowledge' - Principle Y
Product Behaviours
Knowledge of Product Behaviours – Principle C
Useability of Rationale
Minimise Variability of 'Rationale' - Guideline 1
Minimise Subjectivity of 'Rationale' - Guideline 2
Distinguish Acceptable / Unacceptable 'Rationale' - Guideline 3
Rationale Explained for 'Knowledge of Behaviours' - Principle X
Impact of Limitations of 'Evidence' - Principle Y
Evidence (Product)
Relevance of Evidence to Purpose – Principle D
Rationale Explained for 'Trustworthiness in Product Evidence' - Principle X
Impact of Limitations of 'Trustworthiness' - Principle Y
Evidence (Process)
Relevance of Evidence to Purpose – Principle D

Figure 5: Principles and Guidelines of Safety Assurance

The following sub-sections describe the principles/guidelines and their instantiation in the context of this model.

3.4.1 Principles of Entities of Risk, Behaviours and Evidence

Principle A – Safety assurance should inform risk treatment and retention decisions

Description: The primary purpose of safety assurance is to inform risk decisions by identifying, analysing and proposing evaluations[15] of risks to establish if risks are acceptable, tolerable or unacceptable in a given context. This permits duty holders to make informed decisions pertaining to risk treatment or retention. It also informs duty holders of their operational obligations pertaining to management of risk.

Rationale: The authorities responsible for risk treatment decisions should be informed as to when the minimum standard of safety has been achieved and when additional risk treatment is necessary.

Related Purpose Statement: This principle relates to the purpose *require a minimum standard of safety to be achieved*.

Principle B – Safety assurance should prompt stakeholders to treat risks ALARP

Description: Through the systematic identification and analysis of treatment options by relevant stakeholders, the act of undertaking safety assurance should prompt stakeholders to continue to treat risks when the minimum standard of safety hasn't yet been achieved, and where further safety is economic to achieve.

Rationale: Only through risk treatment is it possible to achieve the minimum standard of safety required in a given context. Unacceptable and intolerable risks should be clearly identified as requiring treatment. Where further safety can be achieved above the minimum, safety improvement should be balanced with the cost of safety improvement.

Related Purpose Statement: This principle relates to the purposes:

- *require a minimum standard of safety to be achieved*, and
- *where further safety can be achieved above the minimum, require safety improvement to be balanced with the cost of safety improvement.*

Principle C – Knowledge of behaviours should be established.

Description: Knowledge of behaviours of the system under normal operating and failure circumstances should be established for use in the assessment of risk.

[15] Risk steps (identify, analyse, evaluate) taken from ISO 31000:2009 (ISO, 2009)

Rationale: The behaviours of a product under normal operating and failure circumstances dictate its suitability as a system from both functional and safety perspectives. Although the knowledge of product behaviours will never be absolute, because there is never unlimited time or money to identify and analyse them, by identifying the product behaviours systematically under both normal operating and failure circumstances it is possible to reason about their suitability.

Related Purpose Statement: This principle relates to the purposes:

- *require the achievement of a minimum standard of safety to be informed by a minimum set of information*
- *require a minimum standard of demonstration;*

<u>Principle D – Evidence should be relevant to the rationale for its purpose.</u>

Description: Evidence (both product and process) should be used where it is appropriate, and it should be obvious when evidence is being misused, and this impact on the demonstration captured as a limitation.

Rationale: The role of each piece of evidence should be relevant to its use for providing knowledge of the product behaviours. For example, evidence from white box unit testing has very limited relevance to a claim about requirements validity. Product evidence that is not relevant to its role (in the rationale) results in:

- a limitation in evidence if the role is not fulfilled by another piece of evidence, or
- is counter evidence for process assurance as it indicates that this specific evidence has been used for the wrong purposes, and this may lead to uncertainty regarding the use and role of other evidence.

The set of evidence is never infinite (because there isn't infinite time or money). The way the evidence combines is important for characterising the trustworthiness of product evidence and the product behaviours deduced from it. Misused evidence costs money to produce, and is probably not contributing materially to safety. Costs will only be minimised when unnecessary activities are avoided.

Related Purpose Statement: This principle relates to the purposes:

- *require the achievement of a minimum standard of safety to be informed by a minimum set of information;*
- *require a minimum standard of demonstration;*

- *where further safety can be achieved or demonstrated above the minimum, require safety improvement to be balanced with the cost of safety improvement; and*
- *minimise the cost required to achieve and demonstrate safety.*

3.4.2 Principles Relating to Relational Associations

Linking each entity of evidence, product behaviours and risk in Figure 5 are relational associations. These relational associations need to articulate two types of information, as described by the following two relational principles. Note alternative labelling (X, Y) are used for these principles to distinguish them from the entity principles described above.

Principle X – The rationale should be explained.

Description: The rationale that relates the one entity to another entity should be explained so that achievement of goals of the higher entity can be assessed. The rationale is in effect the underlying argument specific to the relationship between the particular entities. The rationale allows the chain of argument and evidence to continue from evidence, to product behaviours through to risk.

Rationale: The explanation of the rationale between one entity and another entity communicates the achievement relationship, and is fundamental to providing and reasoning about the demonstration. When there are limitations in the rationale, then higher entities may be impacted.

Principle Y – The impact of limitations should be explained

Description: The impact of limitations of one entity on another should be determined and explained in relation to the rationale.

Rationale: No entity is ever without its limitations, either because of physical limitations or because practical cost (resources, time, money) constraints. Knowledge is thus always coupled with uncertainty. Understanding these limitations informs achievement and the confidence in the demonstration. Counter evidence is a powerful indicator of non-achievement. Understanding the impact informs overall achievement and the confidence in the demonstration.

3.4.3 Establishing Usability Guidelines

The purposes of assurance standards identified in Section 3.2 suggest that there are usability guidelines that support the application of principles. This is because humans are required to comprehend and communicate many aspects of safety assurance (i.e. safety assurance had a large human component, no matter how much modelling or how many tools we apply to it). As such safety assurance must have a human factors element to its definition. Section 3.3 provides further confirmation that usability is an important aspect of archiving the purposes for which standards are used. Thus the implementation of the aforementioned principles with respect to the entities of Figure 5 in certification frameworks must be guided by usability guidelines if they are to be practical. There is also evidence that where such usability guidelines do not exist, then many of the limitations identified by Section 2.6 may result. The guidelines will be used to make trade-offs between theory and application in the development of the framework described by this thesis (refer to Chapters 4 through 6). The usability guidelines are defined as follows.

Guideline 1 – Minimise variability

Rationale: Variability of communications of risk, rationale and limitations can lead to difficulties in comprehension. This is because the variability leads to variability of interpretation and potentially inconsistent decision making. Therefore variability should be reserved for circumstances where explicit comprehension of a difference, limitation, decision or action is required.

Guideline 2 – Minimise subjectivity

Rationale: Subjectivity of risk assessment, rationale and limitation information can lead to limitations in the extent to which the information is compelling, or the ease with which agreement can be reached over that information. Therefore subjectivity of information should be minimised by either eliminating subjectivity, or by providing a means within the certification framework for resolving it.

Guideline 3 – Straightforward[16] for {assessor} to distinguish acceptable / unacceptable achievement/demonstration of {outcome}

Rationale: The implementation of the aforementioned principles should permit a ‘reasonable’ assessor within the domain to establish that the outcome is acceptable or

[16] Straightforward does not imply the absence of judgement. It implies that routine judgements, which fall within established airworthiness practice (i.e. the majority of design decisions in airworthiness certifications), are easy to identify. Time spent making judgements should be reserved for genuinely novel solutions or problems.

unacceptable in a manner that is suitable for repeated application and efficiency of resources. Decisions by assessors regarding acceptable/unacceptable outcomes should be consistent amongst average assessors.

Guideline 4 – Straightforward for {decision maker} to determine {action}

Rationale: The implementation of the aforementioned principles should permit a typical ‘reasonable’ decision maker to determine an appropriate action. While the decision will always be the responsibility of the decision maker, all decisions imply there are options, and thus ensuring that a suitable range of options are presented to the decision maker is important.

3.5 The Role of Argumentation for Rationale

In the principles defined by Section 3.4 rationale forms a key part of relating evidence to product behaviours and product behaviours to risk. Contemporary approaches suggest that argument (Kelly, 1998) is a widely used tool for expressing rationale. Argument is evident in many aspects of human society, with argument most prominent in disciplines such as:

- *legal processes of the judicial system*, for which arguments are expressed by the defendant and accuser/prosecutor to relate evidence to each party’s respective version of the truth for evaluation by an independent judge and jury[17];
- *philosophy*, for which some branches utilise argument for expressing relationships between truths and knowledge, beliefs, and theories of justification and reason (Nuttall, 2002); and
- *scientific method*, for which principles of reasoning are applied to evidence for investigating phenomena, formulating new knowledge, or revising and correcting existing knowledge (Nola & Sankey, 2007).

Perhaps less prominent as a discipline or profession, but one to which most people can relate is the ‘debate’ commonly practiced in high school education. Debating is also used during political campaigning as a method of contrasting political policies between differing party representatives. A debate includes affirmative and negative arguments and rebuttals, and is adjudicated by an independent panel (Murphy, et al., 2003).

[17] A jury is only present when a determination of guilt is required. There are legal processes such as a tribunal where a jury is not typically involved (Harris, 2006).

Argument has also come to prominence in safety assurance, with literature such as (Kelly, 1998) and (Hawkins, et al., 2011) developing methods for the development, presentation, maintenance and reuse of safety arguments. However, closer examination of the usage of argument in the prominent disciplines mentioned above and the usage of argument in safety assurance reveals noteworthy differences.

3.5.1 Argument in Legal Process

In legal process, both the defendant and accuser present their respective arguments to the court. While there are some legal forums where the defendant and accuser may not be so explicit, such as in a court of inquiry, there is still debate of the affirmative and negative to capture a range of perspectives on the argument (Harris, 2006). This implies there are essentially two arguments, an argument for and an argument against. These arguments are then subjected to challenging by the opposite party through cross examination, counter argument and rebuttal. This process of challenging the arguments is aimed to establish the relative truth of the respective arguments for deliberation by judge and jury. Hence the role of the argument is not simply the development and presentation of the two arguments, but also the process of challenging the arguments for the visibility of the decision makers.

3.5.2 Argument in Philosophy and Scientific Method

This concept of challenging of arguments also exists in philosophy and scientific method. In philosophy supposed truths and knowledge are often expressed as rhetorical arguments when they are proposed (Nuttall, 2002). In general the nature of philosophy is then that supposed truths only become 'nearer truths' when their falsifiability has resisted enquiring counter-argument by other philosophers. A similar concept applies also to scientific method where all scientific knowledge is the subject of scientific enquiry and the revising and correcting of existing knowledge is prompted wherever evidence of falsification can be reasoned to invalidate previous scientific theory (Nola & Sankey, 2007). Theories that stand up to scientific enquiry by the scientific community will persist and may eventually become acknowledged as scientific laws.

Whether it is legal processes, philosophy or scientific method, a key aspect of assurance of decisions or knowledge is based on the concept of challenging arguments.

3.5.3 Why is Argument in Safety Assurance Different?

However, the same cannot be said for the present usage of arguments in the domain of safety assurance. Literature on safety cases has mostly focussed on the development and presentation of arguments by the developer (e.g. (Kelly, 1998), (Habli & Kelly, 2007), (Weaver, 2003)). There is very limited literature examining the parallels with legal, philosophical and scientific domains for methods for challenging the safety arguments. To some extent is has been expected of assessors in reviewing safety arguments (Kelly, 2007), but with limited guidance and authority. While (Kelly, 1998) identified the concept of anti-patterns as a basis for challenging safety cases, and some further patterns have been developed in (Weaver, 2003), further research of this topic has been limited. The recent literature that has suggested the concept of challenging the safety argument (e.g. (Kinnersly, 2011), (Haddon-Cave, 2009), (Graydon, et al., 2010)), hasn't yet suggested a context in which the challenging should take place that permits it to be effective and efficient in typical certification environments.

(Hawkins, et al., 2011) has proposed that confidence arguments can be used to express confidence in the product argument (refer Section 2.4.4). However, in practice can it be expected to achieve any more than simply to convey a self-fulfilling opinion of high confidence? Other domains suggest that it is the challenging of the opinion that contributes greatly to decision making on outcomes. If the concept of challenging the argument and adjudicating this process is so important to other societal usages of arguments, then does this also imply that safety arguments should be subject to a similar challenging? Further, does an unchallenged safety argument have any basis for being societally compelling?

To answer these questions, it is necessary to consider ways in which the challenging of a safety argument can occur. There appears to be a choice of two general ways to achieve the challenging of the argument:

- have a product development, acquisition and certification process that permits a systematic challenging of the argument on a case by case basis, or
- pre-constrain the argument by challenging the arguments in the process of developing the standards that express those arguments, and require suppliers to conform to those arguments.

The following sub-sections examine these two approaches.

3.5.4 Case by Case Challenging of Safety Arguments

The case-by-case challenging of safety arguments approach is broadly akin to the goal-based approach, but for which methods for challenging of arguments haven't yet been widely addressed beyond those defined by (Kelly, 2007). The legal process, philosophy and scientific methods all offer approaches that provide a means for challenging arguments, but are any of these suitable for the safety argument context?

Is the Legal Challenging Model Feasible?

If the challenging of safety arguments was to adopt a legal process analogy, then there are several interpretations on how the legal process roles could be achieved for a safety argument. One might be that the supplier provides the affirmative argument for achievement and demonstration of safety, an acquirer representative provides the rebuttal argument against safety, and the regulator provides mediation and adjudication. While this interpretation captures the distinct roles of the legal process, it does potentially suffer from conflict of interest that the legal process model does not. For example the acquirer wants or needs the product for some reason, and thus the strength of their challenge may vary dependent on this pressure. (Australian National Audit Office, 2009) and (Haddon-Cave, 2009) suggest that commercial pressures strongly dominate decision making processes.

An alternative that would address this drawback would be to have an independent safety assessor (or similar role) perform the rebuttal argument against safety. The independent safety assessor might be empowered by the acquirer to work with their supplier, subject to agreement by the regulator on their competence to perform the role. In essence the acquirer would pay for two arguments to be developed, with the intent being that both arguments have the opportunity to include counter arguments of the other argument. This would imply a degree of iteration in the development of such arguments that could be perhaps controlled through systems engineering milestone reviews or other such milestones in the project. On the surface, this approach appears to be feasible, but there is a cost impost on the acquirer in that they are paying for two safety arguments to be developed and iterated in their projects. This would seem to work against the purpose of safety assurance that suggests minimising the cost of achievement and demonstration.

Is the Philosophical or Scientific Challenging Model Feasible?

If on the other hand, challenging of safety arguments was to adopt a philosophical or scientific method analogy, then this proposes that safety arguments must be published in the domain of system and safety professionals such that they can be subject to wide scrutiny and challenge by stakeholders and other practitioners. There are several difficulties with this approach. Firstly safety arguments often contain supplier proprietary information and thus suppliers won’t typically authorise wide release. Secondly the process of assembling and collating the open criticisms of the safety arguments may be difficult to control for the regulator because of the number of prospective stakeholders involved. Thirdly there are timeliness goals for the fielding of systems, and thus suppliers will seek assurances that the challenge process can be completed within a bounded period of time. Philosophy and scientific endeavour are not bound by these same timeliness aspirations as is the fielding of systems.

Of the two analogies the legal process analogy seems more feasible as a means of challenging safety arguments. However there are cost implications which work against the cost minimisation goals of the purpose statement.

3.5.5 The Pre-constrained Argument

In some respects pre-constraining the argument is what the traditional prescriptive standards tried to do. However, because of the focus on methods and techniques, they were often unclear about rationale, and thus equally unclear on whether this rationale was being challenged in the process of achieving consensus on the standard.

There have also been attempts to document the rationale behind standards (e.g. RTCA SC-205 SG2 (RTCA Inc., 2012) efforts to document rationale behind DO-178B objectives for the DO-178C revision, (Holloway, 2012)). However, these approaches often reiterate the goals, objectives or sub-objectives of the standard, and don’t explain why (or the rationale for why) these goals or objectives relate to safety.

Presuming though that appropriate arguments could be expressed in standards, and that the process of achieving consensus between stakeholders in development and review of standards effectively constitutes a challenging of the arguments, then it seems feasible to pre-constrain parts of arguments. Stakeholders typically represent suppliers, acquirers and regulators; and so to draw a legal process analogy, this provides a means for both defendants, accuser, judge and jury to witness the challenging of the arguments intended for capturing in the standard. The open circulation of drafts proposals of arguments for review, comment and refuting by stakeholders is also analogous to the philosophical and scientific uses of arguments, albeit constrained in time.

Such a process would avoid the requirement for the arguments (at least the holistic ones) to be subjected to further challenging on a case by case basis, except perhaps periodically when the community suggests the standard requires review because societal or scientific acceptance of those arguments has changed. Whilst the goals of novelty and flexibility for supplier solutions will prevent the entire argument being pre-constrained, specific elements of the argument could be constrained and yet still permit appropriate flexibility in design solutions. Only in the case of entirely novel systems and technologies would pre-constraining the argument not be feasible. This is because there wouldn't yet be the basis of experience from which to form a standards committee.

This pre-constraining approach has the benefit that regulators do not have to spend time challenging core philosophical arguments about how safety can be achieved and demonstrated, and can focus on other aspects of compliance, such as examining the evidence and specific product behavioural attributes. This approach may also be beneficial in domains where regulation is more reactive rather than proactive because it means that suppliers aren't at risk of proposing arguments that have not been subjected to challenge already and that thus would remain unchallenged.

3.5.6 To Pre-constrain Arguments or to Assess Case-by-Case

The more effective approach will depend on several factors. In domains where there are large numbers of certifications involving essentially the same argument pattern for familiar problems and solutions, then to pre-constrain key parts of the arguments by the standards process is probably a more effective method of assuring safety for their domain. This is the vast majority of safety certifications undertaken. Where there are very large numbers of participant suppliers and assessors, the pre-constrained parts of the argument may also offer benefits of reducing variability and subjectivity in arguments. Pre-constraining parts of the argument is also relevant where regulators are reactive or have very limited resources, because the opportunity to review or challenge the argument may be limited. The pre-constrained argument also promises greater cost minimisation from a compliance evaluation perspective than the fully case-by-case assessment, and this may be favourable to regulators and acquirers.

On the other hand, for systems where pre-constraining appropriate elements of the argument isn't practical, because too much of the argument is specific to a novel problem or solution rather than general, then the case-by-case assessment is probably better (e.g. entirely new fields of problem or solution).

This relationship between problems, solutions, novelty and consistency in arguments has been previously articulated by the McDermid Square (Ministry of Defence, 2007). However the existing McDermid Square expresses that the variable is the amount of argument (i.e. minimal, focussed or extensive). Considering the McDermid Square in the context of this discussion reveals that what should be variable is not the completeness of the rationale or argument, but instead the way in which it is expressed.

Adapting the McDermid Square to reflect pre-constrained arguments versus case-by-case arguments results the Modified McDermid Square shown in Figure 6.

Based on this discussion, this thesis will investigate the feasibility of pre-constraining elements of the argument in order to achieve the challenging of safety arguments.

		Solution	
		Familiar	**Unfamiliar**
Problem	**Familiar**	Pre-constrained Argument Evidence assessed with respect to supporting the pre-constrained argument	Adapted pre-constrained argument Evidence assessed with respect to supporting the adapted pre-constrained argument
	Unfamiliar	Adapted pre-constrained argument Evidence assessed with respect to supporting the adapted pre-constrained argument	Case-by-case assessment of argument and evidence

Figure 6: The Modified McDermid Square

3.6 Applying Safety Assurance to Software

Discussion thus far has been predominantly related to safety assurance. It is also necessary to consider the purpose of software safety assurance, and how it may be achieved. Section 3.4 has established principles and guidelines for achieving safety assurance. To consider if the principles and guidelines require any further refinement when applied to the context of assurance of software systems will require an understanding of how software might impact the meta-models defined so far.

To do this, it is important to understand how software might contribute to risks. (McDermid, 2001) identifies that software failures arise most often from:

- discrepancies between documented requirements specifications and the behaviours needed for correct and safe functioning of the system; and
- misunderstandings by software developers about the software's behavioural interface with the rest of the system.

Software-related incidents and accidents have still occurred when the software satisfied its specification and when the operational reliability of the software was perceived to be very high (McDermid, 2001). This is due to:

- requirements that specify behaviour that is not appropriate from a system perspective;
- requirements that do not specify some particular safety behaviour and therefore the developers have made invalid assumptions or omissions about those particular behaviours; or
- software that has unintended (and unsafe) behaviour beyond that which is specified in requirements.

The primary themes that emerge are the suitability of the behaviours of the software, and the knowledge of behaviours. Software events and failures are systematic, and thus the behaviours of any system resulting from behaviours of its software will also be systematic. Inspection of both Figure 3 and Figure 5 reveals that the behavioural properties are identified as a specific aspect of characterising a system. Thus it follows that characterising the behavioural properties of the software is part of characterising the overall system behaviours. Such an observation is consistent with established practice that software itself is not safe or unsafe, but it is the system in which the software resides whose behaviours may be desirable or otherwise.

Because software behaviours are systematic, the rationale relating evidence, product behaviours and risks should also adopt a systematic viewpoint. Hence, a systematic approach of establishing and measuring these properties is sought.

3.7 Conceptual Framework

In this section a framework is outlined which is intended to provide an instantiation of the principles and guidelines outlined within this chapter. The intent of this framework is not to prescribe the specific techniques and evidence that must be used to provide assurance of safety. Instead, the framework identifies the underlying rationale for establishing knowledge and uncertainty of risks, the role of evidence, and gives guidance on how to establish the suitability of evidence. The framework described in this section is conceptual and independent of existing standards and frameworks. The framework is presented diagrammatically in Figure 7, and is summarised by the following sub-sections. Figure 7 also provides references to the chapters and sections within this thesis where specific topics are elaborated.

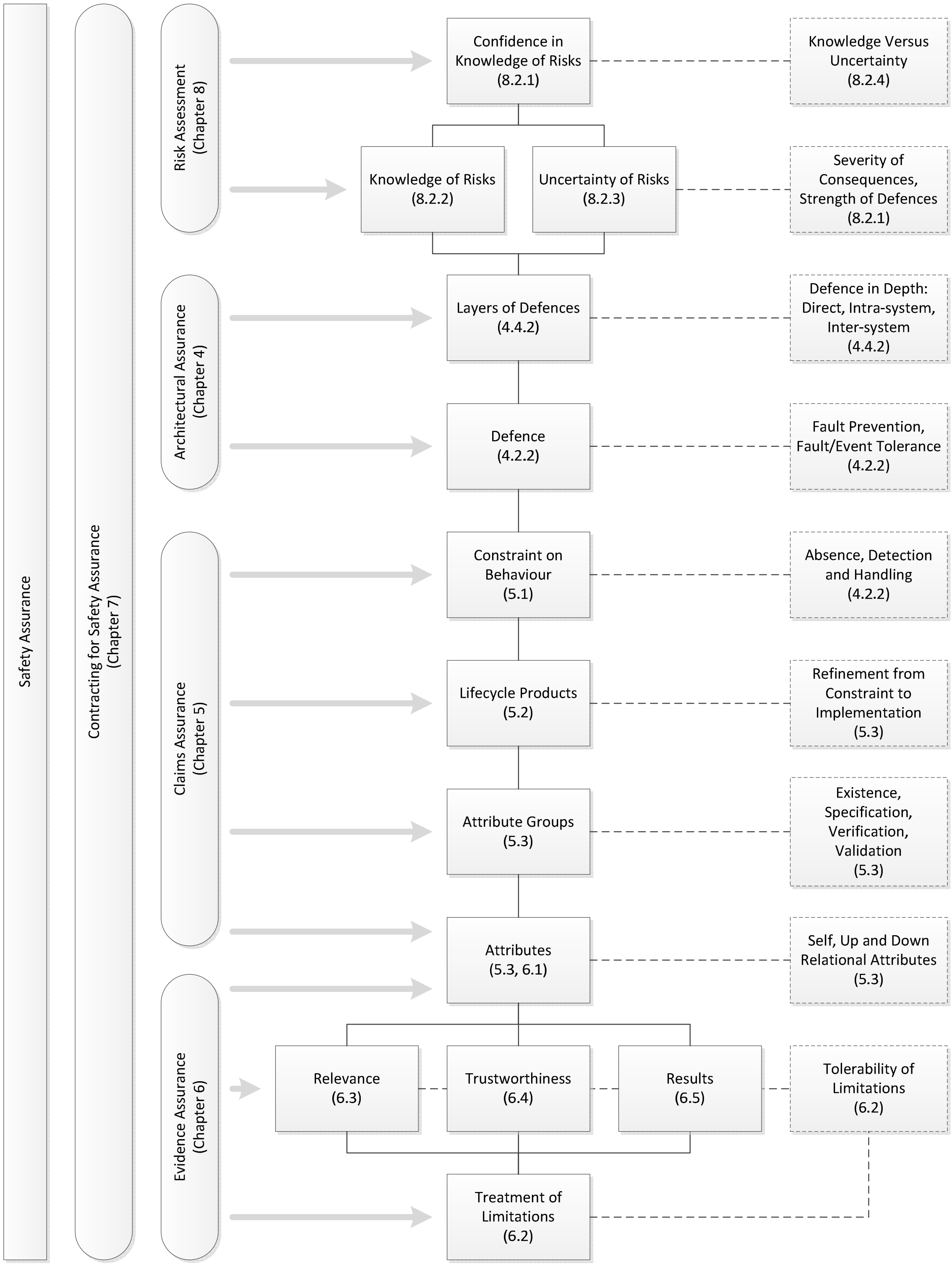

Safety Assurance
Contracting for Safety Assurance (Chapter 7)
Risk Assessment (Chapter 8)
Architectural Assurance (Chapter 4)
Claims Assurance (Chapter 5)
Evidence Assurance (Chapter 6)
Confidence in Knowledge of Risks (8.2.1)
Knowledge Versus Uncertainty (8.2.4)
Knowledge of Risks (8.2.2)
Uncertainty of Risks (8.2.3)
Severity of Consequences, Strength of Defences (8.2.1)
Layers of Defences (4.4.2)
Defence in Depth: Direct, Intra-system, Inter-system (4.4.2)
Defence (4.2.2)
Fault Prevention, Fault/Event Tolerance (4.2.2)
Constraint on Behaviour (5.1)
Absence, Detection and Handling (4.2.2)
Lifecycle Products (5.2)
Refinement from Constraint to Implementation (5.3)
Attribute Groups (5.3)
Existence, Specification, Verification, Validation (5.3)
Attributes (5.3, 6.1)
Self, Up and Down Relational Attributes (5.3)
Relevance (6.3)
Trustworthiness (6.4)
Results (6.5)
Tolerability of Limitations (6.2)
Treatment of Limitations (6.2)

Figure 7: Conceptual Framework Architecture

3.7.1 Risk Assessment

Operational authorities and duty holders (here-after referred to as the 'authority') are required to make decisions regarding treatment and retention of risks during development and in-service operation of systems. The authority must establish that they are confident in their knowledge of risks in order to make effective risk decisions.

Confidence in knowledge of risks is a function of the extent to which knowledge of risks outweighs uncertainty of risks. For an assurance framework, evidence must be presented that shows that knowledge and uncertainty of risks have been characterised.

Taking a systematic perspective on risks means that risks are a function of both their severity of consequences, as well as the technological and operational defences (i.e. strength of defences) that are in place to prevent their manifestation. The stronger the defences against a particular risk, the lower the risk. Evidence should be presented that risk estimates have been informed by characterisation of knowledge and uncertainty.

3.7.2 Architectural Assurance

Defences are a means of constraining the behaviour of a system. They are preventative (fault prevention) or based on tolerance (fault/event tolerance). The suitability of individual defences, and the way multiple defences combine, affects the overall strength of defences. Evidence must be presented that the strength of defences for undesirable behaviours are commensurate with the severity of consequences.

As the designer has numerous options for embodying both preventative and tolerance based defences within the system architecture, evidence must be presented that the architecture implements defences (individually and collectively) in an appropriate way.

3.7.3 Claims Assurance

A defence provides a ‘constraint’ on the behaviour of the system and its software. The requirement on the system for the defence is a ‘constraint’. The translation of a ‘constraint’ into implementation is characterisable by lifecycle products that capture the refinement from requirements, through design, to implementation and the verification and validation thereof. The extent to which the chain of evidence is preserved affects the extent to which knowledge of the ‘constraint’ is achieved and demonstrated.

In addition to establishing the existence of a lifecycle product, evidence can be grouped around the specification, verification and validation of a lifecycle product. Within each group, attributes provide knowledge of the lifecycle product itself (i.e. self attributes), or of the relationships between lifecycle products (i.e. relational attributes). Collective knowledge of these attributes forms the overall chain of evidence.

3.7.4 Evidence Assurance

Suitability of evidence is characterisable by its ‘relevance’ to how it is being used, and the ‘trustworthiness’ of its origins (Weaver, 2003). The results contained within the evidence may also be a source of counter evidence. Therefore, in characterising any specific attribute of a lifecycle product, the relevance, trustworthiness and results[18] of the evidence must be characterised. Where there are limitations in evidence, then the tolerability of those limitations is dependent on how they relate to overall knowledge and uncertainty of product behaviours and resulting risks.

3.7.5 Contracting for Safety Assurance

For a safety assurance framework to be useful in practice, it must be compatible with the contractual arrangements used for acquiring such systems. This thesis examines the topic of military aviation systems, but factors governing usefulness have relevance to other domains also. Enabling the application of the framework and the criteria for ensuring its usefulness are important for real world implementation. A key element of evaluation is establishing that the framework is feasible and useful in practice.

3.8 Introducing the Example - A-DHC-4 Advanced Caribou

To aid the reader in understanding the framework a partially worked example is provided alongside the explanation. The A-DHC-4 Advanced Caribou is a fictional upgrade to the retired Royal Australian Air Force (RAAF) DHC-4 Caribou tactical transport aircraft (Figure 8). The Caribou is a twin-engine high-wing monoplane with full-span double-slotted flaps and reversible propellers, which allow it to achieve steep approaches and short take-offs and landings.

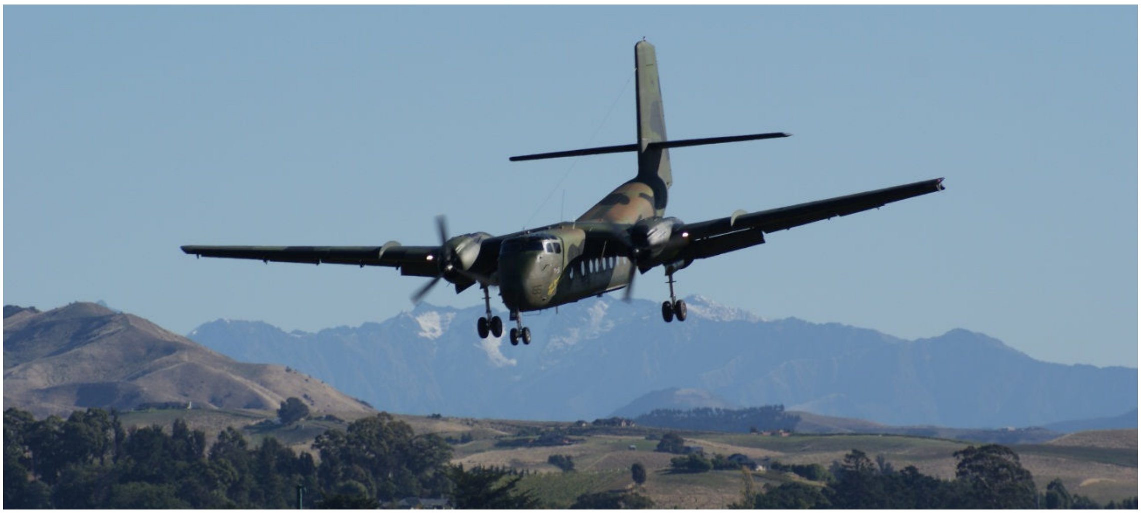


Figure 8: Royal Australian Air Force DHC-4 Caribou (photo by the author)

[18] ‘Results’ encompasses the outcome, meaning, interpretation and/or consequences of the evidence.

This example focuses upon the upgraded digital avionics systems, with specific emphasis on the digital fly-by-wire Flight Control System (FCS). Figure 9 identifies the flight control surfaces which are to be controlled by the FCS.

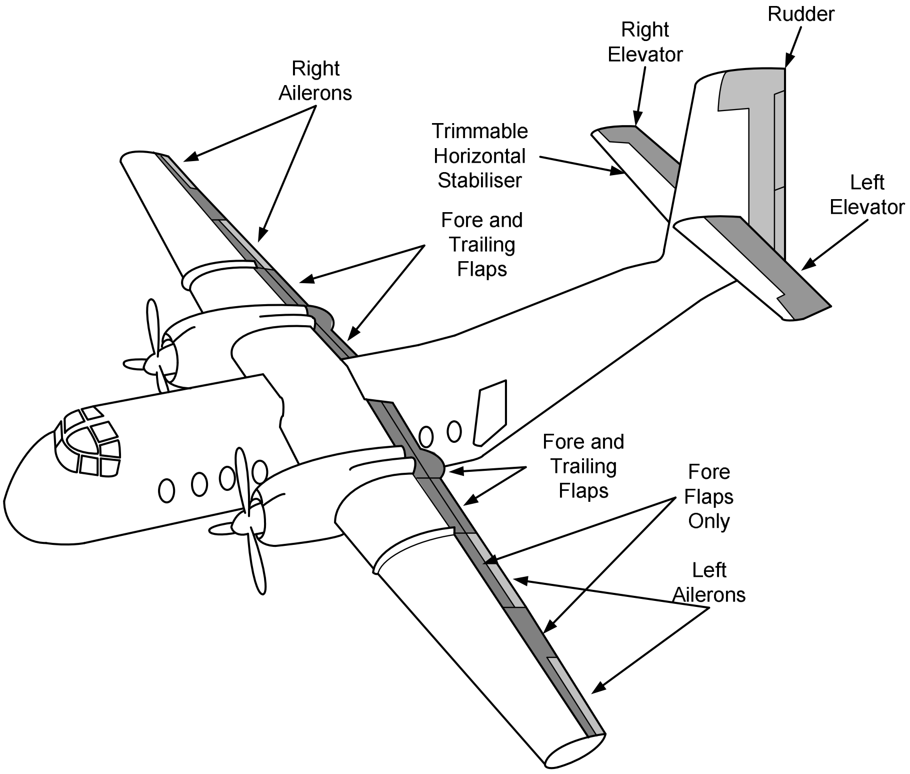


Figure 9: A-DHC-4 Flight Control Surfaces

This example focusses on the FCS. The FCS is an embedded computer system which controls the pitch, roll and yaw. The remaining chapters of this thesis develop this example further to aid in understanding the framework.

3.9 Summary

This chapter has developed principles and usability criteria pertaining to the achievement and demonstration of safety assurance. This chapter has also introduced a conceptual framework for implementing the principles and usability criteria. The framework has been developed from a knowledge and uncertainty of risk perspective, whereby knowledge and uncertainty are characterised from architectural, claims and evidence viewpoints. The aim of this framework is not to prescribe techniques and evidence to be used. Instead, the framework identifies the role of evidence in providing safety assurance, and gives guidance on how to establish the suitability of evidence based on its role in establishing knowledge of product behaviours and risk. The framework provides an evidence-based approach to the risk assessment of a system and its operation.

4 Assuring Against Systematic Failures Using Architecture

Architecture is widely recognised as an important aspect of achieving systems with predictably dependable behaviours. Such dependable behaviours are an essential pre-requisite for controlling risks and thus achieving safety. In Chapter 3, a model containing principles and guidelines was established (refer Figure 5). Chapter 4 examines how architectural properties can be used by an assurance framework to satisfy the principles and guidelines pertaining to product behavioural elements of this model (shown in bold italics within Figure 10).

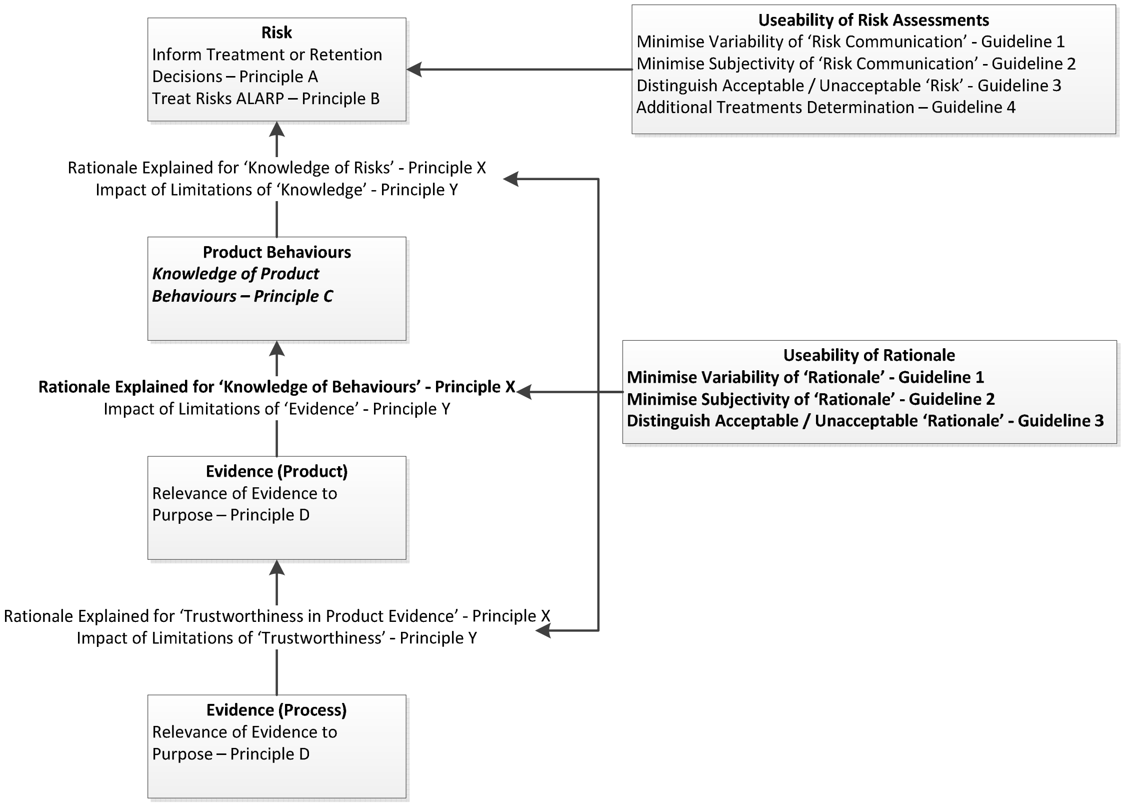

Risk
Inform Treatment or Retention Decisions – Principle A
Treat Risks ALARP – Principle B
Useability of Risk Assessments
Minimise Variability of 'Risk Communication' - Guideline 1
Minimise Subjectivity of 'Risk Communication' - Guideline 2
Distinguish Acceptable / Unacceptable 'Risk' - Guideline 3
Additional Treatments Determination – Guideline 4
Rationale Explained for 'Knowledge of Risks' - Principle X
Impact of Limitations of 'Knowledge' - Principle Y
Product Behaviours
Knowledge of Product Behaviours – Principle C
Useability of Rationale
Minimise Variability of 'Rationale' - Guideline 1
Minimise Subjectivity of 'Rationale' - Guideline 2
Distinguish Acceptable / Unacceptable 'Rationale' - Guideline 3
Rationale Explained for 'Knowledge of Behaviours' - Principle X
Impact of Limitations of 'Evidence' - Principle Y
Evidence (Product)
Relevance of Evidence to Purpose – Principle D
Rationale Explained for 'Trustworthiness in Product Evidence' - Principle X
Impact of Limitations of 'Trustworthiness' - Principle Y
Evidence (Process)
Relevance of Evidence to Purpose – Principle D

Figure 10: Implementing Key Principles of Safety Assurance Using Architecture

Sections 4.1 through 4.6 of this chapter examine the properties of architecture that relate to fail safe design and thus contribute to providing deterministic knowledge of product behaviours. These properties illustrate how architecture can be used to satisfy Principle C. Through examining these properties, the rationale of how architectural properties contribute to knowledge of product behaviours is documented using meta-arguments, thus providing a means to satisfy Principle X. Finally, the rationale for satisfying these principles is explained in Section 4.7, through the definition of an architectural assurance framework, such that it is feasible to adhere to Guidelines 1, 2 and 3.

4.1 Exploring the Role of Architecture

Architecture is an encompassing term for the elements of a system and their interconnections. It is an abstract concept and is dependent on viewpoint and notation. The elements and interactions of a system will lead to a set of behaviours at the architectural level which are more elaborate than simply the collective set of behaviours of the individual elements. It is these additional behaviours (sometimes referred to as emergent behaviours) resulting from the interactions of elements, which give architecture the opportunity to control divergent or unintended behaviours of individual elements. If it is assumed that safety risk is dependent in part on the suitability of behaviours under normal operating and failure circumstances, then architecture can provide a degree of control over the suitability of behaviours, and thus knowledge of architecture is useful for identifying and analysing risks.

Given the opportunity for architecture to control behaviours of a system, then how can architecture be used for providing knowledge of behaviours of a system, and controlling those behaviours with respect to risks? To answer this question, it is necessary to examine how and where architecture is already used in aviation system certification.

4.2 Safety Outcomes of Architecture

(Avizienis, et al., 2004) identifies that faults, errors and failures are threats to the achievement of safety (refer to Section 1.2 for definitions). In this context errors and failures may be the result of both internal faults (e.g. a fault in a software component) as well as external events (e.g. sensors experience a set of conditions that may not be anticipated). Therefore it follows that safety is only achieved when faults, errors and failures are appropriately controlled. This implies that an important property of a system is the suitability of its behaviours in the presence of faults, errors and failures, and their associated propagation and transformation.

This concept is not novel, as existing aviation system design requirements recognise this (e.g. the fail-safe design concept (Federal Aviation Administration, 1988)).

4.2.1 The Concept of Fail Safe Design

The concept of fail-safe design exists within several prominent system safety standards. The two most prominent examples are from the civil aviation certification requirements and the United States Department of Defense safety standards.

Fail-Safe Design in Civil Aviation

Advisory Circular (AC) 25.1309-1A (Federal Aviation Administration, 1988) defines the fail safe design concept as follows:

"In any system or subsystem, the failure of any single element, component, or connection during any one flight (brake release through ground deceleration to stop) should be assumed, regardless of its probability. Such single failures should not prevent continued safe flight and landing, or significantly reduce the capability of the airplane or the ability of the crew to cope with the resulting failure condition."

"Subsequent failures during the same flight, whether detected or latent, and combinations thereof, should also be assumed, unless their joint probability with the first failure is shown to be extremely improbable."

The definition emphasises two key concepts:

- no single failure should prevent safety being achieved, not matter how unlikely the failure is presumed to be, and
- combinations or sequences of failures should also not prevent safety being achieved unless the likelihood of the combinations can be shown to be so incredible that it is virtually impossible.

A similar definition is also shown in the European Acceptable Means of Compliance for the Certification Specifications (CS) 25.1309 rule (EASA, 2011). The consistency of these definitions is indicative of consensus on this concept in the civil aviation domain. Both the (Federal Aviation Administration, 1988) and (EASA, 2011) state that the fail safe design concept implies the application of fault tolerant design approaches including:

- Designed Integrity and Quality
- Redundancy or Backup Systems, Monitors
- Isolation of Systems, Components, and Elements
- Failure Warning or Indication
- Flight Crew Procedures
- Check-ability
- Designed Failure Effect Limits
- Designed Failure Path
- Margins or Factors of Safety
- Fault and Error Tolerance

Interestingly, almost all of these suggested approaches have a bearing on architecture. For example, the design approaches either deal with a property of an element of the system, or a property of the interactions between elements of the system. Hence the fail-safe design concept is in many respects an implementation of the concept of using architecture to control the behaviour of a system in the presence of faults, errors or failures. This concept is not unique to civil aviation systems.

Fail Safe Design for Military Systems

Inspecting the US military safety standards reveals that (US DoD, 2000)[19] defines fail-safe design as:

"A design feature that ensures the system remains safe, or in the event of a failure, causes the system to revert to a state that will not cause a mishap."

(US DoD, 2000) suggests achievement of the fail-safe design through the inclusion of a set of unacceptable and acceptable conditions within the solicitation specification or as contract requirements for the system design. The suggested requirement within (US DoD, 2000) is that *"positive action and verified implementation is required to reduce the mishap risk associated with these situations to a level acceptable to the program manager."* Examples of unacceptable conditions pertaining to failures are provided as follows:

- *"Single component failure, common mode failure, human error, or a design feature that could cause a mishap of Catastrophic or Critical mishap severity"*
- *"Dual independent component failures, dual independent human errors, or a combination of a component failure and a human error involving safety critical command and control functions, which could cause a mishap of Catastrophic or Critical mishap severity"*

Examples of acceptable conditions pertaining to command and control functions are provided as follows:

[19] Note that the fail-safe concepts of (US DoD, 2000) are present in earlier versions of this standard also (US DoD, 1993). The most recent revision (US DoD, 2011), which has yet to see widespread use on projects, has removed this text, although it has been retained within the (G-48 Technical Committee, 2008) document.

- *"For non-safety critical command and control functions: a system design that requires two or more independent human errors, or that requires two or more independent failures, or a combination of independent failure and human error."*
- *"For safety critical command and control functions: a system design that requires at least three independent failures, or three independent human errors, or a combination of three independent failures and human errors."*

(US DoD, 2000) does not limit its suggested unacceptable conditions to those pertaining to failures (as quoted above), but recommends additional classes of acceptable and unacceptable conditions. These additional conditions reflect an instantiation of the *'system safety design order of precedence'*, another concept of the United States military system safety standards. For completeness of understanding, the system safety design order of precedence for mitigating identified hazards is:

a. *"Eliminate hazards through design selection. If unable to eliminate an identified hazard, reduce the associated mishap risk to an acceptable level through design."*

b. *"Incorporate safety devices. If unable to eliminate the hazard through design selection, reduce the mishap risk to an acceptable level using protective safety features or devices."*

c. *"Provide warning devices. If safety devices do not adequately lower the mishap risk of the hazard, include a detection and warning system to alert personnel to the particular hazard."*

d. *"Develop procedures and training. Where it is impractical to eliminate hazards through design selection or to reduce the associated risk to an acceptable level with safety and warning devices, incorporate special procedures and training. Procedures may include the use of personal protective equipment. For hazards assigned Catastrophic or Critical mishap severity categories, avoid using warning, caution, or other written advisory as the only risk reduction method."*

On the other hand, inspection of UK Defence Standard 00-56 Issue 4 (Ministry of Defence, 2007) reveals that it does not contain such explicit fail safe design criteria, either as requirements or guidance. This is a notable difference between the US and UK military paradigms. While its absence from requirements from (Ministry of Defence, 2007) may be a symptom of the more holistic flexibility sought from the application of goal-based concepts by UK Defence Standard 00-56 Issue 4, this does not explain its absence from the guidance within Part 2 of Defence Standard 00-56 Issue 4. The application of the ALARP principle and the consideration of counter evidence may implicitly prompt fail safe design, but it is not explicit. Through the author's involvement with several UK based military projects, there is also anecdotal evidence that the application of ALARP has not meant the application of fail-safe design.

From inspection, the MIL-STD-882 fail-safe design criteria are broadly equivalent to their civil 25.1309 counterpart. Both are explicit in requirements for controlling single failures in circumstances when safety is impacted, and for subsequent or combinations of failures in certain serious circumstances. These themes are important and are suggestive of some more general principles that can be examined in the context of aviation software systems.

4.2.2 Controlling Hazards Caused by Faults, Errors and Failures

(Avizienis, et al., 2004) identifies the means of achieving safety and dependable systems as follows:

- *Fault prevention* – to prevent the occurrence or introduction of faults.
- *Fault tolerance* – to avoid service failures in the presence of faults.
- *Fault removal* – to reduce the number and severity of faults.
- *Fault forecasting* – to estimate the present number, the future incidence, and the likely consequences of faults.

(Avizienis, et al., 2004) distinguishes that fault prevention and fault tolerance aim to provide the ability to deliver a service that is dependable and that achieves safety. Whereas fault removal and fault forecasting aim to provide confidence by justifying that the behaviour specifications are adequate and that the system is likely to meet them. As such, this chapter focusses on fault prevention and fault tolerance.

Fault Prevention

Achievement of fault prevention, and thus each specific fault condition being absent, is shown through the provision of evidence of prevention of the introduction of faults (i.e. show that faults weren't introduced, and therefore they are absent). Examples of such evidence include reviews, inspections and proofs of the correctness of specification of requirements, design and implementation. The main thrust of fault prevention is that there is an absence of faults because development errors were prevented, or detected and removed, and thus failures won't occur. Traditional assurance standards use the concept of putting controls on the development process for requirements, design and implementation to limit the occurrence or introduction of faults/errors, and thus prevent faults. Figure 11 describes a model for fault prevention using the definitions for concepts provided previously in Figure 1 and Figure 3.

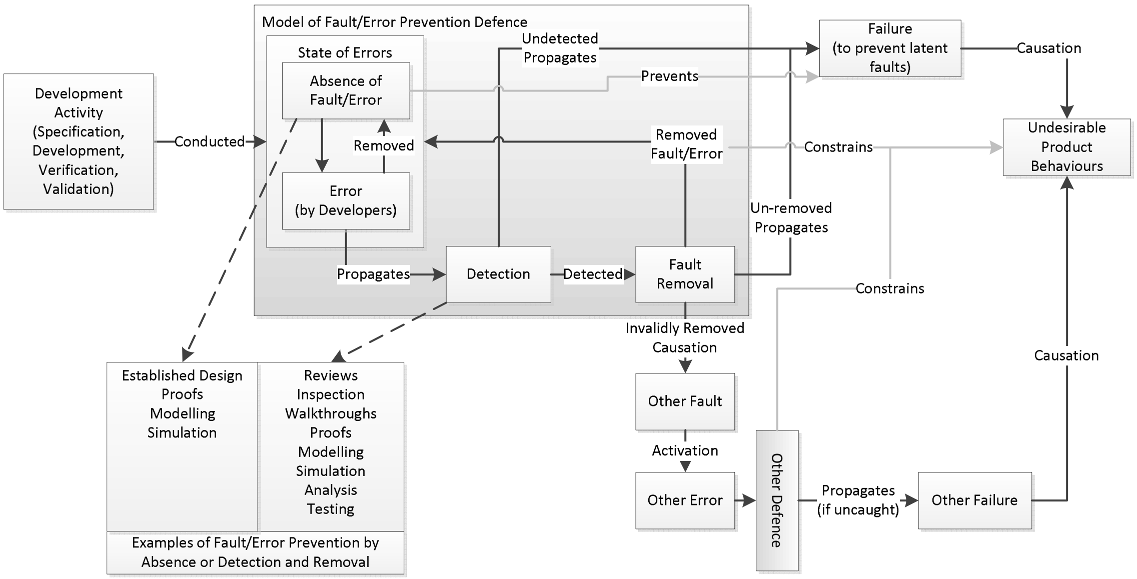


Figure 11: Fault Prevention Defence

In Figure 11, failures are only prevented and undesirable product behaviours constrained when faults and errors are absent. Therefore, a requirement on an assurance framework to address fault prevention should be to reason about fault prevention by an assertion of absence of faults/errors, and the provision of suitable evidence to support the assertion this assertion. This requirement is expressed in GSN as per Figure 12.

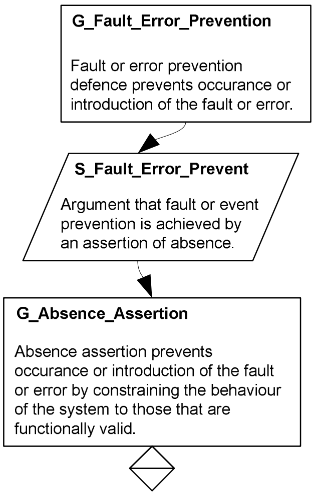

G_Fault_Error_Prevention
Fault or error prevention defence prevents occurance or introduction of the fault or error.
S_Fault_Error_Prevent
Argument that fault or event prevention is achieved by an assertion of absence.
G_Absence_Assertion
Absence assertion prevents occurance or introduction of the fault or error by constraining the behaviour of the system to those that are functionally valid.

Figure 12: Fault Prevention by Assertion of Absence

The child claim G_Absence_Assertion is developed further in Chapter 5. Fault prevention is dependent on knowledge of sources of error, and the failures that might occur in the presence of those errors. If there are limitations in knowledge in this regard, then fault prevention may not in isolation be sufficient to achieve safety. In these cases it may be supplemented with fault tolerance.

Fault Tolerance

Achievement of fault tolerance focuses on showing that system level failures are avoided in the presence of faults. Fault tolerance is the ability for a system to detect an error, fault or failure condition and then undertake a level of reconfiguration/handling to prevent the fault or localised failure propagating to a failure at the sub-system boundary, or a system hazard at the system level. Figure 13 describes a model for fault tolerance using the definitions for concepts provided previously in Figure 1 and Figure 3.

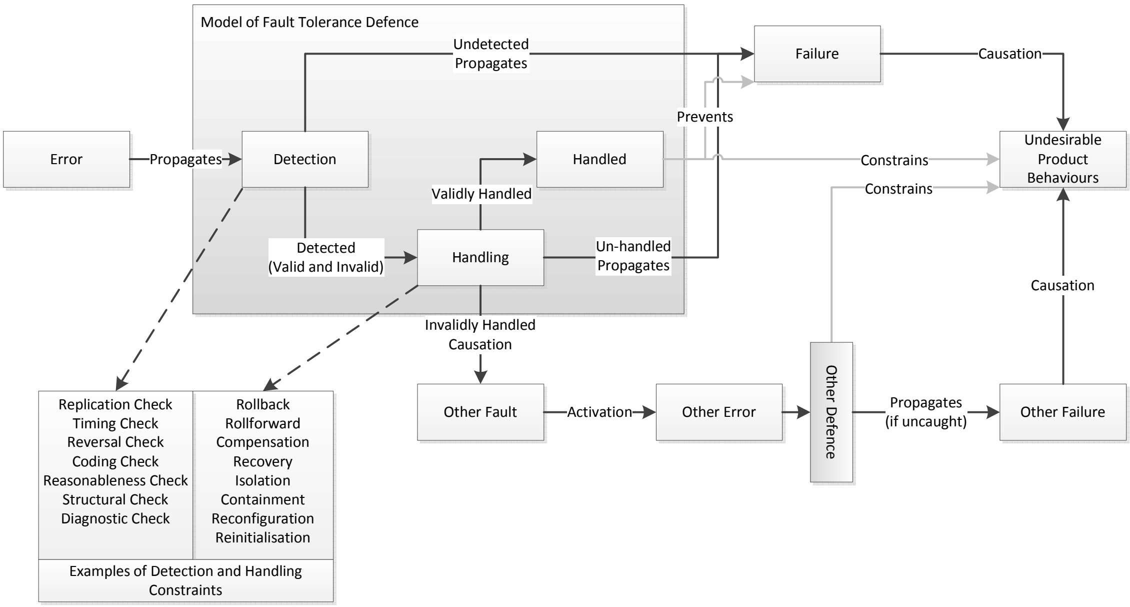
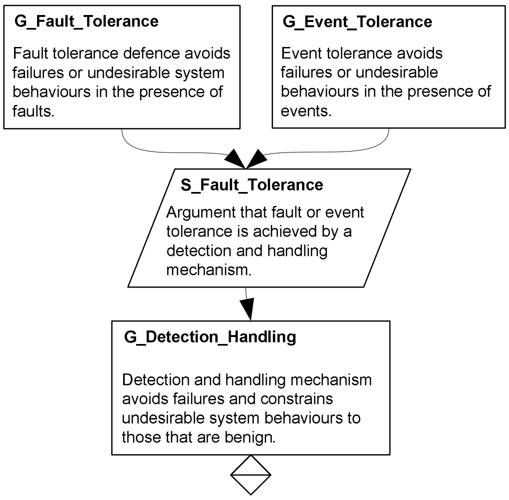


Figure 13: Fault Tolerance Defence

In Figure 13, failures are only prevented and undesirable product behaviours constrained when errors caused by faults and events are detected and handled. Therefore, a requirement on an assurance framework should be to reason about fault tolerance by detection and handling of errors caused by faults/events, and the provision of suitable evidence to support the claim. This requirement is expressed in Figure 14.



G_Fault_Tolerance
Fault tolerance defence avoids failures or undesirable system behaviours in the presence of faults.
G_Event_Tolerance
Event tolerance avoids failures or undesirable behaviours in the presence of events.
S_Fault_Tolerance
Argument that fault or event tolerance is achieved by a detection and handling mechanism.
G_Detection_Handling
Detection and handling mechanism avoids failures and constrains undesirable system behaviours to those that are benign.

Figure 14: Fault or Event Tolerance by Detection and Handling

The child claim G_Detection_Handling is developed further in Chapter 5.

Defence

In both Figure 11 and Figure 13 the concept of a 'Defence' has been annotated, either to label the fault prevention or fault/event tolerance defence. The term 'Defence' will be used widely through this thesis, and is defined as follows:

Defence

A design feature or action intended to prevent faults/events and errors propagating to failures, such that they cannot cause undesirable product behaviours with respect to safety. A defence may be a fault prevention defence, or a fault tolerance defence.

Working Definition

As the 'Defence' refers to the applicable instantiation of fault prevention (Figure 12) or fault/event tolerance (Figure 14), in assuring a claim about the suitability of a defence, it is necessary to present evidence based on the type of defence in each specific instance, as shown in Figure 15[20].

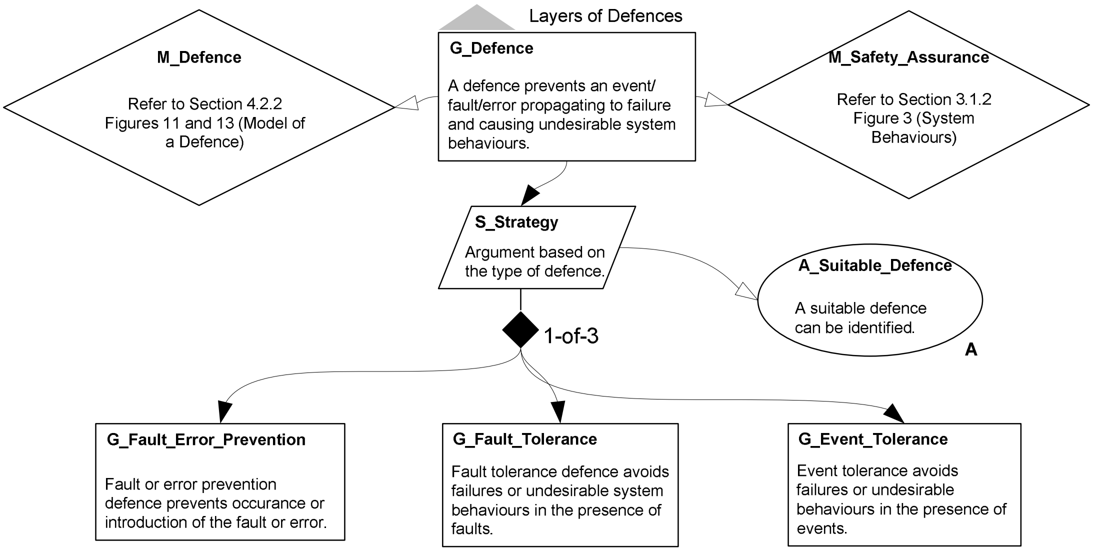

Layers of Defences
M_Defence
Refer to Section 4.2.2
Figures 11 and 13 (Model of a Defence)
G_Defence
A defence prevents an event/ fault/error propagating to failure and causing undesirable system behaviours.
M_Safety_Assurance
Refer to Section 3.1.2
Figure 3 (System Behaviours)
S_Strategy
Argument based on the type of defence.
A_Suitable_Defence
A suitable defence can be identified.
A
1-of-3
G_Fault_Error_Prevention
Fault or error prevention defence prevents occurance or introduction of the fault or error.
G_Fault_Tolerance
Fault tolerance defence avoids failures or undesirable system behaviours in the presence of faults.
G_Event_Tolerance
Event tolerance avoids failures or undesirable behaviours in the presence of events.

Figure 15: Defence pattern

[20] Faults and Events are shown as separate goals to capture their difference in origin. In practice, they can be reasoned about in a similar way, as will be shown by the further development of these goals later in this thesis.

Where Fault Prevent and Fault Tolerance is Performed

Because fault prevention and fault tolerance implies a consideration of preventing faults propagating to system level behaviours, the means (of the process and system) to achieve this are important. The developer has important choices to make regarding how and where these defences are provided. Figure 16 illustrates the places where defences may be used.

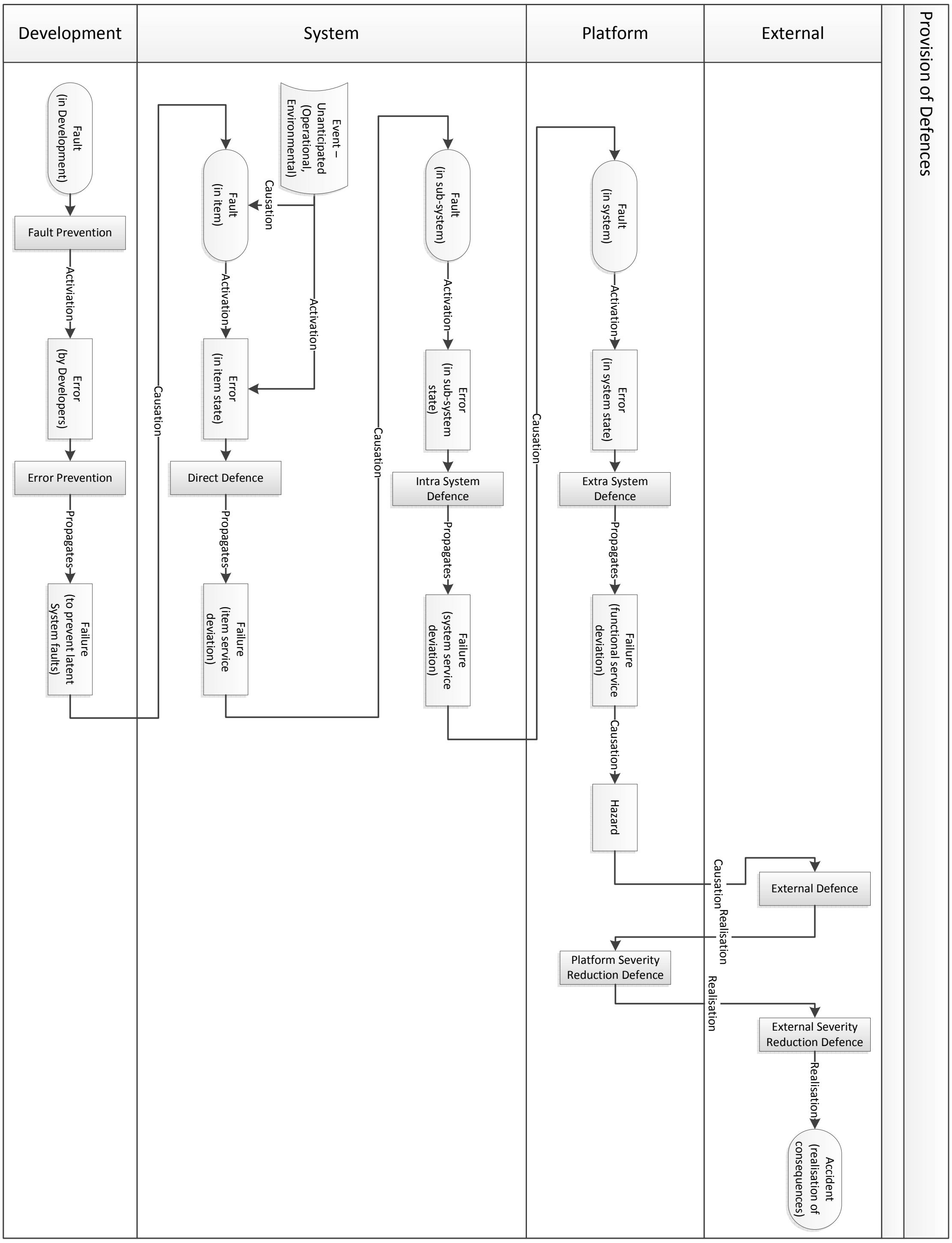


Figure 16: Provision of Defences

Figure 16 suggests that where the ‘Defence’ is implemented is categorised per Table 9.

	Category	Outcome	Perspective	Action	Example
Fault Prevention	*Fault Prevention Defence*	Freedom from fault because the fault isn't activated	Development activities – non-activation	Prevent the fault at its source	modelling and simulation competency from training and experience
	Error Prevention Defence	Freedom from error because the error doesn't propagate	Development activities – non-propagation	Prevent the error at its source	reviews, inspections, proofs a proof that set of possible results does not product/contribute to a causal chain to a hazard
Fault Tolerance	*Direct Defence*	Prevent the originating fault/error propagating to an item failure	Fault perspective – what fault has occurred?	Detection and handling at the source of the fault/error	a reasonability check of inputs or outputs within the controller a reversal check (analytic redundancy) algorithm within the controller for pre-checking computed outputs
	Intra-system Defence	Prevent the item failure propagating to a system failure	System perspective – what item has failed?	Use another sub-system to detect and handle the item's sub-system fault	a monitor channel within a command/monitor architecture LRU with disengagement capability in the event of a fault bit fault reporting and pilot fault management procedures
	Extra-system Defence	Prevent the system failure propagating to a platform level failure	Functional Perspective – what should the system do or not do?	Use another system to detect and handle the item's sub-system fault	an analogue backup for a digital flight control computer advisory, caution and warning system and pilot fault management procedures
	External Defence	Prevent the platform failure propagating to an accident	Platform accident prevention perspective – how can the accident be prevented?	Use external measures to prevent the accident	air traffic advisory information regarding en-route and terminal area weather air traffic controller commands aircraft handling procedures for ground crew to prevent ground crew being injured in prop/turbine line
	Platform Severity Reduction Defence	Reduce the severity of the platform failure	Accident severity – how can the severity of the accident be reduced by the platform?	Use platform-level measures to reduce the severity of the accident	zonal design features on aircraft belly designed to prevent flammable liquid related fire during a wheels up landing 16G seats with seat belt airbags to reduce injury severity
	External Severity Reduction Defence	Reduce the severity of the accident	Accident prevention – how can the severity of the accident be reduced by external measures	Use external measures to reduce the severity of the accident	arrestor cable on airfield to ensure aircraft stops within defined space fire crews in proximity to site at airfield

Table 9: Defence Categorisation

From Figure 16 and Table 9, it is evident that architecture is often used to provide detection and handing mechanisms to achieve fault tolerance. For aviation systems, fault tolerance mechanisms may also be classified as the following:

- *system level fault tolerance* – mechanisms usually provided at a system or line replaceable unit (LRU) level to provide tolerance to sub-system faults (noting that the sub-system fault may be caused by factors internal or external to the system);
- *hardware implemented fault tolerance* – implementation of system level fault tolerance mechanisms by hardware;
- *software implemented fault tolerance* – implementation of system fault tolerance mechanisms by software; and
- *software fault tolerance* – mechanisms provided at software level for containing or mediating software errors, faults and failures.

To provide examples of these mechanisms, Table 10 summarises commonly used fault tolerance mechanisms. These are sourced from (Hammett, 2001) and (Hitt & Mulcare, 2001). Section 4.3 undertakes an examination of actual aviation systems, in which many of these are also evident.

Fault tolerance is dependent on knowledge of forecast faults/events and failures that might occur in order to establish strategies for detection and handling. If there are limitations in knowledge in this regard, then fault tolerance may not be totally effective in achieving safety in the presence of unanticipated faults/events.

To provide practical understanding of how the aforementioned fault tolerance mechanisms are used within actual aviation systems, the following section examines a number of real world aviation systems.

4.3 Examination of Actual Aviation Systems

Fault tolerance mechanisms are remarkably prevalent in critical aircraft systems, suggesting that the architectural benefits of including them in system designs are widely recognised. To illustrate the prevalence of fault tolerance, this thesis examines a number of actual aviation systems, with specific focus on Automatic Flight Control Systems (AFCS) and Flight Management Systems (FMS). The rationale for selecting these types of systems is that they are representative of aircraft systems with moderate to severe failure consequences. Specific attention will also be afforded to those fault tolerance mechanisms that detect or handle systematic faults, as this will be relevant to determining criteria for such mechanisms in later parts of this chapter.

System Level Fault Tolerance	**Hardware-Implemented Fault Tolerance**	**Software Implemented Fault Tolerance**	**Software Fault Tolerance**
• Simplex, no fault tolerance • Simplex, with disengagement features • Dual standby • Self-checking pair (single or dual) • Self-checking pair with simplex fault down • Triple modular redundancy o fault down to self-checking pair or fault down to simplex	• Redundancy • Dissimilar Hardware • Distinct Hardware • Command / • Monitors • Voter Comparators o Average o Middle Value Selection o 2/3 Majority Vote • Watchdog Timers	• Error Detection – recognition of the incidence of a fault o Replication Checks o Timing Checks o Reversal Check (Analytical Redundancy) o Coding Checks o Reasonableness Checks o Structural Checks o Diagnostic Check • Damage Confinement / Fault Containment – restriction of the scope of effects of a fault • Damage Assessment – diagnosis of the locus of a fault • Error Recovery – restoration of a restartable service • Service Continuation – sustained delivery of system services • Fault Treatment – repair of a fault • Distributed Fault Tolerance	• Multi-version software o N-version program o Cranfield Algorithm for Fault Tolerance (CRAFT) o Distinct and Dissimilar software • Recovery Blocks o Deadline mechanism o Dissimilar Backup Software • Exception Handlers o Hardened Kernels o Robust Data Structures and Audit Routines o Run-Time Assertions • Hybrid Multi-version Software and Recovery Block Techniques o Tandem o Consensus Recovery Block

Table 10: Examples of Fault Tolerance Mechanisms

4.3.1 Automatic Flight Control Systems

The AFCS have been examined for the following aircraft types:

- Boeing 777 – Civil Transport Category
- Airbus A330 / Airbus Military KC-30A – Civil / Derivative Transport Category
- Boeing (McDonnell Douglas) C-17 – Military Strategic Air Lift
- Boeing (McDonnell Douglas) F/A-18A/B – Military Air Combat

Appendix A provides a tabulated summary of the architectures of AFCS for these aircraft, and identifies those design features that provide fault tolerance. The information has been obtained from the public domain. Where this has been insufficient, then additional behaviours and treatments have been inferred using flight manuals, pilot briefing notes and maintenance publications. Table 11 identifies the main sources of information used.

Aircraft Flight Control System	**Information Sources**
Boeing 777	(Buus, et al., 1995) (Hornish, 1994) (Yea, 1996) (Yea, 2001) (Bartley, 2001)
Airbus A330 / Airbus Military KC-30A	(Airbus, 1999) (Briere & Traverse, 1993) (Briere, et al., 2001)
Boeing (McDonnell Douglas) C-17	(Kowal, et al., 1992) (Pop & Kahler, 1992)
Boeing (McDonnell Douglas) F/A-18A/B	(Girard & Sharpe, 1999) (Royal Australian Air Force, 2008) (Royal Australian Air Force, 2012)

Table 11: Flight Control System Information

4.3.2 Flight Management, Navigation

The FMS and navigation systems have been examined for the following aircraft types:

- Boeing 777 – Civil Transport Category
- Airbus A330 / Airbus Military KC-30A – Civil / Derivative Transport Category
- Lockheed Martin C-130J-30 – Military Strategic and Tactical Air Lift
- Boeing (McDonnell Douglas) F/A-18A/B – Military Air Combat

Appendix A provides a tabulated summary of the architectures of FMS for these aircraft with the purpose of identifying those design features that might provide fault tolerance. The information has been obtained from the public domain. Where this has been insufficient, then additional behaviours and treatments have been inferred using flight manuals, pilot briefing notes and maintenance publications. Table 12 identifies the sources of information used.

Aircraft Flight Control System	**Information Sources**
Boeing 777	(Driscoll & Hoyme, 1992) (Morgan, 2001) (Uczekaj, 1995) (Witwer, 1995)
Airbus A330 / Airbus Military KC-30A	(Airbus, 1999) (Potocki de Montalk, 2001)
Lockheed Martin C-130J-30	(Royal Australian Air Force, 2005)
Boeing (McDonnell Douglas) F/A-18A/B	(Royal Australian Air Force, 2008) (Royal Australian Air Force, 2012)

Table 12: Flight Management Systems Information

4.4 Observations of Fault Tolerance in Actual Aviation Systems

Analysing the properties of the actual aviation systems leads to observations about the inclusion of fault tolerance within these systems. The following sub-sections describe these observations.

4.4.1 Fault Tolerance for Random and Systematic Faults

There is evidence from the examples of Section 4.3 that fault tolerance exists in these systems for both random and systematic sources of faults. Some types of fault tolerance have been used to provide protection against random and systematic sources of faults, while others only provided protection against a specific type of threat. Hence fault tolerance is relevant to both random and systematic sources of faults, and that software faults can be treated using fault tolerance in addition to fault prevention.

4.4.2 Layered Fault Tolerance

Some sources of fault tolerance have a high degree of fidelity at detection of and handling of faults, while others have a much lower degree of fidelity. There was also evidence that fault tolerance mechanisms may be implemented at item, intra-system and extra-system levels, and that more serious sources of faults were protected against using several fault tolerance mechanisms in a way that resembles layers of defences. This is an important observation, as it provides some confirmation that fail-safe design requires defences against occurrences of combinations of faults. Figure 17 provides a diagrammatic representation of the layers of defences observed. It illustrates the different perspectives each defence type has, as well as the concepts of coverage of propagation paths and defence in depth. Note also the each defence can cause its own faults (as per Figure 13), and this is illustrated in Figure 17 by the alternative fault propagation paths shown.

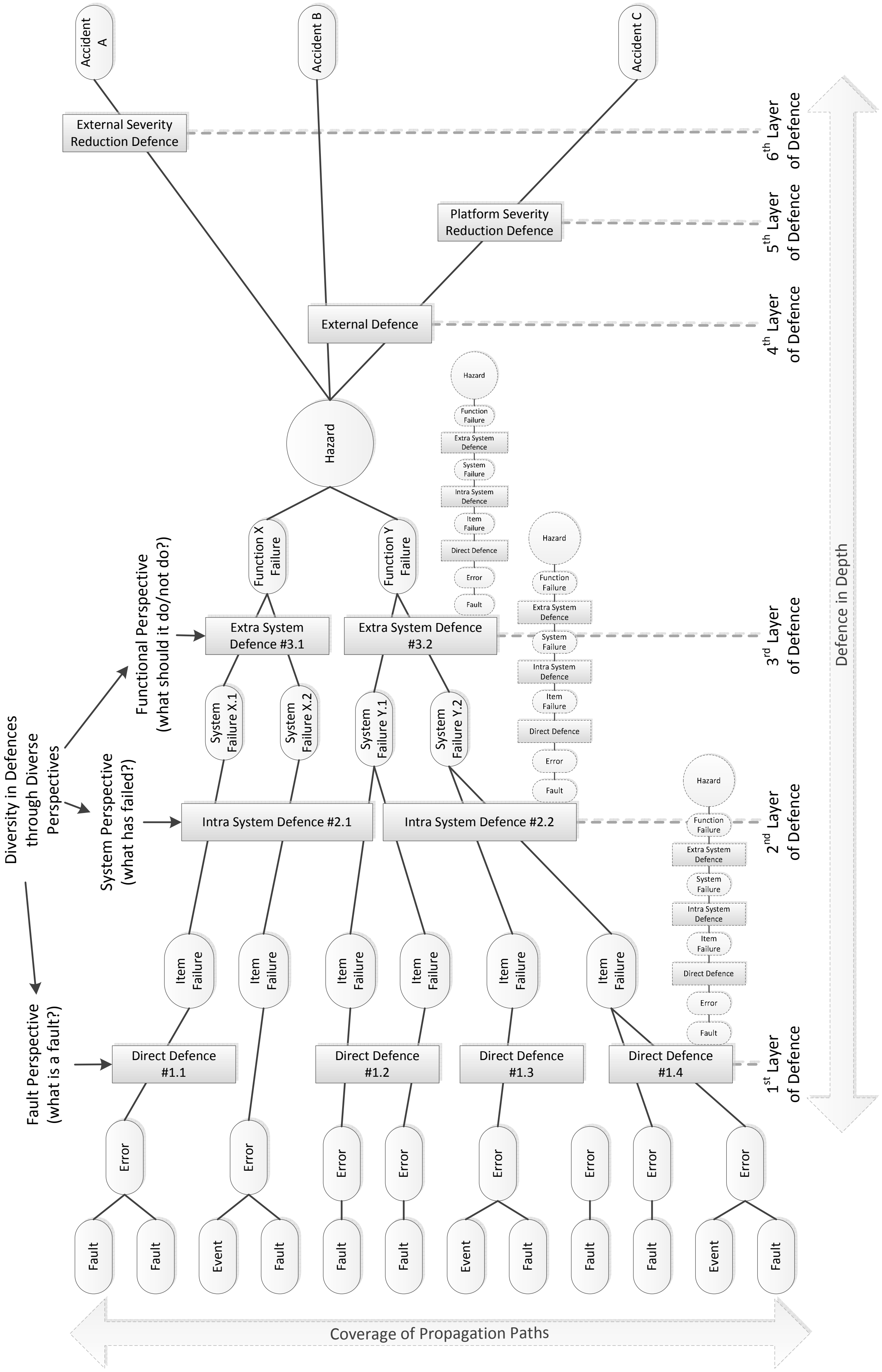


Figure 17: Layered Fault Tolerance Defences and Defence in Depth

In Figure 17, the overall strength of defences is characterised by the defence in depth for each propagation path between the initiating events or faults and the respective hazard and accidents. This property can be expressed as meta-claim as per Figure 18.

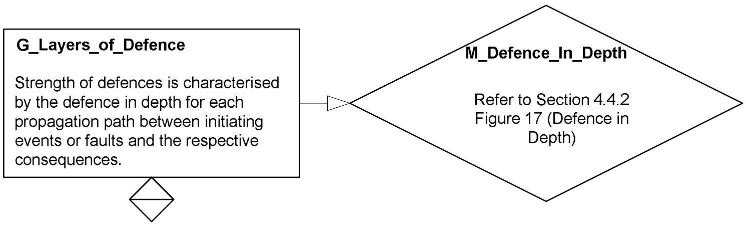

G_Layers_of_Defence
Strength of defences is characterised by the defence in depth for each propagation path between initiating events or faults and the respective consequences.
M_Defence_In_Depth
Refer to Section 4.4.2 Figure 17 (Defence in Depth)

Figure 18: Strength of Defences by Defence in Depth

This meta-claim is developed further in Sections 4.5 and 4.6.

4.4.3 Fault Tolerance from Fail-Safe Design

Based on the architectural reasoning and justification provided within the information sources examined for these systems, the most likely source of the systematic fault tolerance present in aviation systems is from the application of the fail-safe design. The application of the system safety or design assurance standards would seem to be less prevalent in achieving this outcome. Whilst this observation is limited because the study did not have full insight into development, these observations favour the prominence of the fail-safe design criteria. A possible way to reason about this observation is to ask: do developments that apply design or safety assurance, but not the fail-safe design criteria, include equivalent levels of fault tolerance? Those examples provided in Sections 1.1.1 and 1.1.2 suggests that they may not.

This is interesting, since fault tolerance seems to be a compelling goal for design and safety assurance, and yet it isn't prominent within the assurance frameworks, and it doesn't seem to be achieved in isolation of the fail-safe design criteria. This prompts the question: can the fail-safe design criteria be integrated within design and safety assurance frameworks to assure that an equivalent degree of fault tolerance is achieved? The following sections examine this further. Fault tolerance is also fundamentally dependent on knowledge of forecast faults and failures that might occur in order to establish strategies for detecting and handling these faults. If there are limitations in knowledge in this regard, then fault tolerance may not be entirely effective in achieving safety in the presence of unanticipated faults. Therefore, it is important to establish ways of predicting the overall adequacy of the fault tolerance mechanisms within a system's architecture.

4.5 Interpreting the Fail Safe Criteria for Systematic Faults

Since the observations of actual systems suggest that fault tolerance for systematic sources of faults is prompted by the fail-safe design criteria, it is worthwhile to understand how the fail-safe design criteria is interpreted, and whether is requires any refinement to properly address systematic faults and failures. This is a valid question to ask because the fail-safe design criteria from the civil aviation domain includes some probabilistic criteria, which Section 1.2 proposed to be inappropriate for systematic faults and failures. Such consideration is also necessary because the military fail-safe design proposes cardinal quantities of failure combinations, but without rationale as to how these were established, and why?

For the purposes of clarity throughout this discussion, it is assumed that systematic faults can be classified according to (Pumfrey, 1999)'s taxonomy (refer Section 2.4.3), and that treatment strategies include approaches such as those articulated by (Weaver, 2003) (i.e. fault prevention – absence, fault tolerance – detection and handling).

The following sub-sections examine the above questions in detail.

4.5.1 No Single Failure Criterion

The fail-safe design criteria for civil aviation and military systems state the following regarding single failures:

- **Civil Aviation** – *"In any system or subsystem, the failure of any single element, component, or connection during any one flight (brake release through ground deceleration to stop) should be assumed, regardless of its probability. Such single failures should not prevent continued safe flight and landing, or significantly reduce the capability of the airplane or the ability of the crew to cope with the resulting failure condition."*
- **Military Systems (US)** – *"Single component failure, common mode failure, human error, or a design feature that could cause a mishap of Catastrophic or Critical mishap severity categories"* is an *"unacceptable condition"*.

The criteria effectively state that no single failure of a software or hardware component or item should lead to any of the more serious failure circumstances (e.g. Major, Hazardous or Catastrophic using (Federal Aviation Administration, 1988) terminology or Catastrophic or Critical using (US DoD, 2000) terminology).

Therefore when dealing with the presence of systematic faults it must be assumed that:

- any given fault prevention (absence) assertion might be invalidated (due to unknown faults, irrespective of how well assured it might be), or
- any given fault tolerance (detection and handling) mechanism might also be invalidated (due to unknown faults in the detection and handing mechanism, irrespective of how well assured it might be).

It also implies that each of these requires a means to avoid these circumstances preventing the achievement of safety. In essence it implies that there must be additional fault tolerance within the system architecture (elements and interactions) to mitigate these sources of faults. The following paragraphs examine each of these in turn.

Invalidating Fault Prevention (Absence) Assertions

If a fault prevention (absence) assertion is invalidated then only a fault tolerance (detection and handling) mechanism can address the invalidation. This is because once the fault prevention (absence) assertion is invalidated the failure has now occurred and is no longer absent. While the appropriate system architecture might mask that fault at higher levels of abstraction thus making it (or its effects) absent, the system will employ fault tolerance (detection and handling) to achieving this masking.

In typical aviation system architectures, there are several choices available as to where such a subsequent fault tolerance (detection and handling) mechanism might be implemented. These are:

- *direct defence* – downstream in the control and data flow of the same component or item, provided this subsequent detection and handling mechanism isn't violated due to a common mode failure;
- *intra-system defence* – at the typical avionics assembly level or Line Replaceable Unit (LRU) level (hardware and/or software implemented); or
- *extra-system defence* – at the system/platform architecture level, which predominantly concerns itself with the requirements for additional systems, elements or interactions between systems.

Which choice is most appropriate depends on where the fault is best able to be detected (which depends on the type of fault), and also where the fault is best able to be handled. The designer will also have to make architectural choices about how fault tolerance mechanisms can be integrated and combined to provide an architecture with a consistent and yet effective strategy for detecting and handling the totality of all faults, and not just each fault in isolation.

Invalidating Fault Tolerance (Detection and Handling) Mechanisms

If a fault tolerance (detection and handling) mechanism is invalidated then only an additional higher abstraction fault tolerance (detection and handling) mechanism can address the invalidation. The choices for treatment available to the designer are the same as identified above. Because it is implied that any fault tolerance mechanisms could be invalidated, including those that treat the invalidation of other fault prevention assertions or other fault tolerance mechanisms, regardless of whether they have in fact been invalided, then even the no single failure criterion suggests that layers of fault tolerance may be required. The requirement for layers will become more evident when the combinations of failure criterion is examined.

It is also important to note that it only takes either the detection OR the handling mechanism to be invalidated to invalidate the effectiveness of the whole detection and handling mechanism. Detection may be at a different level of abstraction to the handling – although most often handling is at the same or higher level than the detection feature.

Impact of the No Single Failure Criterion

The no single failure criterion therefore places constraints on the structure of the rationale for treatment of the more serious failure circumstance (i.e. Major through to Catastrophic failures). Table 13 identifies the effect of these constraints on the fault prevention and fault tolerance strategies typically necessary for any given failure mode. It also suggests which level of architectural abstraction is typically used to treat the failure mode. Note that the previous paragraphs have focussed on the more serious failure circumstance (i.e. Major, Hazardous and Catastrophic failure conditions). Implicitly the no single failure criterion also implies that a single failure can acceptably lead to a Minor or No Safety Effect failure condition. Thus it is also possible to represent these failure conditions in Table 13. The Item columns in Table 13 (columns B[21] and D) both refer to the same configuration item. This is because it is possible to provide an initial fault prevention (absence) assertion or fault tolerance (detection and handling) mechanism (column B) and then provide the subsequent fault tolerance (detection and handling) capability at a later point in the functional flow, or architecturally (column D). The second Item column (column D) should be interpreted

[21] Column identifiers are alphabetic values starting at Column A for the Severity Column.

as a separate configuration item, perhaps resident in a monitor for example. This is considered at the intra-system (LRU) level.

Severity	**Direct (Item$^{\%}$)**		**Direct (Item)**		**Intra-System (LRU)**		**Extra-System**
Catastrophic, Hazardous / Major	Absence (Primary, Secondary, and Control)	**AND**	Detection AND Handling*	OR	Detection AND Handling	*OR*	Detection AND Handling
			Detection*	AND	Handling	*OR*	Handling
			-	-	Detection	*AND*	Handling
	Detection AND Handling	**AND**	Detection AND Handling$^{\#}$	OR	Detection AND Handling	*OR*	Detection AND Handling
			Detection$^{\#}$	AND	Handling	*OR*	Handling
			-	-	Detection	*AND*	Handling
Minor, No Safety Effect	Absence (Primary, Secondary, and Control)	**OR**	-	-	Detection AND Handling	*OR*	Detection AND Handling
			-	-	Detection	*AND*	Handling
	Detection AND Handling	**OR**	-	-	Detection AND Handling	*OR*	Detection AND Handling
			-	-	Detection	*AND*	Handling

% - initiating fault invalidates this assertion under no single failure criterion

* - provided invalidating the fault prevention (absence) assertion doesn't also lead to invalidation of its detection and handling by the item

# - provided invalidating the original fault tolerance (detection and handling mechanism) doesn't also lead in invalidation of its detection and handling by the item

Logical conventions: Logical operators and conditions within the same cell assume parenthesis

Italics – evaluate the logical operator first – assume parenthesis encapsulates the cells either side

Bold – evaluate the logical operator last (after italics and normal type face operators)

Table 13: No single failure criterion implications for fault prevention and fault tolerance

To illustrate the intent of Table 13, a simple example will be considered. Consider a system with a catastrophic failure condition. Table 13 implies that there are two approaches the system designer could use to address the no single point of failure criterion. The first is to assure both the absence (Table 13 row 2[22], column B) or direct/immediate detection and handling (Table 13 row 3, column B) of the initiating fault and provide a supplemental detection and handling mechanism at a downstream direct (column D), intra-system (column F) or extra-system level (column H). Thus at least two failures are required to realise the catastrophic failure condition.

However, this is only the first criterion we need to examine, the next section considers further constraints on these identified effects, and will likely further constrain Table 13.

[22] Row identifiers are positive integer values starting at Row 1 for the Heading row of Table 2.

4.5.2 Combinations of Failure Criterion

The fail-safe design criteria for civil aviation and military systems state the following regarding combinations of failures:

- **Civil Aviation** – *"Subsequent failures during the same flight, whether detected or latent, and combinations thereof, should also be assumed, unless their joint probability with the first failure is shown to be extremely improbable."*
- **Military Systems (US)** – *Unacceptable conditions are defined as follows:*
  - *"Dual independent component failures, dual independent human errors, or a combination of a component failure and a human error involving safety critical command and control functions, which could cause a mishap of Catastrophic or Critical mishap severity categories"*
  - *"For non-safety critical command and control functions: a system design that requires two or more independent human errors, or that requires two or more independent failures, or a combination of independent failure and human error."*
  - *"For safety critical command and control functions: a system design that requires at least three independent failures, or three independent human errors, or a combination of three independent failures and human errors."*

Both criteria effectively state that credible combinations of failure of a software or hardware item shouldn't lead to any of the more serious failure circumstances. However, the means for establishing how many combinations are necessary differs. The civil aviation approach uses the threshold that the joint probability is Extremely Improbable, whereas the military systems approach specifies ordinal numbers directly. Do these two different measures imply the same thing, or do are they different?

Extremely Improbable can never be defensibly argued for any single component. Instead Extremely Improbable is reasoned by combining sequences of event likelihoods which are in isolation more likely than Extremely Improbable. For example, using terminology from (Federal Aviation Administration, 1988), the following combinations of failure likelihoods are often used:

- Extremely Remote AND Remote,
- Extremely Remote AND Probable,
- Remote AND Remote, or
- Probable AND Probable AND Probable.

All these statements are based on the presumption of independence between elements of the design. It is this criterion that leads to triple redundancy in civil aviation systems with catastrophic failure conditions and dual redundancy in most civil aviation systems with hazardous or major failure conditions (as is apparent in the aviation systems examined in Section 4.3, and observations made by (Edwards, et al., 2010)). However for items with systematic failure modes (e.g. software, complex electronic hardware), probability and likelihoods have traditionally had little relevant meaning (refer to Section 1.2). Therefore it is necessary to resolve an equivalent interpretation that doesn't use probabilities.

One way is to speculate that the joint likelihood of no two combinations of systematic failures can ever be demonstrated to be commensurate with extremely improbable. This hypothesis, based on examination of actual systems, is that the burden of demonstrating this level of knowledge of the system or the stochastic model would generally be unattainable. Section 4.6 provides further discussion on knowledge and uncertainty. Therefore, no two systematic failures should lead to a catastrophic failure condition. Interestingly, this is equivalent to the military systems (US) approach from (US DoD, 2000), which reasons that dual failure combinations are insufficient for the catastrophic case.

This implies that there is at least sufficiently independent fault tolerance (detection and handling) of the initiating failure mode within the item itself, and at the intra-system level or extra-system architecture level for catastrophic failure conditions. For Major and Hazardous failure conditions, two independent failures may be tolerable (because the consequences would require another event to realise a catastrophic failure condition). The independence is most practically achieved by detecting and handling the faults/events at a level outside the item. Overall, the outcome is broadly comparable to the outcome for probabilistic hardware failure assessments. It is also supported by the observations made on the examination of the aviation systems discussed in Section 4.3, and observations made by (Edwards, et al., 2010).

For combinations of three systematic failure modes, it may be possible to reason that they are Extremely Improbable, provided there is fault tolerance (detection and handling) of item failure modes outside of the item in question (i.e. at either the LRU level or system architecture level). With each layer of fault tolerance (detection and handling) mechanisms, the burden of demonstrating this level of knowledge of the system or the stochastic model is more attainable. This is because the opportunity to generalise the detection and handling of classes of faults becomes more tractable. Again the outcome is broadly comparable to the outcome for hardware failures, and is supported by the examination of the aviation systems discussed in Section 4.3, and observations made by (Edwards, et al., 2010). Section 4.6 examines the effects of layered fault tolerance (detection and handling) mechanisms on uncertainty in the stochastic model in more detail. The following paragraphs consider the implications of combinations of failure in more detail.

Invalidating the Fault Prevention (Absence) Assertion and the Fault Tolerance (Detection and Handling) Mechanism – Catastrophic Only

In this case we invalidate the fault prevention (absence) assertion, but also the fault tolerance (detection and handling) mechanism that provides the treatment to the invalidation of the absence assertion. This leads to it being necessary to detect and handle the failure mode outside of the item – either at the intra-system (LRU) level (e.g. through a monitor) and/or by the extra-system architecture level (e.g. through combinations of redundancy, analogue backup, diverse system components, etc.), or both. These circumstances are supported by observations from the examination of the aviation systems in Section 4.3.

Impact on Argument of Combinations of Failure Criterion

The combinations of failures criterion therefore places further constraints on the structure of the argument for serious failure circumstance (i.e. Major, Hazardous and Catastrophic failure conditions). Table 14 identifies the effect of these constraints on the number of treatments required for any given failure mode – and at what level within the system the failure mode is typically mitigated, as determined from the aviation systems considered in Section 4.3 and observations made by (Edwards, et al., 2010). Note that severity is the accident effect if the intra-system (LRU) level and extra-system level mechanisms were absent or the item fault was permitted to propagate without intervention at the intra-system or extra-system levels.

Severity	**Direct (Item$^{\%}$)**	**Direct (Partitioned Item)**			**Intra-System (LRU)**		**Extra-System**
Catastrophic	Absence (Primary, Secondary, and Control)	**AND**	Detection AND Handling$^{\&*}$	*OR*	Detection AND Handling$^{\&}$	AND	Detection AND Handling$^{\&}$
			Detection AND Handling$^{\&*}$	AND	Detection AND Handling$^{\&}$	*OR*	Detection AND Handling$^{\&}$
	Detection AND Handling	**AND**	Detection AND Handling$^{\&\#}$	*OR*	Detection AND Handling$^{\&}$	AND	Detection AND Handling$^{\&}$
			Detection AND Handling$^{\&\#}$	AND	Detection AND Handling$^{\&}$	*OR*	Detection AND Handling$^{\&}$
Hazardous, Major	Absence (Primary, Secondary, and Control)	**AND**	Detection AND Handling$^{\&*}$	*OR*	Detection AND Handling	*OR*	Detection AND Handling
	Detection AND Handling	**AND**	Detection$^{\&\#}$	AND	Handling	*OR*	Handling
			-	-	Detection	*AND*	Handling
Minor, No Safety Effect	Absence (Primary, Secondary, and Control)	**OR**	-	-	Detection AND Handling	*OR*	Detection AND Handling
			-	-	Detection	*AND*	Handling
	Detection AND Handling	**OR**	-	-	Detection AND Handling	*OR*	Detection AND Handling
			-	-	Detection	*AND*	Handling

% - initiating fault invalidates this argument under no single failure criterion
& - additional faults may invalidate these arguments under combinations of failure criterion
* - provided invalidating the absence argument doesn't also lead to invalidation of its detection and handling in software
# - provided invalidating the original detection and handling argument doesn't also lead in invalidation of its detection and handling in software
Logical conventions: Logical operators and conditions within the same cell assume parenthesis
Italics – evaluate the logical operator first – assume parenthesis encapsulates the cells either side
Bold – evaluate the logical operator last (after italics and normal type face operators)

Table 14: Combinations of failure criterion implications for fault prevention and fault tolerance

Specific Circumstances for Fault Prevention (Absence) Assertions

Absence assertions (for omission, commission, early, late and value) should never be valid for input data (i.e. data originating outside the item of the LRU, e.g. from a sensor) to the item within an LRU. This is because the item has no control over the validity of this information. These types of faults are better detected and handled at the input to the item, as is evident in many aviation systems; or by ensuring that the fault propagates to a detectable fault at a higher system level. Detection will usually need to be more extensive than simply checking the valid flag provided with the data from the sensor because this doesn't provide detection of timing or omission related failures, and because the valid flags coverage of credible value failures is often very limited. There are instances (e.g. Qantas QF72 07 Oct 2008 (Australian Transport Safety Bureau, 2008)) where a sensor doesn't only 'lie' about the value it is providing, but it also 'lies' about the validity of that value with its valid flag. Typically a combination of range, rate, physical world model checks, or comparison to redundant or diverse sources is required.

While the fault tolerance (detection and handling) of this class of faults may be deferred until later in the system functional flow, this is rarely suitable. For example, in Flight Control Systems, there are minimal benefits to processing control laws based on invalid input data and then attempting to trap the failure at the system's output or control actuator. Flight control systems that have adopted this strategy have shown it to be problematic. This is because the vast majority of input data failures are not easily discernible at this point in the system. The only times it might be suitable is if through physical limiting (e.g. mechanical limiting) the Flight Control System's authority is limited to a worse credible failure severity of minor, which is clearly not applicable to full authority systems, or systems where limited authority cannot be guaranteed across the flight envelope.

4.6 Using Architectural Fault Tolerance to Bound Uncertainty

The rationale for proposing the layering of fault tolerance (detection and handling) mechanisms at different levels of abstraction in a system (e.g. direct (item) level, direct (partitioned item level), intra-system (LRU) level, and extra-system level) is that it permits the uncertainty associated with detecting and then providing a suitable handling response to the fault to be bounded to an amount that is useful for reasoning about knowledge and the safety of the system. This section examines how architecture is used to bound uncertainty.

4.6.1 Using Architecture to Bound Uncertainty

To examine the effect of architecture on uncertainty, consider a series of cascading faults in a system with fault tolerance (detection and handling) mechanisms at the direct item (software/hardware), inter-system (LRU) and extra-system levels.

<u>Examining the 1st Fault</u>

At the occurrence of the 1st fault at the item level (i.e. invalidation of the 1st fault prevention (absence) assertion or fault tolerance (detection and handling) mechanism), the knowledge is a function of the following:

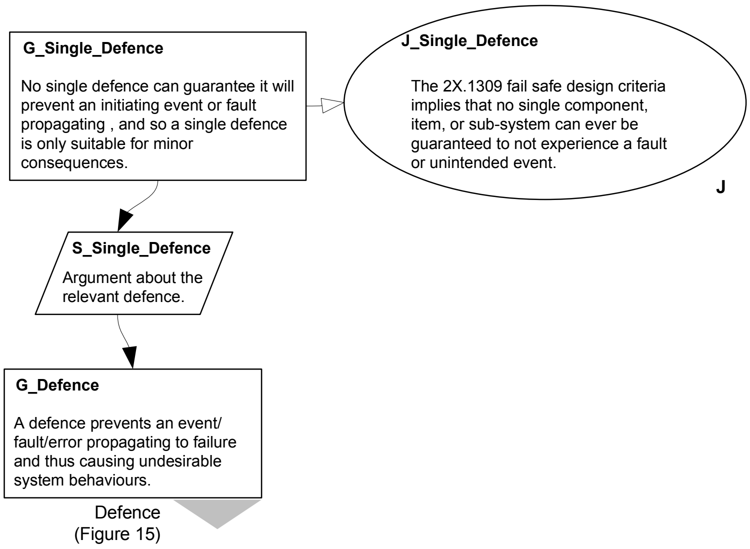
- the understanding of types of failure that might occur (i.e. to what extent is an appropriate mechanism provided to achieve coverage of all classes of the taxonomy of potential item failure modes, noting that the lower the level faults are examined at, the greater the fidelity of faults to be considered); and

- the appropriateness of fault prevention (absence) or fault tolerance (detection and handling) mechanisms given the specific fault under consideration.

Conversely, the uncertainty is a function of the following:

- the extent to which the taxonomy of potential item failures modes is incomplete for the specific failures that could occur in the system (i.e. are there sources of failure that haven't been understood?);
- the effect of failure sources that haven't been understood (i.e. is the effect something that has been left unanticipated, even in a generalised sense?); and
- the suitability of the extant fault prevention (absence) or fault tolerance (detection and handling) mechanisms for these unknown sources of failure (i.e. will it do something undesirable in the presence of an unknown fault?).

Therefore, for the 1st fault with no additional fault tolerance (detection and handling) mechanisms, uncertainty is difficult to bound. Even if a fault tolerance (detection and handling) mechanism is employed, the ratio of uncertainty to knowledge may be large depending on the extent of the fault coverage by the mechanism. This poses problems for failures with severe consequences, but may be suitable for failures with minor consequences. For minor consequences, the requirement for evidence showing the prevention or tolerance of such faults/errors should be such that occurrences of these consequences does not undermine the system dealing with more serious consequences. Therefore a single defence will usually be suitable, as shown in Figure 19.



G_Single_Defence

No single defence can guarantee it will prevent an initiating event or fault propagating , and so a single defence is only suitable for minor consequences.

J_Single_Defence

The 2X.1309 fail safe design criteria implies that no single component, item, or sub-system can ever be guaranteed to not experience a fault or unintended event.

S_Single_Defence

Argument about the relevant defence.

G_Defence

A defence prevents an event/ fault/error propagating to failure and thus causing undesirable system behaviours.

Figure 19: Single Defence

For more severe consequences, additional propagation of faults needs to be examined.

Examining the 2nd Fault

At the occurrence of the 2nd fault, this time at the intra-system (LRU) level (i.e. failure of the 2nd fault tolerance (detection and handling) mechanism), the knowledge is a function of the following:

- the extent to which the taxonomy of failures should resolve the failures of the 1st mechanism, which should be finite at this level (the existence of the detection and handling mechanism is explicitly having to detect the consequences of the failure of the 1st mechanism);
- the degree to which it is possible for the 2nd detection and handling mechanism to be activated from the cascading fault condition;
- the coverage of intended coupling paths between software and LRU level mechanisms; and
- the appropriateness of the detection and handling mechanisms at the LRU given the specific known fault class that has occurred (i.e. is the behaviour of the mechanism valid at this level of abstraction).

The uncertainty is a function of the following:

- the extent to which the cascading faults don't resolve to the taxonomy of faults handled at this layer;
- the suitability (or unsuitability) of detection and handling mechanisms for unknown sources of failure, and its effects; and
- the extent to which unintended independence violators might be active (but should be limited by the degree of physical partitioning).

With two (2) layers of protection, uncertainty may be significant, but it is likely to be much less and may be much easier to bound depending on the extent to which the classes of cascading faults resolve to the taxonomy at the second layer. Therefore a system with two layers of protection may be reasoned as suitable for any system except for those with the most severe failure modes, provided suitable protections are employed at each layer. This can be expressed as shown in Figure 20.

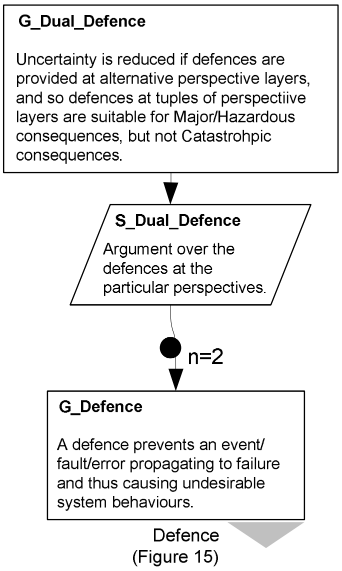

G_Dual_Defence
Uncertainty is reduced if defences are provided at alternative perspective layers, and so defences at tuples of perspectiive layers are suitable for Major/Hazardous consequences, but not Catastrohpic consequences.
S_Dual_Defence
Argument over the defences at the particular perspectives.
n=2
G_Defence
A defence prevents an event/ fault/error propagating to failure and thus causing undesirable system behaviours.
Defence (Figure 15)

Figure 20: Dual Defence

For the most severe consequences, another fault occurrence needs to be examined.

Examining the 3rd Fault

At the occurrence of the 3rd fault, this time at the extra-system level (i.e. failure of the 3rd fault tolerance (detection and handling) mechanism), the knowledge and uncertainty parallel the observations listed above for the 2nd mechanism, with the following key difference:

- the extent to which the taxonomy of failures at this level resolves the failures of the 2nd mechanism should be better than at the 2nd level as the number of classes of failures the cascading faults need to resolve to should be decreasing (with ultimate convergence at two general classes of failures modes – i.e. loss of the function and malfunction of the function).

With three (3) layers of protections, uncertainty may exist, but it is likely to be manageably reduced and bounded depending on the extent to which the cascading faults resolve to the taxonomy at the second and third layers. Therefore a system with three layers of protection may be suitable for any system, even those with severe failure modes, provided suitable mechanisms are employed at each layer. This can be expressed as shown in Figure 21.

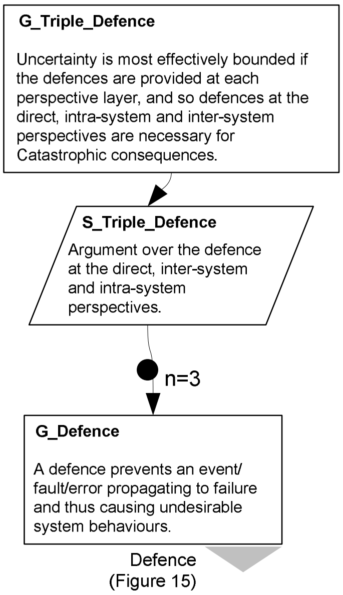

G_Triple_Defence
Uncertainty is most effectively bounded if the defences are provided at each perspective layer, and so defences at the direct, intra-system and inter-system perspectives are necessary for Catastrophic consequences.
S_Triple_Defence
Argument over the defence at the direct, inter-system and intra-system perspectives.
n=3
G_Defence
A defence prevents an event/ fault/error propagating to failure and thus causing undesirable system behaviours.
Defence
(Figure 15)

Figure 21: Triple Defence

Diversity of Layers

In essence, what the 2nd and 3rd layers of defence at differing perspective are providing is a way for defences to combine to reduce uncertainty regarding propagation of the fault/error condition to the system boundary. This can be expressed as meta-claims as per Figure 22.

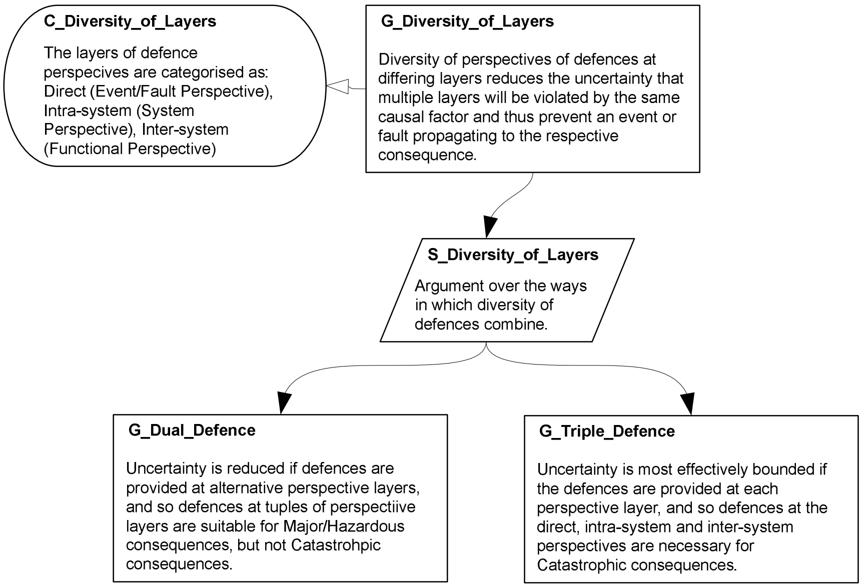

C_Diversity_of_Layers
The layers of defence perspecives are categorised as: Direct (Event/Fault Perspective), Intra-system (System Perspective), Inter-system (Functional Perspective)
G_Diversity_of_Layers
Diversity of perspectives of defences at differing layers reduces the uncertainty that multiple layers will be violated by the same causal factor and thus prevent an event or fault propagating to the respective consequence.
S_Diversity_of_Layers
Argument over the ways in which diversity of defences combine.
G_Dual_Defence
Uncertainty is reduced if defences are provided at alternative perspective layers, and so defences at tuples of perspectiive layers are suitable for Major/Hazardous consequences, but not Catastrohpic consequences.
G_Triple_Defence
Uncertainty is most effectively bounded if the defences are provided at each perspective layer, and so defences at the direct, intra-system and inter-system perspectives are necessary for Catastrophic consequences.

Figure 22: Diversity of Layers

There is still the need to address the suitability of each individual defence, which is further addressed in Chapter 5. In addition to examining each individual defence, the adequacy of the totality of defences is also important for informing risk assessments. While the above paragraphs have addressed the defence in depth aspect, coverage of the propagation paths is also relevant.

Coverage of Propagation Paths

Figure 17 illustrates that a network of potential propagation paths may exist within any system architecture. Therefore, to establish the adequacy of the knowledge of propagations paths, analysis is required on the extent to which there is:

- knowledge of possible propagation paths,
- the suitability of each individual defence based on its location in the propagation path, and
- the defence in depth for each propagation path.

The rationale for these concepts is presented in Figure 23.

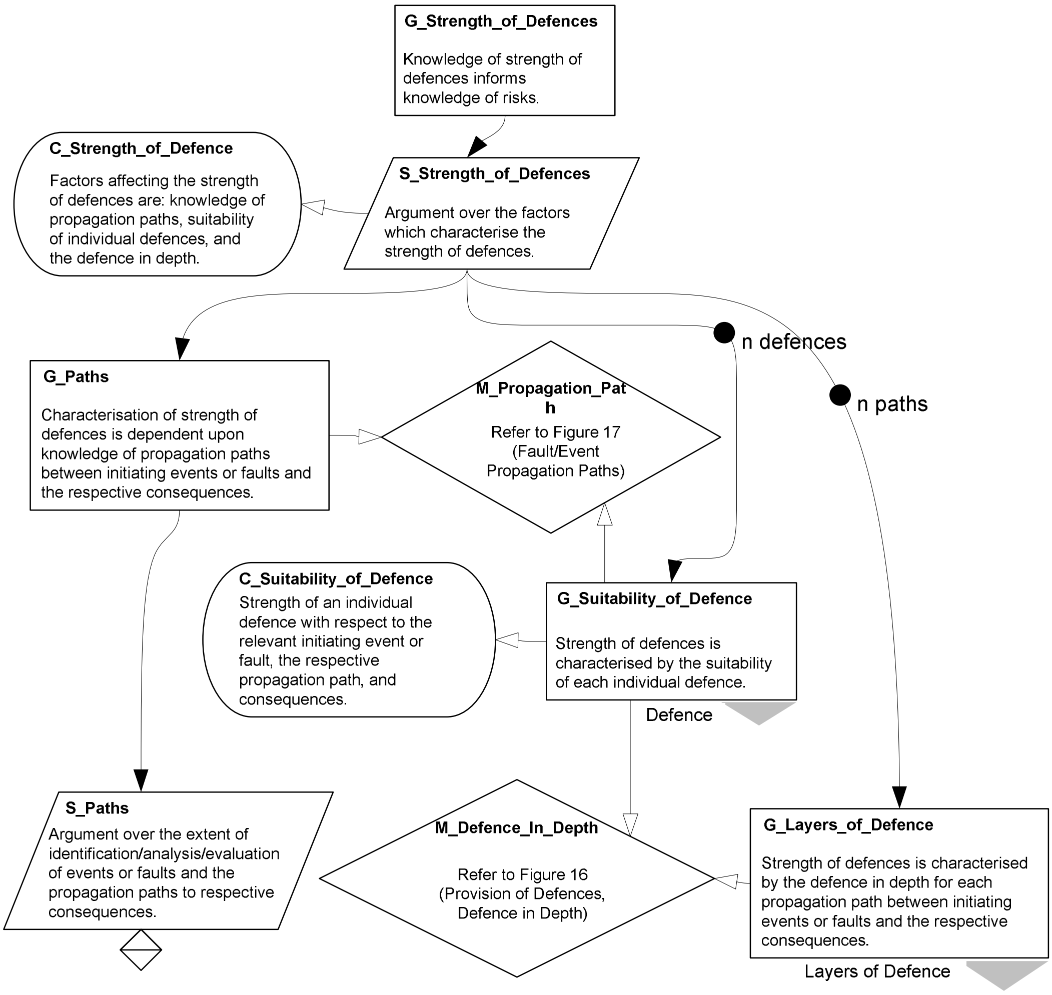

G_Strength_of_Defences
Knowledge of strength of defences informs knowledge of risks.
C_Strength_of_Defence
Factors affecting the strength of defences are: knowledge of propagation paths, suitability of individual defences, and the defence in depth.
S_Strength_of_Defences
Argument over the factors which characterise the strength of defences.
n defences
n paths
G_Paths
Characterisation of strength of defences is dependent upon knowledge of propagation paths between initiating events or faults and the respective consequences.
M_Propagation_Path
Refer to Figure 17 (Fault/Event Propagation Paths)
C_Suitability_of_Defence
Strength of an individual defence with respect to the relevant initiating event or fault, the respective propagation path, and consequences.
G_Suitability_of_Defence
Strength of defences is characterised by the suitability of each individual defence.
Defence
S_Paths
Argument over the extent of identification/analysis/evaluation of events or faults and the propagation paths to respective consequences.
M_Defence_In_Depth
Refer to Figure 16 (Provision of Defences, Defence in Depth)
G_Layers_of_Defence
Strength of defences is characterised by the defence in depth for each propagation path between initiating events or faults and the respective consequences.
Layers of Defence

Figure 23: Strength of Defences and Coverage of Propagation Paths

The ways knowledge of defence in depth and coverage of propagation paths are used in risk assessment is discussed further in Chapter 8.

Inferences about Knowledge and Uncertainty

From this rationale it is possible to see that ultimately each additional fault tolerance (detection and handling) mechanism layer bounds the uncertainty of the extent to which the cascading faults from the lower level resolve to the taxonomy of faults handled at the current layer.

Relating the layers of defences, through the characteristic of bounding uncertainty, provides a means by which to measure the overall strength of defences. This relationship is expressed in Figure 24.

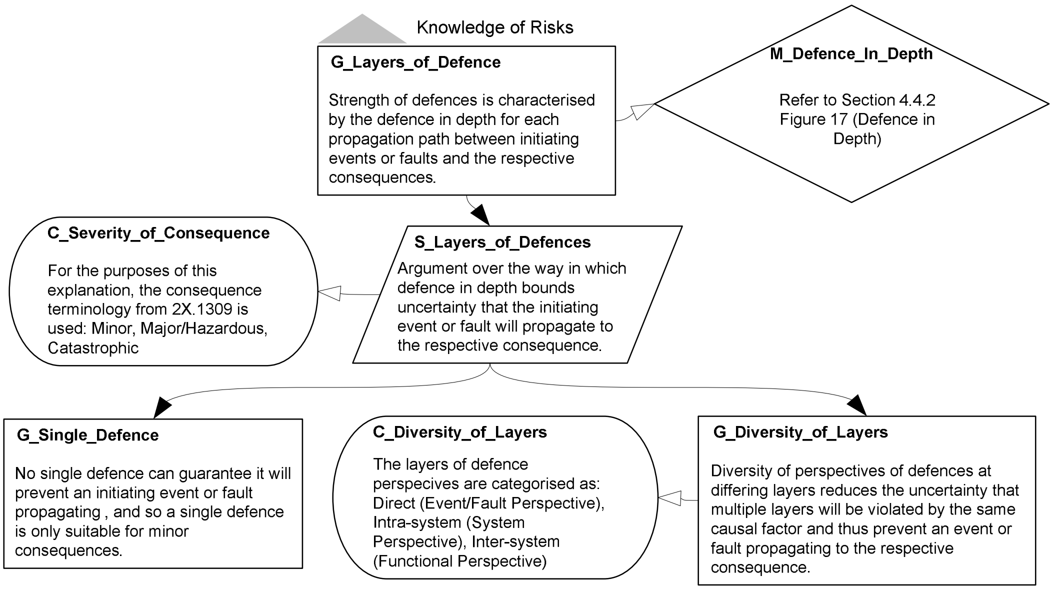

Knowledge of Risks
G_Layers_of_Defence
Strength of defences is characterised by the defence in depth for each propagation path between initiating events or faults and the respective consequences.
M_Defence_In_Depth
Refer to Section 4.4.2 Figure 17 (Defence in Depth)
C_Severity_of_Consequence
For the purposes of this explanation, the consequence terminology from 2X.1309 is used: Minor, Major/Hazardous, Catastrophic
S_Layers_of_Defences
Argument over the way in which defence in depth bounds uncertainty that the initiating event or fault will propagate to the respective consequence.
G_Single_Defence
No single defence can guarantee it will prevent an initiating event or fault propagating , and so a single defence is only suitable for minor consequences.
C_Diversity_of_Layers
The layers of defence perspecives are categorised as: Direct (Event/Fault Perspective), Intra-system (System Perspective), Inter-system (Functional Perspective)
G_Diversity_of_Layers
Diversity of perspectives of defences at differing layers reduces the uncertainty that multiple layers will be violated by the same causal factor and thus prevent an event or fault propagating to the respective consequence.

Figure 24: Layers of Defence to Bound Uncertainty

Summarising the effects of bounding uncertainty, from what was observed with actual aviation systems (Sections 4.3 and 4.4) and the analytic perspective provided in Section 4.5, results in the following:

- With no fault prevention (absence) assertion or fault tolerance (detection and handling) mechanisms, uncertainty is difficult to bound. This type of architecture should only be employed when there is no safety effect.
- With one (1) fault prevention (absence) assertion or fault tolerance (detection and handling) mechanism, uncertainty may still be large depending on the extent of the fault coverage. Therefore, a system with only one layer of protection must not have severe failure modes.
- With two (2) layers of protection, uncertainty may exist, but it is likely to be reduced and bounded depending on the extent to which the classes of cascading faults resolve to the taxonomy at the second layer. Therefore a system with two

layers of protection is suitable for any system except for those with the most severe failure modes, provided suitable protections are employed at each layer.

- With three (3) layers of protections, uncertainty may exist, but it is likely to be manageably reduced and bounded depending on the extent to which the cascading faults resolve to the taxonomy at the second and third layers. Therefore a system with three layers of protection may be suitable for any system, even those with severe failure modes, provided suitable mechanisms are employed at each layer.
- Additional layers of protection may bound the uncertainty further, provided they continue to enforce the resolution of fault classes to those analysed and treatable at the subsequent layers of protection. However, observations from the review of actual aviation systems suggest that they don't occur in practice due to cost/benefit.

Therefore, the bounding of uncertainty provides conceptually a compelling case for structuring specific layers of fault prevention (absence) and fault tolerance (detection and handling) for treating systematic faults. Combining these concepts with the observations from aviation system has permitted architectural assurance requirements to be developed and expressed as GSN meta-claims. Section 4.7 examines one possible approach to implementing these meta-claims, and also uses the usability criteria from Chapter 3.

4.7 Assurance of Architecture

This section proposes a framework for assurance of architecture of aviation systems. The framework provides a set of architectural assurance requirements based on a specific instantiation of the meta-claims presented in the previous sections of this chapter. The framework is also intended to address the principles of safety assurance pertaining to knowledge and uncertainty of product behaviours. Practicality is also emphasised through addressing usability criteria.

4.7.1 An Assurance Framework Based on Assurance Levels

Section 2.6.2 established the benefits of using assurance levels, and particularly how assurance levels assist with managing variability, subjectivity and compliance assessment. Assurance levels therefore provide a means for adhering to the usability guidelines specified in Figure 10. However, Section 2.6.1 also discussed numerous limitations with the assurance level approach which should be heeded in proposing an assurance framework based on an assurance level concept. Specifically Section 2.6

identified that the primary limitations with existing assurance standards is with the direct provision of evidence that the behaviours of the system are acceptable with respect to safety. This limitation exists because:

- the assurance levels used in existing standards don't have explicit product meaning; and
- the objectives (where used) are all expressed as outcomes of the development process, rather than in terms of their contribution to assuring behaviours of the product with respect to safety.

Therefore it is important that any safety assurance framework used ensures that the relevance of the rationale and evidence to the assurance of behaviours of the software with respect to safety remains explicit. The assurance framework should also be explicit in how much (and what strength) of evidence is necessary to make the rationale compelling and bound uncertainty. Hence it is important to establish guidelines for assurance level definitions for the assurance framework defined in this thesis to ensure that the limitations don't undermine the framework.

4.7.2 Guidelines for Safety Assurance Level Definitions

Using the benefits and limitations of assurance levels from Section 2.6, it is possible to propose some guidelines for the use of assurance levels for addressing usability criteria. This thesis proposes the following guidelines for assurance level definitions:

Guideline A – Safety assurance levels should have or be relatable to a product meaning.

Rationale: Safety assurance levels should either directly specify some physical property of the product and its behaviours, or be relatable to something that does. Non-satisfaction of the assurance level should be inferable to uncertainty of specific product behaviours or a product behavioural difference.

Guideline B – Safety assurance levels should focus on outcomes rather than activities.

Rationale: Safety assurance levels should not concern themselves with prescribing specific techniques or methods as this limits flexibility and novelty for supplier solutions. They should instead set objective benchmarks for properties of the product, rationale and its evidence that should be established.

Guideline C – Safety assurance levels should be explicit in their rationale relating evidence to product behaviours.

The relevance of the claims underpinning the assurance level definition should be made explicit i.e. the generic argument pattern to which the assurance framework conforms should be made explicit so that it is obvious when the pattern is not relevant to a specific problem or solution.

Guideline D – Safety assurance levels should provide suppliers a means to establish the suitability of supplier proposed methods and techniques.

Rationale: The framework incorporating the assurance levels should include a mechanism for inferring the relationship between any supplier proposed technique and method, and the outcomes or objectives they satisfy by ensuring that the factors/properties underpinning each objective are explicit. From this it should make transparent any limitations in supplier proposed methods and techniques, and any related evidence shortfalls.

Guideline E – Safety assurance levels should balance prescription and goal setting based on the principles and usability criteria.

Rationale: The assurance framework should be goal setting in terms of outcomes and objectives of the framework, and only as prescriptive as necessary to ensure explicit benchmarking for compliance with respect to product related behaviours. To achieve this, the framework should balance the implementation of principles based on theoretical aspiration with the usability criteria to ensure effectiveness in practice.

The framework proposed in this thesis is intended to satisfy the identified principles and usability guidelines.

4.7.3 Defining the Architectural Safety Assurance Level (ASAL) Concept

This thesis proposes an Architectural Safety Assurance Level (ASAL) concept. The ASAL provides an outcome based benchmark or measurement of the extent to which the system's architecture is tolerant to sources of systematic faults based on the concept of fail-safe design and defence in depth through layers of defences. The degree of fault tolerance can be directly associated with the aircraft failure condition severities categories defined by standards such as (SAE Aerospace, 2010) or (SAE International, 1996). Note that throughout this thesis, the failure conditions severities have been used based on the civil aviation paradigm, although alternative categories could be used from either the UK or US military paradigms. Four ASAL levels are proposed in Table 15.

Failure Condition Severity[1]	**Architectural Safety Assurance Level (ASAL)**	**Required Systematic Fault Tolerance in Fault/Event Propagation Paths**
Catastrophic	ASAL 3	At least three (3) diverse[2] systematic faults are necessary for the aircraft failure condition to be realised
Hazardous / Major	ASAL 2	At least two (2) diverse[2] systematic faults are necessary for the aircraft failure condition to be realised
Minor	ASAL 1	At least one (1) systematic fault is necessary for the aircraft failure condition to be realised
No Safety Effect	ASAL 0	Systematic fault tolerance is not required from a safety perspective, however the designer may choose to incorporate fault tolerance to provide assurance of system availability and reliability

1. The worst credible failure condition severity of loss of and malfunction of the aircraft function with which the system and its software is associated.

2. For a systematic fault to be diverse of another systematic fault, it must be shown that the activation of one fault does not automatically lead to the activation of another systematic fault. In practice this is achieved by ensuring that the faults must occur in independent components and/or at differing layers of abstraction (e.g. direct, intra-system, extra-system) where the correct functioning of the subsequent detection and handling mechanisms can be shown to be independent of the initiating fault condition or the detectable class of fault at the next layer is distinct of the initiating class of fault.

Table 15: Architectural Safety Assurance Level (ASAL) Definitions

4.7.4 ASAL Assignment Methodology

Section 2.4 identified that there are two factors that must be considered in the assignment of an assurance level, regardless of how it is defined. These are:

- what the software level is being assigned to, and
- how the assignment is performed.

Section 2.4 also identified that there are established instances (e.g. standards) where assurance levels are assigned to safety functions, configuration items, safety requirements or safety objectives.

The ASAL is intended to be assigned to the overall system. However, unlike the practices of many other assurance levels, an overall system may have multiple ASALs assigned to it. This is because ASALs are assigned based on the tuple of the:

- system (aircraft) failure mode (failure condition) (e.g. loss of roll control, malfunction of pitch control, erroneous display of altitude data);
- the established severity of this failure mode (failure condition) (i.e. Catastrophic, Hazardous, Major, Minor, or No Safety Effect); and
- the overall system as a configuration item.

In essence the ASAL relates to the severity of the consequences at the end of each identified propagation path from fault/event through the hazard. ASALs are not intended to be assigned to subordinate configuration items, because they relate to properties of the overall system. Discussion on how sub-ordinate configuration items are addressed is provided in Chapters 5 and 6.

ASAL assignment should be performed by doing the following:

1. Identify the aircraft system using one or more existing system description methodologies, such as architectural notations, descriptive languages, etc.
2. Determine potential aircraft system failure modes systematically using one or more existing methods such as Functional Hazard Assessment (FHA), System Hazard Analysis (SHA), Hazard and Operability Studies (HAZOP), etc.
3. Determine the severity of the aircraft system failure modes using established severity assignment methodologies, such as those in (SAE Aerospace, 2010) or (US DoD, 2000).
4. Perform ASAL assignment using Table 15, as follows:
    a. For each identified aircraft system failure mode (failure condition) and associated severity, identify the relevant row from column A of Table 15.
    b. Assign an ASAL per column B of Table 15 based on the relevant row.
    c. Establish a safety objective for the aircraft system based on column C of Table 15 and the relevant row for the respective aircraft system failure.

One of the advantages of assigning ASALs in this way is that the ASAL and respective safety objective are contextualised by the relevant failure mode and its severity. As most systems typically perform multiple functions, and the failure modes associated with these functions will often differ in severity, then the ASAL approach ensures that the safety objectives are commensurate with the seriousness of each specific failure mode. This is advantageous as it ensures the protection measures employed by a system are applied where they are needed most. It also overcomes some of the criticism of existing assurance level approaches that assign a level to an entire configuration irrespective of the differing severities of the failure modes of that configuration item.

While there are similarities to FDAL/IDAL assignment of ARP4754A, there are important differences. ASALs are assigned based on the top level aircraft system failure mode, and are not reducible. They set a benchmark for the defence in depth of architectural defences, and the knowledge needed of each defence and the collective defences. FDALs (and IDALs where functions are assigned to systems) are assigned at the top level, but then can be reduced based on the architectural mitigations of functional and item independence. While FDAL/IDAL reductions have to be justified, including those made to architectural elements, architecture levels typically can't be reduced under ARP5754A. However, every time a DAL is reduced on an element that provides a defence in the failure propagation path, the body of evidence is reduced on the behaviours of that element providing the defence. While there are limits to how many times an FDAL/IDAL may be reduced, a reduction in DAL reduces the evidence of the architectural mitigation with respect to the severity of the aircraft system failure mode. This differs from using architecture to bound uncertainty from Section 4.6.

4.7.5 Specifying Requirements for Fault Tolerance Mechanisms

A key factor in providing for diverse systematic faults identified in Table 15 is providing fault tolerance (detection and handling) mechanisms at differing layers of abstraction of a system. The differing layers of abstraction provide an opportunity for independence in implementation of these mechanisms. They also provide an opportunity for differing fault detection perspective and to bound the uncertainty of fault coverage of subordinate fault tolerance mechanisms.

Using the taxonomy of layers of detection and handling mechanisms identified in Section 4.4.2, the proposed ASAL framework uses the following identified layers of fault tolerance (detection and handling) mechanisms[23]:

- *Direct* – at the typical software or hardware item level, and includes fault prevention features, fault tolerance features and software/hardware implemented fault tolerance features.
- *Intra-System* – at the typical avionics equipment level (e.g. Line Replaceable Unit (LRU)) within an aircraft, and includes fault tolerant features such as:
    - command/monitors[24],
    - voting planes,
    - output wraparounds[25],
    - hardware BIT, etc.
- *Extra-System* – at the typical system architecture level within an aircraft and may include redundancy[26], analogue backup, diverse system components, etc.

[23] Layers were refined from software, LRU, and system level as a result of evaluation feedback.

[24] Note that additional software/hardware in the monitor is considered at the intra-system (LRU) level, although the safety argument for that monitor would also consider its effects at the component level.

[25] Although the feedback is usually hardware implemented, the comparison is usually in software.

Relating the layers of fault prevention (absence) assertions and fault tolerance (detection and handling) mechanisms to the ASAL concept results in Table 16.

ASAL	**1st Layer of Defence** **Fault Prevention (Absence) or Fault Tolerance (Detection and Handling)**	**2nd Layer of Defence** **Fault Tolerance (Detection and Handling)**	**3rd Layer of Defence** **Fault Tolerance (Detection and Handling)**
ASAL3	Direct	Partitioned Direct# or Intra-System*	Intra-System* or Extra-System
ASAL2	Direct	Partitioned Direct# or Intra-System or Extra-System	Not Required
ASAL1	Direct OR Intra-System OR Extra-System	Not Required	Not Required
# must be independent of the initiating failure and the 1st Absence / Detection and Handling mechanism (i.e. through a partitioning mechanism) * Must be independent of the proceeding detection/handling mechanism			

Table 16: ASAL Architecturally Layered Fault Tolerance

4.7.6 Using Architecturally Layered Fault Tolerance

While Table 15 sets outcomes for the safety performance of the system, Table 16 is intended to be a source for design requirements for the system. Table 16 was established by analysing how the fail-safe design criteria applies to systematic faults, and examining actual aviation systems to validate how this is achieved in practice. Table 16 does not prescribe specific fault prevention or fault tolerance measures, but is intended to set benchmarks for where the system architecture should exhibit these for aviation systems. On this basis, Table 16 is intended to be the default requirement for layers of defence (protection) against faults, but deviation is permitted provided it is justified and approved via consultation with the relevant certifying authority.

Table 16 should be used by:

1. For each tuple of (system, system failure mode, failure mode severity), and the allocated ASAL, establish potential initiating faults and fault propagation paths within the system using existing methods such as SHARD (Pumfrey, 1999), Fault Propagation and Transformation Notation (Fenelon & McDermid, 1994), etc.

[26] Note that redundant components running the same software configuration only provides protection against hardware related failures, or failures of independent input sensors. It provides no protection against systematic failures of the software. The emphasis here is on system level architectural features that provide protections against systematic failures by detection and handling of faults.

2. Propose, via design analysis, fault prevention and fault tolerance strategies that:
    a. conform to the layers of defences specified by Table 16, and
    b. achieve the safety objectives for the assigned ASAL per Table 15.
3. Revise the design with specific implementations of the proposed fault prevention and fault tolerance strategies.
4. Reanalyse and iterate these steps as necessary until the safety objectives of Table 15 are achieved for each tuple of (system, system failure, failure mode severity).

4.7.7 Potential Benefits of the ASAL Concept

The ASAL concept potentially provides the following perceived benefits to assurance frameworks:

- The ASAL concept explicitly integrates requirements for fault prevention and fault tolerance to systematic faults, through architectural treatments, into the traditional assurance approach, and it doesn't conflict with the existing safety analysis of civil and military standards.
- The ASAL concept provides a multi-dimensional (better than binary) perspective on the fault prevention (absence) assertions and fault tolerance (detection and handling) of systematic faults commensurate with the worst credible failure condition. This is an improvement over existing software safety assurance paradigms as it more accurately reflects the achievement of fail-safe design.
- The ASAL concept quantifies (in the product context) the degree of fault tolerance for each system's contribution to aircraft level failure conditions. Therefore, the ASAL as a level inherently has a product meaning.
- The ASAL concept is simple, and therefore doesn't burden assurance frameworks with complex, non-objective prescriptions. The rationale has been encoded within the meta-claims presented in earlier sections of this chapter.
- The ASAL concept doesn't prescribe specific architectures, although it does imply an aviation system context. It is therefore inherently flexible and doesn't constrain designer's choices on the specific fault prevention or fault tolerance mechanisms they believe are best. It focuses on the treatment of systematic faults by the architecture.
- The ASAL concept encourages fault tolerant architectures for the systems whose functions most need fault tolerance (i.e. those with severe hazards or failure conditions).

- The ASAL concept is compatible with observations of systematic fault tolerance management in actual aviation systems from Section 4.3, and those observations made by (Edwards, et al., 2010).

The validity of these benefits is evaluated in Chapter 10 of this thesis.

4.7.8 Potential Limitations of the ASAL Concept

The ASAL concept introduces or highlights the following potential limitations:

- The ASAL concept sets no benchmarks for the level of evidence required to demonstrate that numbers of diverse systematic faults do not contribute to identified failure modes. The ASAL concept does not address 'how much is enough?' for design assurance evidence. Chapters 5 and 6 will examine this in further detail.
- The ASAL concept relies on bounding uncertainty, of which a fundamental factor is the extent to which faults at one layer of abstraction resolve to a detectable set in the perspective of the next layer of abstraction. However, the ASAL concept doesn't provide an explicit measure of the specific contextual claims about fault prevention (absence) and fault tolerance (detection and handling) of systematic faults as they propagate to high levels of system abstraction. Thus it doesn't support inferences about the suitability of the specific proposed detection and handling capabilities of the system architecture. Chapters 5 and 8 examine this aspect further.

The impact of these limitations is evaluated in Chapters 9 and 10.

4.7.9 Additional Factors Effecting Assurance Level Assignment

Section 2.4 also identified factors that are used to reduce or refine the assigned assurance levels in existing frameworks. The following paragraphs provide discussion of the relevance of these concepts to the ASAL approach.

Conceptual and Mechanistic Independence

Conceptual and mechanistic independence have been suggested by (Weaver, 2003) as playing an important factor in assurance of arguments constructed around (Pumfrey, 1999)'s software failure taxonomy. How does conceptual and mechanistic independence relate to the ASAL concept defined?

The definitions within the ASAL concept specify several diverse faults. This implies that there is conceptual independence between the initiating direct, the intra-system level and extra-system level detection and handling mechanisms (where relevant). Systems sharing common software and/or hardware may be prone to common mode failure conditions and are not considered to be diverse. Unless mechanistic independence delivers conceptually different architectures during the design process, it does not play a role in the ASAL concept directly. The role of mechanistic independence will be considered in Chapter 6.

On-demand versus Continuous-demand Systems

The concept of on-demand versus continuous-demand systems has most prominence within the IEC61508 standard (IEC, 2010). However, some conceptual consideration of the potential impact on the ASAL concept is beneficial because it reveals how the approach deals with established concepts.

The ASAL concept was largely derived in the context of actual aviation systems that are inherently continuous demand systems, although specific functions provided by individual safety functions maybe also classified as on-demand. But, how does the ASAL concept apply for on-demand systems versus continuous demand systems?

On-demand systems (usually used for protection systems) are usually associated with an availability requirement (which is continuous demand) on a related aviation system associated with the protection mechanism. Therefore in most cases there is little practical difference between an on-demand system and continuous demand system with respect to the ASAL concept. Actually, this insight is useful, because it also provides evidence that on-demand versus continuous demand is not a useful distinguisher when taking a perspective of systematic faults and fault tolerance.

4.8 Defining a Process for Applying the ASAL Concept

Having defined architectural assurance requirements in both Table 15 and Table 16, it is necessary to define an overall lifecycle process for using these concepts. Figure 25 provides an overview of the process, which incorporates those sub-processes defined in Sections 4.7.4 and 4.7.6. Figure 25 has been developed to be consistent with the typical systems engineering ‘V’ model, such as described by (Leveson, 1995), (Storey, 1996) and (Kossiakoff, et al., 2011).

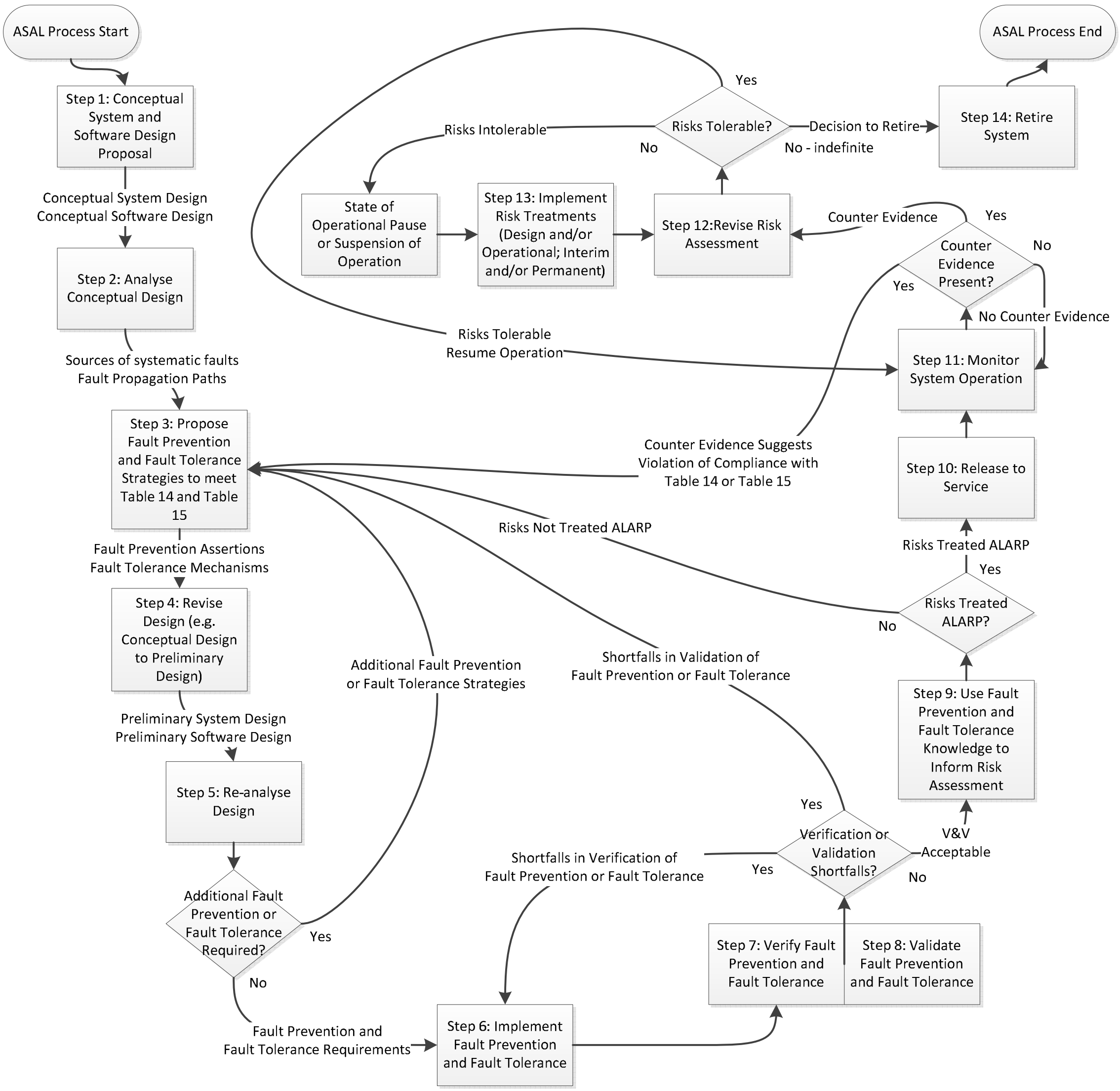

ASAL Process Start
Step 1: Conceptual System and Software Design Proposal
Conceptual System Design
Conceptual Software Design
Step 2: Analyse Conceptual Design
Sources of systematic faults
Fault Propagation Paths
Step 3: Propose Fault Prevention and Fault Tolerance Strategies to meet Table 14 and Table 15
Fault Prevention Assertions
Fault Tolerance Mechanisms
Step 4: Revise Design (e.g. Conceptual Design to Preliminary Design)
Preliminary System Design
Preliminary Software Design
Step 5: Re-analyse Design
Additional Fault Prevention or Fault Tolerance Required?
Yes
No
Fault Prevention and
Fault Tolerance Requirements
Step 6: Implement Fault Prevention and Fault Tolerance
Step 7: Verify Fault Prevention and Fault Tolerance
Step 8: Validate Fault Prevention and Fault Tolerance
Verification or Validation Shortfalls?
Yes
Yes
No
Shortfalls in Verification of
Fault Prevention or Fault Tolerance
Shortfalls in Validation of
Fault Prevention or Fault Tolerance
Additional Fault Prevention
or Fault Tolerance Strategies
V&V
Acceptable
Step 9: Use Fault Prevention and Fault Tolerance Knowledge to Inform Risk Assessment
Risks Treated ALARP?
No
Yes
Risks Not Treated ALARP
Risks Treated ALARP
Step 10: Release to Service
Step 11: Monitor System Operation
No Counter Evidence
Counter Evidence Present?
Yes
No
Yes
Counter Evidence
Counter Evidence Suggests
Violation of Compliance with
Table 14 or Table 15
Risks Tolerable
Resume Operation
Step 12:Revise Risk Assessment
Risks Tolerable?
Yes
No
Risks Intolerable
Decision to Retire
No - indefinite
State of Operational Pause or Suspension of Operation
Step 13: Implement Risk Treatments (Design and/or Operational; Interim and/or Permanent)
Step 14: Retire System
ASAL Process End

Figure 25: ASAL Process Overview

The following provides elaboration of each of the ASAL process steps. Each step is illustrated by use of the A-DHC-4 fictitious example from Section 3.8.

4.8.1 Step 1 – Conceptual Design Proposal

a. *Propose a conceptual system architectural design based on the allocated requirements and derived requirements, including safety requirements using a process such as described by* (SAE Aerospace, 2010).

Example – A-DHC-4

Figure 26 details the conceptual architecture of the flight control system based on a standard functional hazard assessment and associated processes. This example focusses on the functional failure of 'Malfunction of Roll Control' which has been assessed to be Catastrophic using Functional Hazard Analysis (FHA).

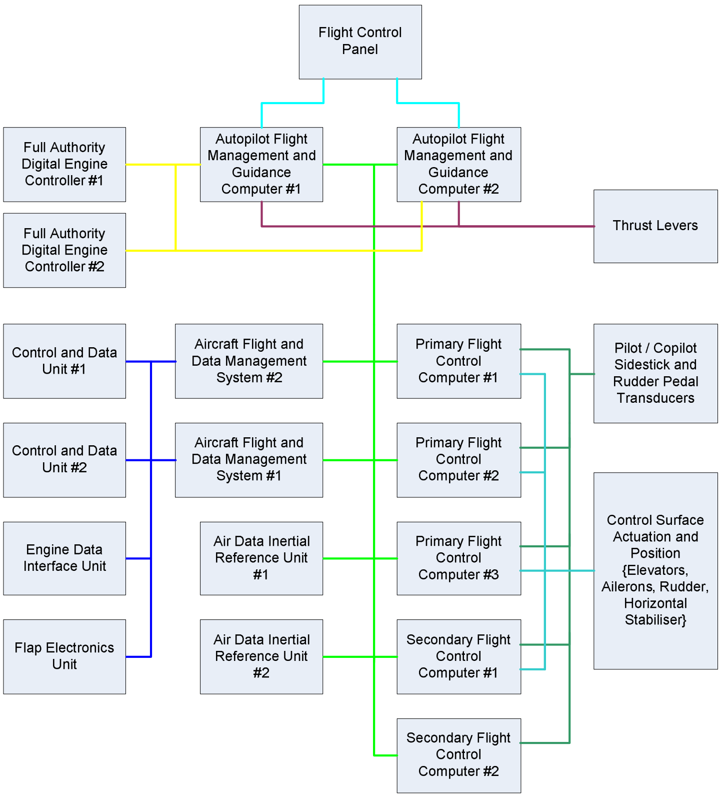
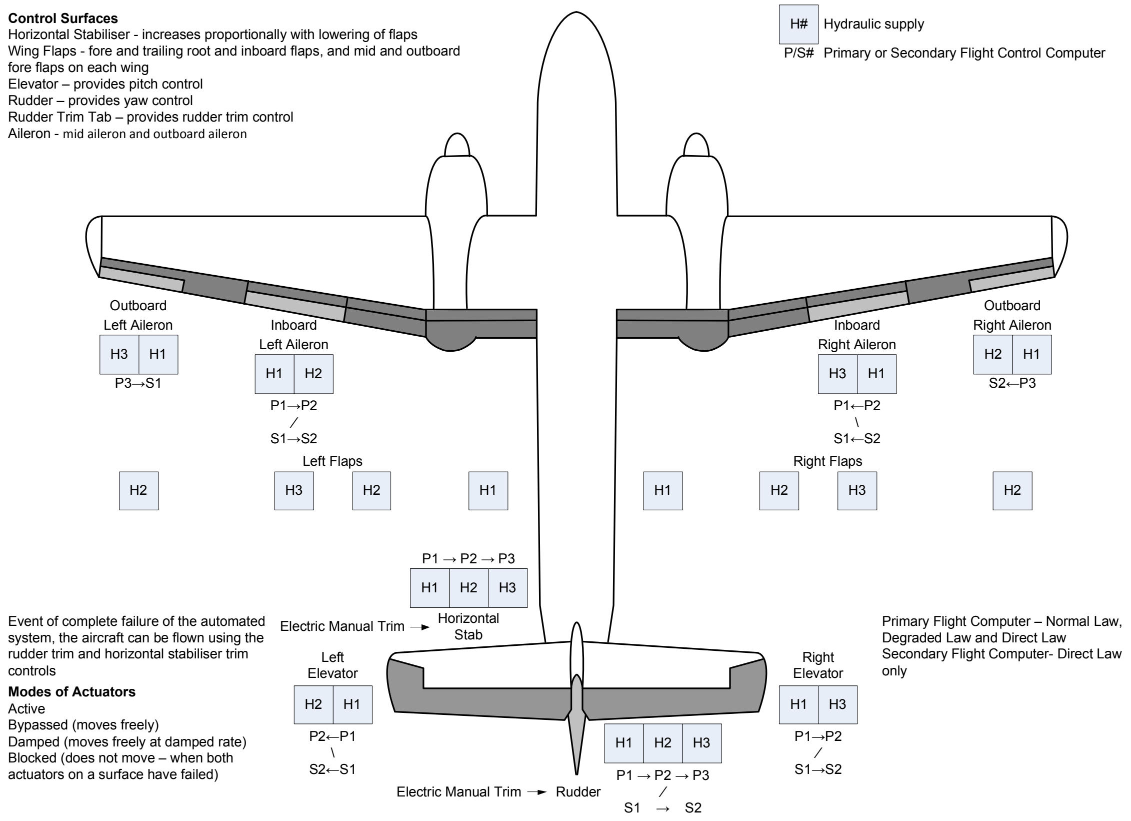

Flight Control Panel
Full Authority Digital Engine Controller #1
Autopilot Flight Management and Guidance Computer #1
Autopilot Flight Management and Guidance Computer #2
Full Authority Digital Engine Controller #2
Thrust Levers
Control and Data Unit #1
Aircraft Flight and Data Management System #2
Primary Flight Control Computer #1
Pilot / Copilot Sidestick and Rudder Pedal Transducers
Control and Data Unit #2
Aircraft Flight and Data Management System #1
Primary Flight Control Computer #2
Control Surface Actuation and Position {Elevators, Ailerons, Rudder, Horizontal Stabiliser}
Engine Data Interface Unit
Air Data Inertial Reference Unit #1
Primary Flight Control Computer #3
Flap Electronics Unit
Air Data Inertial Reference Unit #2
Secondary Flight Control Computer #1
Secondary Flight Control Computer #2

Figure 26: Conceptual Flight Control System Architecture

The A-DHC-4 incorporates a total of five Flight Control Computers (FCC) – three *Primary Flight Control Computers*, and two *Secondary Flight Control Computers*. The five flight control computers are integrated with two *Autopilot Flight Management and Guidance Computers*. The flight control system has three modes of operation:

- *Managed Guidance* from the *Flight and Data Management System – Normal* and *Degraded* modes
- *Selected Guidance* by the pilot (input speed, heading, altitude, vertical speed/flight path angle) entered via the *Flight Control Panel – Normal* and *Degraded* modes
- *Direct Control – Normal*, *Degraded* and *Direct* modes

Figure 27 shows the FCCs used to control the actuators for each control surface, and the associated hydraulic supplies.



Control Surfaces
Horizontal Stabiliser - increases proportionally with lowering of flaps
Wing Flaps - fore and trailing root and inboard flaps, and mid and outboard fore flaps on each wing
Elevator – provides pitch control
Rudder – provides yaw control
Rudder Trim Tab – provides rudder trim control
Aileron - mid aileron and outboard aileron

Event of complete failure of the automated system, the aircraft can be flown using the rudder trim and horizontal stabiliser trim controls

Modes of Actuators
Active
Bypassed (moves freely)
Damped (moves freely at damped rate)
Blocked (does not move – when both actuators on a surface have failed)

Primary Flight Computer – Normal Law, Degraded Law and Direct Law
Secondary Flight Computer- Direct Law only

Figure 27: Conceptual Flight Control Architecture (Computers, Control Surfaces, Hydraulics)

The A-DHC-4 incorporates three independent hydraulic systems (labelled H1, H2 and H3) for actuation of aircraft control surfaces. There are no mechanical backups to the digital flight control system. In the event of complete failure of the automated system, the aircraft can be flown using the electric rudder trim and horizontal stabiliser trim controls. The analysis will assume that this property is established via aerodynamic analysis, simulation and prior flight testing.

b. *Propose a conceptual hardware architectural design based on the requirements allocated to hardware, and additional derived requirements, including safety requirements.*

Example – A-DHC-4

As shown in Figure 28, each primary FCC consists of a command channel and a monitor channel. The command channel processes the control laws, whereas the monitor channel is responsible for monitoring the inputs, processing and outputs of the command channel.

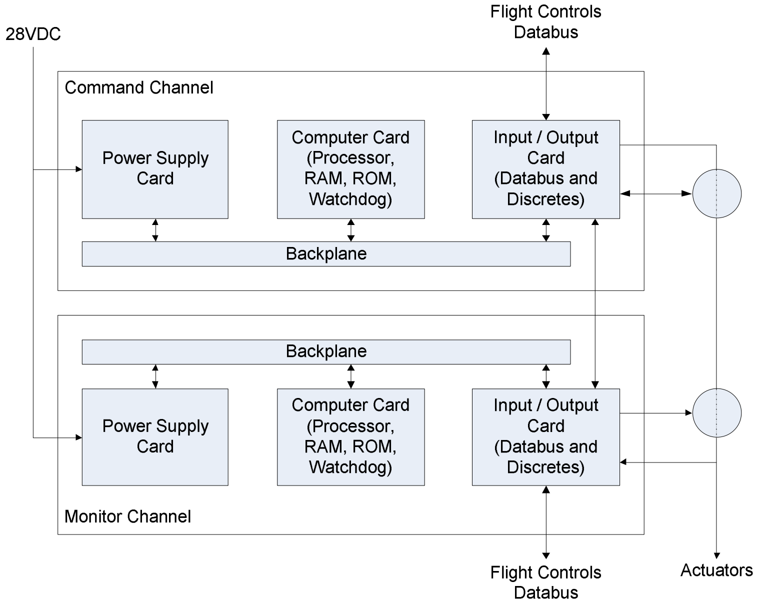

28VDC
Flight Controls Databus
Command Channel
Power Supply Card
Computer Card (Processor, RAM, ROM, Watchdog)
Input / Output Card (Databus and Discretes)
Backplane
Backplane
Power Supply Card
Computer Card (Processor, RAM, ROM, Watchdog)
Input / Output Card (Databus and Discretes)
Monitor Channel
Flight Controls Databus
Actuators

Figure 28: Conceptual Primary Flight Control Computer Hardware Architecture

Each channel is implemented using identical digital microprocessors. The primary flight control computer contains a two software packages (*Primary Command Channel*, *Primary Monitor Channel*). This example will focus specifically on the *Primary Command Channel*, although the *Monitor Channel*, *Secondary Flight Control Computers*, and other features of the architecture will be used to show the handling of failure modes not handled by the *Primary Flight Control Computer*.

c. *Propose the conceptual software architectural design based on the requirements allocated to software, and additional derived requirements, including safety requirements.*

<u>Example – A-DHC-4</u>

The conceptual software architectural of the *Primary Command Channel* design is summarised in Figure 29. The figure uses MASCOT notation, as defined by (Joint IECCA and MUF Committee on MASCOT, 1987), (Simpson, 1986), (Simpson & Jackson, 1979) and uses the Real Time Network protocols of (Simpson, 1996), and (Simpson, 1994). The reader is referred to these references for more information.

In this example we focus on the sensor_data input to the Signal Data Conditioning element. The sensor_data communication is derived from sensor data read from the flight control databus.

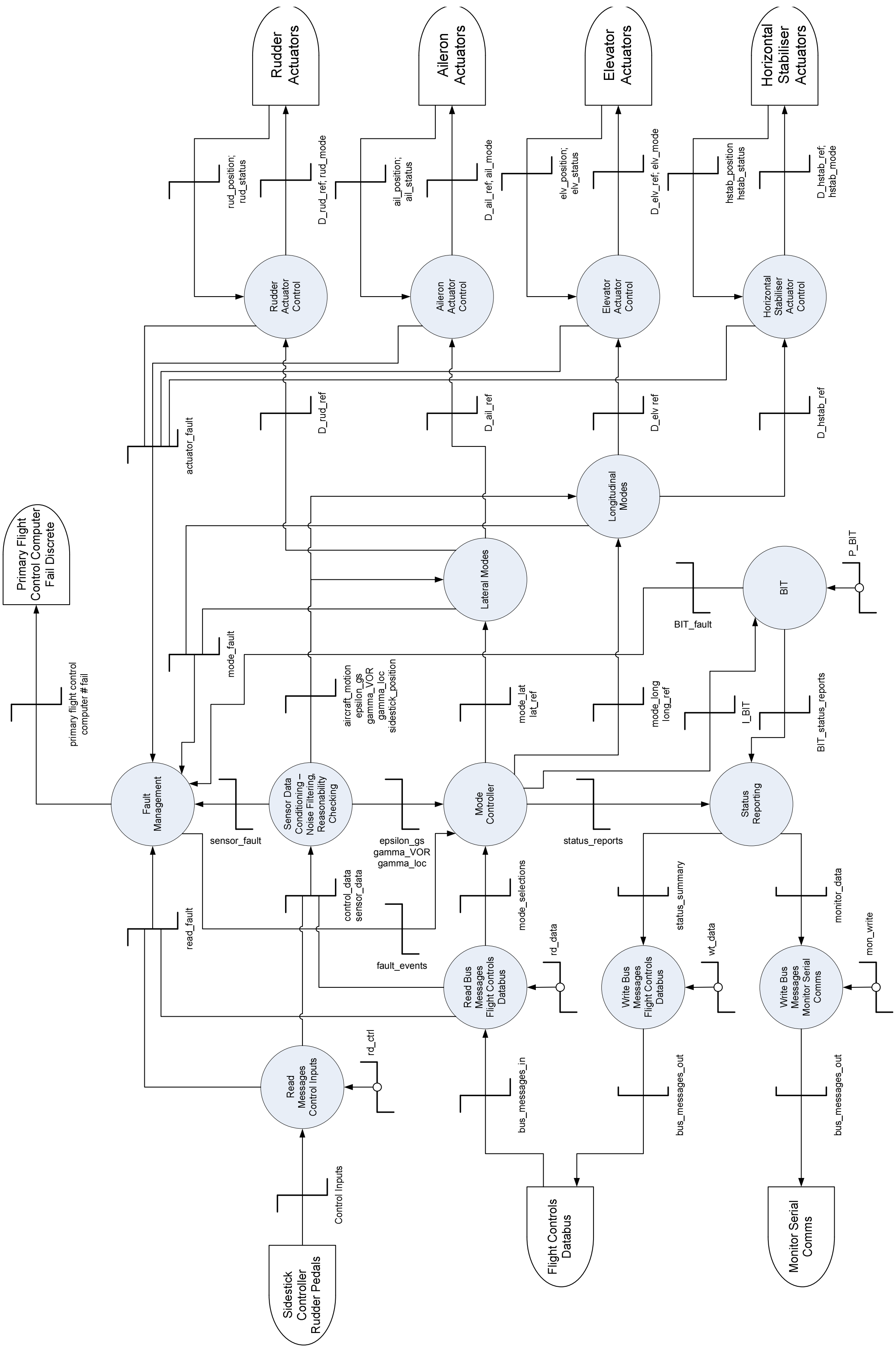


Figure 29: Software Architecture for Primary Command Channel – MASCOT representation

4.8.2 Step 2 – Analyse Conceptual Design

a. *Undertake a HAZOP, SHARD (refer to* (Pumfrey, 1999)*) or equivalent analysis of the conceptual hardware and software architectural design to identify relevant hardware and software failure conditions and to prompt design resolution consideration.*

Example – A-DHC-4

Table 17 shows an extract from a SHARD analysis (refer (Pumfrey, 1999)) for the value failure of the sensor_data attitude source #1 communication. The SHARD analysis identifies one type of value failure for the sensor_data attitude source #1 communication and examines the effects and the proposal of detection/protection mechanisms.

Guide Word	Deviation	Cause	Effect	Detection / Protection
Value	Attitude source #1 has failed fixed at +10deg roll attitude	Hardware failure of attitude sensor	Attitude reference is calculated from average of Attitude source #1 and Attitude source #2. A erroneously fixed attitude will cause the calculated attitude reference to be both erroneous and lag in dynamic response. At an aircraft level, this would mean the flight control system would exhibit incorrect roll attitude command and would lag in dynamic response.	PFCC monitor channel employs attitude source divergence monitoring to detect inappropriate input attitudes. If an inappropriate attitude is detected, then the PFCC mode is set to degraded and the monitor sets the applicable attitude to invalid. SFCC uses turn rate detection to detect inappropriate command response during turns. SFCC reverts FCS to Secondary 'direct' mode and the pilot must establish the correct attitude from the displays, including standby sources.
…	…	…	…	…

Table 17: SHARD for sensor_data Communication 'Value' Failure

b. *Undertake relevant system safety analysis of the system architectural design to identify relevant fault propagation paths for identified hardware and software failure conditions from the software/system interfaces to the system boundary (at which hazards and risks can be identified).*

Example – A-DHC-4

One way of understanding the fault propagation paths applicable to the aforementioned value failure of sensor_data is to produce a diagrammatic representation using Fault Propagation Transformation Notation (FPTN), as defined by (Fenelon & McDermid, 1994). Alternatively, the notation and methods of time triggered architecture could be used, as defined by (Kopetz & Bauer, 2003). Figure 30 presents a simplified FPTN that focusses on the propagation of the sensor_data.attitude#1 value failure. It shows the propagation of the fault through the Primary FCC command channel software, monitor and Secondary FCC. The proposed detection and handling mechanisms are shown as defences at intra-system and extra-system layers.

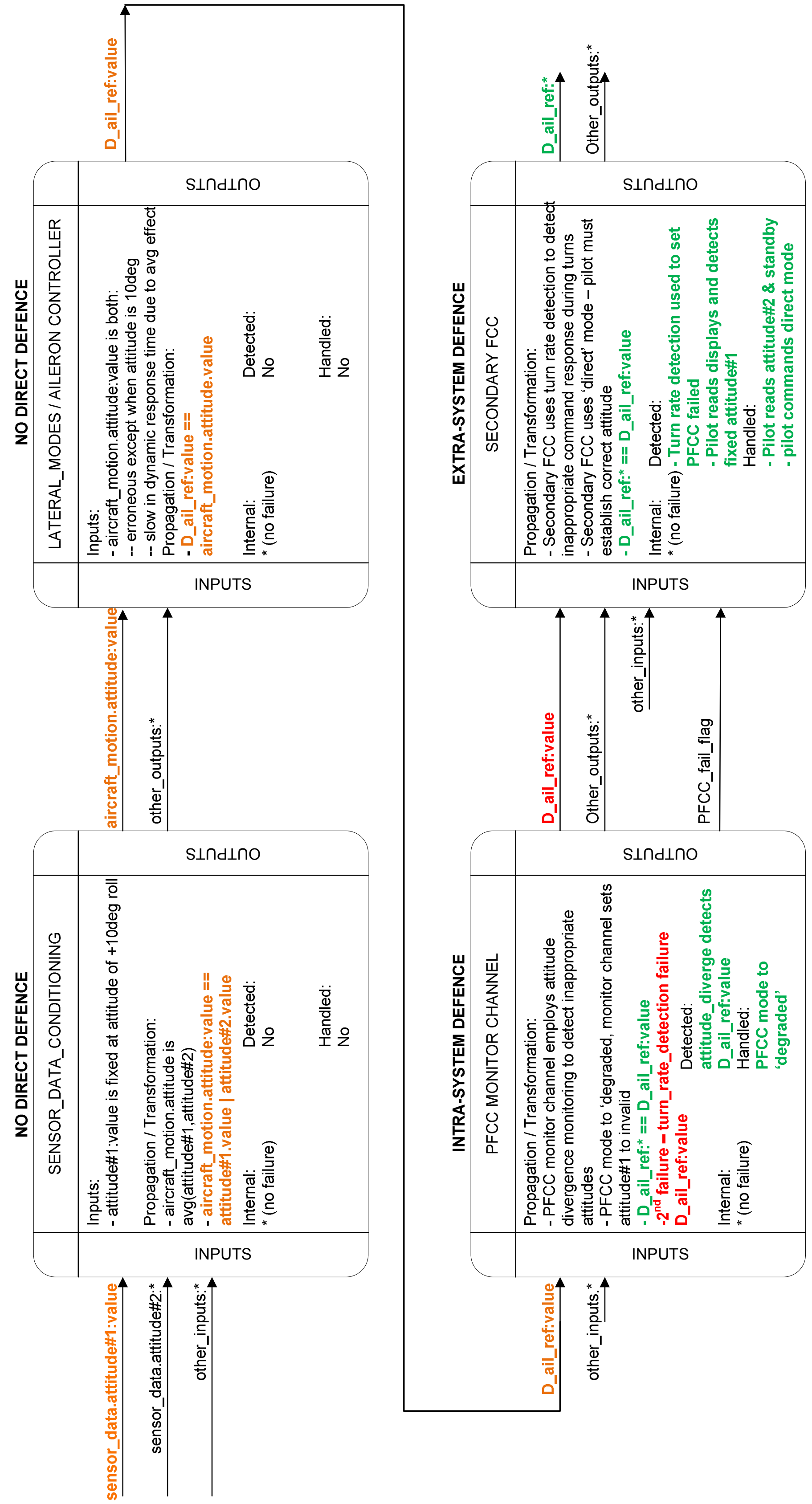

NO DIRECT DEFENCE
SENSOR_DATA_CONDITIONING
INPUTS
OUTPUTS
Inputs:
- attitude#1:value is fixed at attitude of +10deg roll
Propagation / Transformation:
- aircraft_motion.attitude is avg(attitude#1,attitude#2)
- aircraft_motion.attitude:value == attitude#1.value | attitude#2.value
Internal:
* (no failure)
Detected:
No
Handled:
No
sensor_data.attitude#1:value
sensor_data.attitude#2:*
other_inputs:*
aircraft_motion.attitude:value
other_outputs:*
NO DIRECT DEFENCE
LATERAL_MODES / AILERON CONTROLLER
INPUTS
OUTPUTS
Inputs:
- aircraft_motion.attitude:value is both:
-- erroneous except when attitude is 10deg
-- slow in dynamic response time due to avg effect
Propagation / Transformation:
- D_ail_ref:value == aircraft_motion.attitude.value
Internal:
* (no failure)
Detected:
No
Handled:
No
D_ail_ref:value
INTRA-SYSTEM DEFENCE
PFCC MONITOR CHANNEL
INPUTS
OUTPUTS
Propagation / Transformation:
- PFCC monitor channel employs attitude divergence monitoring to detect inappropriate attitudes
- PFCC mode to 'degraded, monitor channel sets attitude#1 to invalid
- D_ail_ref:* == D_ail_ref:value
-2nd failure – turn_rate_detection failure D_ail_ref:value
Detected:
attitude_diverge detects D_ail_ref:value
Handled:
PFCC mode to 'degraded'
Internal:
* (no failure)
D_ail_ref:value
other_inputs:*
D_ail_ref:value
Other_outputs:*
PFCC_fail_flag
EXTRA-SYSTEM DEFENCE
SECONDARY FCC
INPUTS
OUTPUTS
Propagation / Transformation:
- Secondary FCC uses turn rate detection to detect inappropriate command response during turns
- Secondary FCC uses 'direct' mode – pilot must establish correct attitude
- D_ail_ref:* == D_ail_ref:value
Internal:
* (no failure)
Detected:
- Turn rate detection used to set PFCC failed
- Pilot reads displays and detects fixed attitude#1
Handled:
- Pilot reads attitude#2 & standby
- pilot commands direct mode
other_inputs:*
D_ail_ref:*
Other_outputs:*

Figure 30: FPTN Representation of Fault Propagation Path and Preliminary Defences

4.8.3 Step 3 – Propose Fault Prevention and Fault Tolerance Strategies

Identify and analyse potential fault prevention and fault tolerance strategies for each software failure condition and propagation path such that they satisfy the requirements of Table 15 (protection against total number of faults) and Table 16 (architectural location of fault prevention and fault tolerance).

Example – A-DHC-4

As the 'Malfunction of Roll Control' is catastrophic, Table 15 prescribes that the architecture must be assured to ASAL 3, implying that at least three diverse systematic faults are required to leading to the catastrophic outcome. Table 16 therefore states that fault prevention and/or fault tolerance is required at all perspective layers (Direct, Intra-System, and Extra-System) layers.

Inspection of Figure 30 reveals that the conceptual design proposal only includes protection at the intra-system and extra-system layers. Therefore an additional protection is required at the direct layer. It isn't possible to achieve fault prevention for a sensor fault, and so fault tolerance is required at the direct layer.

Figure 31 proposes an additional defence at the Direct layer. The defence is detection by a reasonability check of attitude data sources, and marking of invalid of any attitude source that does not meet the reasonability criteria. For the purposes of this example, it is assumed that the reasonability check can detect a frozen attitude based on integration over time based checks and real time inputs from other sources of information.

4.8.4 Step 4 – Revise Design (e.g. Conceptual to Preliminary Design)

a. Revise system architectural design based on the results of Step 3.

b. Revise software and hardware architectural design based on the results of Step 3.

Example – A-DHC-4

As the additional defence is within the software, the signal conditioning function within Figure 29 should be updated to reference the additional reasonability check specified above. The system and hardware architectures do not require any further updating based on the analysis of this fault propagation path. It is possible, however, that analysis of other faults may require difference changes to the system, hardware and software architectures. As each fault and propagation path is analysed, these will be identified and the architecture refined.

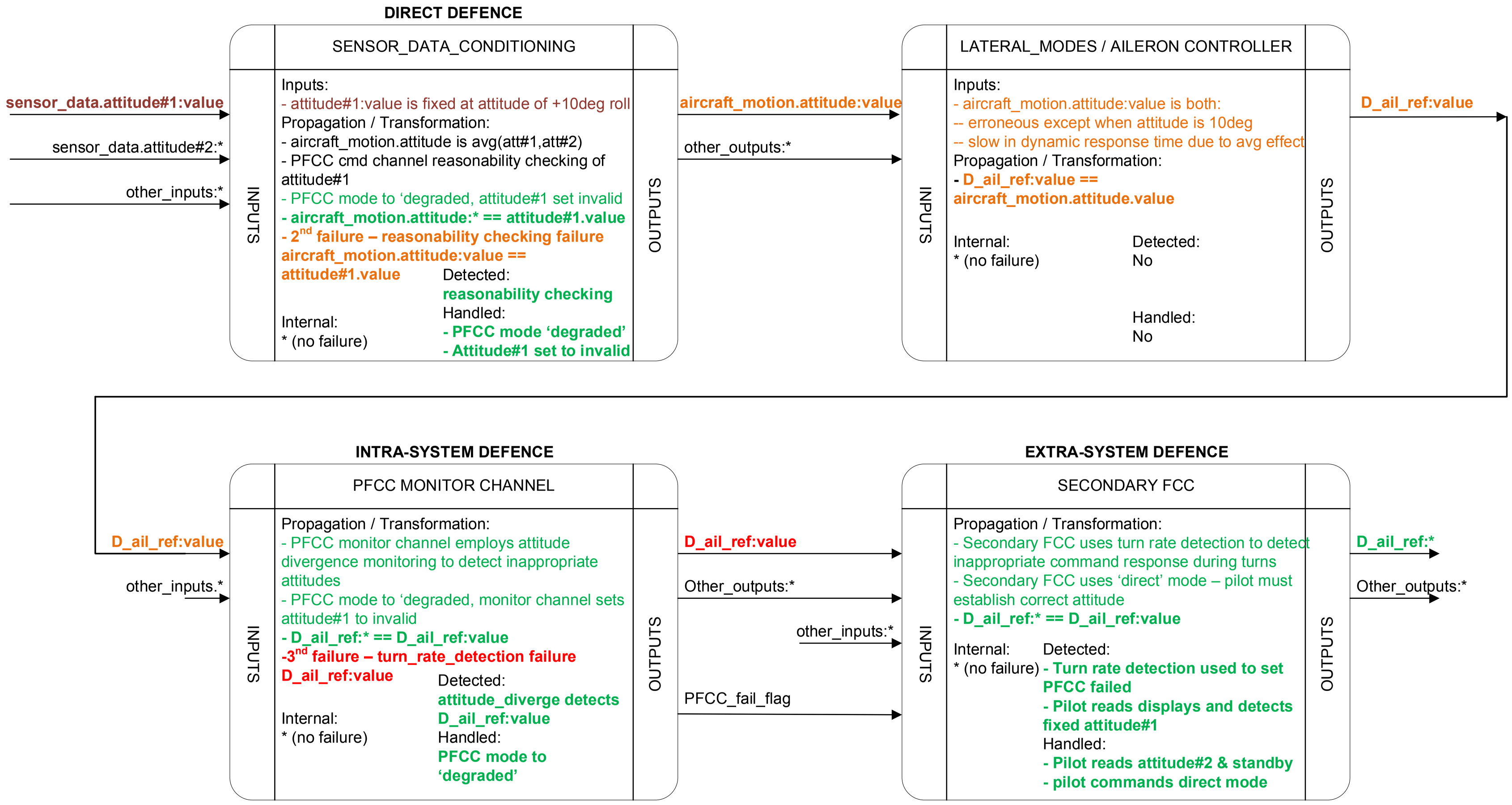

DIRECT DEFENCE
SENSOR_DATA_CONDITIONING
INPUTS
OUTPUTS
sensor_data.attitude#1:value
sensor_data.attitude#2:*
other_inputs:*
Inputs:
- attitude#1:value is fixed at attitude of +10deg roll
Propagation / Transformation:
- aircraft_motion.attitude is avg(att#1,att#2)
- PFCC cmd channel reasonability checking of attitude#1
- PFCC mode to 'degraded, attitude#1 set invalid
- aircraft_motion.attitude:* == attitude#1.value
- 2nd failure – reasonability checking failure aircraft_motion.attitude:value == attitude#1.value
Internal:
* (no failure)
Detected:
reasonability checking
Handled:
- PFCC mode 'degraded'
- Attitude#1 set to invalid
aircraft_motion.attitude:value
other_outputs:*
LATERAL_MODES / AILERON CONTROLLER
INPUTS
OUTPUTS
Inputs:
- aircraft_motion.attitude:value is both:
-- erroneous except when attitude is 10deg
-- slow in dynamic response time due to avg effect
Propagation / Transformation:
- D_ail_ref:value == aircraft_motion.attitude.value
Internal:
* (no failure)
Detected:
No
Handled:
No
D_ail_ref:value
INTRA-SYSTEM DEFENCE
PFCC MONITOR CHANNEL
INPUTS
OUTPUTS
D_ail_ref:value
other_inputs.*
Propagation / Transformation:
- PFCC monitor channel employs attitude divergence monitoring to detect inappropriate attitudes
- PFCC mode to 'degraded, monitor channel sets attitude#1 to invalid
- D_ail_ref:* == D_ail_ref:value
-3rd failure – turn_rate_detection failure D_ail_ref:value
Internal:
* (no failure)
Detected:
attitude_diverge detects D_ail_ref:value
Handled:
PFCC mode to 'degraded'
D_ail_ref:value
Other_outputs:*
other_inputs:*
PFCC_fail_flag
EXTRA-SYSTEM DEFENCE
SECONDARY FCC
INPUTS
OUTPUTS
Propagation / Transformation:
- Secondary FCC uses turn rate detection to detect inappropriate command response during turns
- Secondary FCC uses 'direct' mode – pilot must establish correct attitude
- D_ail_ref:* == D_ail_ref:value
Internal:
* (no failure)
Detected:
- Turn rate detection used to set PFCC failed
- Pilot reads displays and detects fixed attitude#1
Handled:
- Pilot reads attitude#2 & standby
- pilot commands direct mode
D_ail_ref:*
Other_outputs:*

Figure 31: FPTN including Fault Prevention / Tolerance

4.8.5 Step 5 – Re-Analyse Design

a. *Revise SHARD (or equivalent analysis) for the preliminary software/hardware architectural design to include results of Step 3 and 4. Identify any additional software/hardware failure conditions and to prompt design resolution consideration if necessary (in which case return to Step 3 as required).*

Example – A-DHC-4

The SHARD from Table 17 would be updated to include the additional detection/protection mechanism.

b. *Revise relevant system safety analysis of the system architectural design to include results of Step 3 and 4. Identify any new or revised fault propagation paths for new or revised software/hardware failure conditions from the software/hardware/system interfaces to the system boundary (at which hazards and risks can be identified), and return to Step 3 if required.*

Example – A-DHC-4

Figure 31 shows the revised FPTN showing the inclusion of the additional Defence. For each additional fault and fault propagation path, the process is iterated and the architectures refined. For the purposes of simplicity within this thesis, the analysis of additional faults is not shown.

4.8.6 Step 6 – Implement Fault Prevention and Fault Tolerance

Implement fault prevention and fault tolerance in the system, LRU and software/hardware designs.

Example – A-DHC-4

Chapters 5 and 6 provide guidance and continuation of the A-DHC-4 example on implementing fault prevention and fault tolerance defences, and how to provide assurance of these activities.

4.8.7 Step 7 – Verify Fault Prevention and Fault Tolerance

a. *Undertake verification of each fault prevention assertion or fault tolerance mechanism to establish that they satisfy the requirements for protection against the relevant classes of systematic faults, and that they satisfy Table 16. Chapters 5 and 6 provide some guidance on providing assurance of these activities.*

b. *Undertake verification of each fault propagation path to establish appropriate interactions between fault prevention assertions and fault tolerance mechanisms, and that they satisfy Table 15.*

c. *Resolve verification shortfalls by returning to Step 6.*

Example – A-DHC-4

Verification of defences will usually be through one or more of design refinement analysis, implementation analysis, low level software testing, software integration testing, software hardware integration testing, systems integration laboratory testing, rig testing, ground and flight testing. Chapters 5 and 6 provide guidance and continuation of the A-DHC-4 example on verifying fault prevention and fault tolerance defences, and how this evidence relates to provide assurance of these activities.

4.8.8 Step 8 – Validate Fault Prevention and Fault Tolerance

a. *Undertake validation of each fault prevention assertion or fault tolerance mechanism to establish that the system and software behaviour under these conditions is consistent with the achievement of safety.*

b. *Undertake validation of each fault propagation path to establish that the interactions between fault prevention assertions and fault tolerance mechanisms are consistent with the achievement of safety.*

c. *Resolve validation shortfalls by returning to Step 3.*

Example – A-DHC-4

Validation of defences will usually be through one or more of requirements analysis, systems design analysis, systems integration laboratory testing, rig testing, ground and flight testing. Chapters 5 and 6 provide guidance and continuation of the A-DHC-4 example on validating fault prevention and fault tolerance defences, and how this evidence relates to provide assurance of these activities.

4.8.9 Step 9 – Use Fault Prevention and Fault Tolerance to Inform Risk Assessment

Inform the risk assessment using the knowledge of fault prevention and fault tolerance – refer to Chapter 8. If risks are not adequately resolved (refer Chapter 8), then return to Step 3.

4.8.10 Step 10 – Release to Service and commence operating the system.

Once the risk of operating the system has been assessed as tolerable, then the system may achieve release to service, and operation commenced. Faults and failures in-service should be treated using the fail-safe design and layers of defences.

4.8.11 Step 11 – Monitor System Operation

a. *Monitor for counter evidence to verification and validation of fault prevention and fault tolerance.*

b. *If counter evidence indicates that compliance with Table 15 or Table 16 is threatened, then return to Step 3.*

If in-service monitoring detects instances where there are inadequacies in fault/event or failure identification, knowledge of propagation paths, suitability of defences or defence in depth, then there may be a change to the risk of operating the system.

Example – A-DHC-4

The monitoring reveals that the PFCC misses certain sub-classes of the value failure during sensor failure events. This failure case is re-analysed using the earlier steps in this process.

4.8.12 Step 12 – Revise Risk Assessment based on counter evidence.

a. *If risks are intolerable, then enter a state of operational pause or suspended operation until additional risk treatments can be implemented.*

b. *If further risk reduction is no longer practical or cost effective, then it may be necessary to recommend retiring the system.*

c. *If the risks are tolerable, then return to Step 11.*

Example – A-DHC-4

The investigation reveals that the checks carried out by the PFCC miss certain sub-classes of the value failure. Chapter 8 provides guidance and continuation of the A-DHC-4 example on revising risk assessments based on counter evidence.

4.8.13 Step 13 – Implement Risk Treatment

Implement design and /or operational treatments. Treatments may be interim or permanent depending on the operational imperative. Interim operational treatments may be implemented until such a stage as permanent design treatments are implemented.

Example – A-DHC-4

If counter evidence was identified regarding the sensor_data.attitude#1 value failure, then an example of an interim treatment may be to enter a state of operational pause (i.e. ground the aircraft) until such a stage as the investigation is carried out. If for example the investigation reveals that the checks carried out by the PFCC miss certain sub-classes of the value failure, then the interim treatment may be amended to return to flying based on additional serviceability checks of sensors against the specific type of failure to reduce opportunity of the defence to miss this type of failure. The permanent design treatment may be a design change to the monitor software to accommodate handling of the additional fault.

4.8.14 Step 14 – Retire System

Retire system when operation of the system is no longer required, or when risk reduction is no longer practical or cost effective.

While retirement of aviation systems, particularly military aviation systems is political rather than a safety decision, impractical risk reduction or prolonged risk exposure are two factors influencing decisions to retire aviation systems.

4.9 Summary

This chapter has explored the role of architecture for achieving fail safe design and thus providing knowledge of product behaviours. Specifically, properties of architecture have been identified in relation to protecting against systematic faults and failures through layers of defences. Architectures and treatments of systematic faults and failures in a number of actual aviation systems have been examined. The results of this examination have been contrasted with the fail safe design criteria, and general properties of the treatment of systematic faults and failures identified.

The general properties of architecture that contribute to knowledge of product behaviours have identified and categorised, thus satisfying Principle C of Figure 10 for behaviours that emerge from architecture. Meta-arguments have been used to document the rationale of how architectural properties contribute to knowledge of product behaviours (thus also satisfying Principle X from Figure 10).

Using the identified properties, and examining how these factors contribute to bounding the uncertainty of the effects of systematic behaviours, the ASAL framework has been proposed. The assurance framework provides a measure of the system’s fault tolerance against systematic faults and failures, and thus infers the system’s suitability for use in the presence of aircraft level failure conditions of differing severities. An example was used to illustrate the process of applying the ASAL framework.

The ASAL framework has been developed to adhere to the usability guidelines identified in Figure 10. The ASAL framework minimises variability (adhering to Guideline 1) by specifying deterministic requirements for layers of defence, defence in depth, and coverage of propagation paths. These requirements are derived from general principles, and have been validated by benchmarking against a study of actual systems. The ASAL framework minimises subjectivity (adhering to Guideline 2) with respect to architectural assessment by ensuring that the ASAL requirements are numerically measurable based on the existence of defences and their role in providing defence in depth. This measurability to minimise subjectivity also permits adherence to Guideline 3 as assessors can distinguish acceptable from non-acceptable.

5 Assurance of Product Behavioural Knowledge

This chapter is concerned with achieving knowledge of product behaviours in relation to the defences of Chapter 4. The term product behaviour is used to mean the behaviours the realised product actually has. Product behaviours differ from requirements in that a requirement is an aspirational behaviour for the product established by the requirements and development processes, whereas the product behaviour is the actual behaviour the product exhibits. In practice, developers strive for these to be consistent, but there are factors that cause inconsistencies. Requirements satisfaction is the term applied to assuring consistency between requirements and product behaviours.

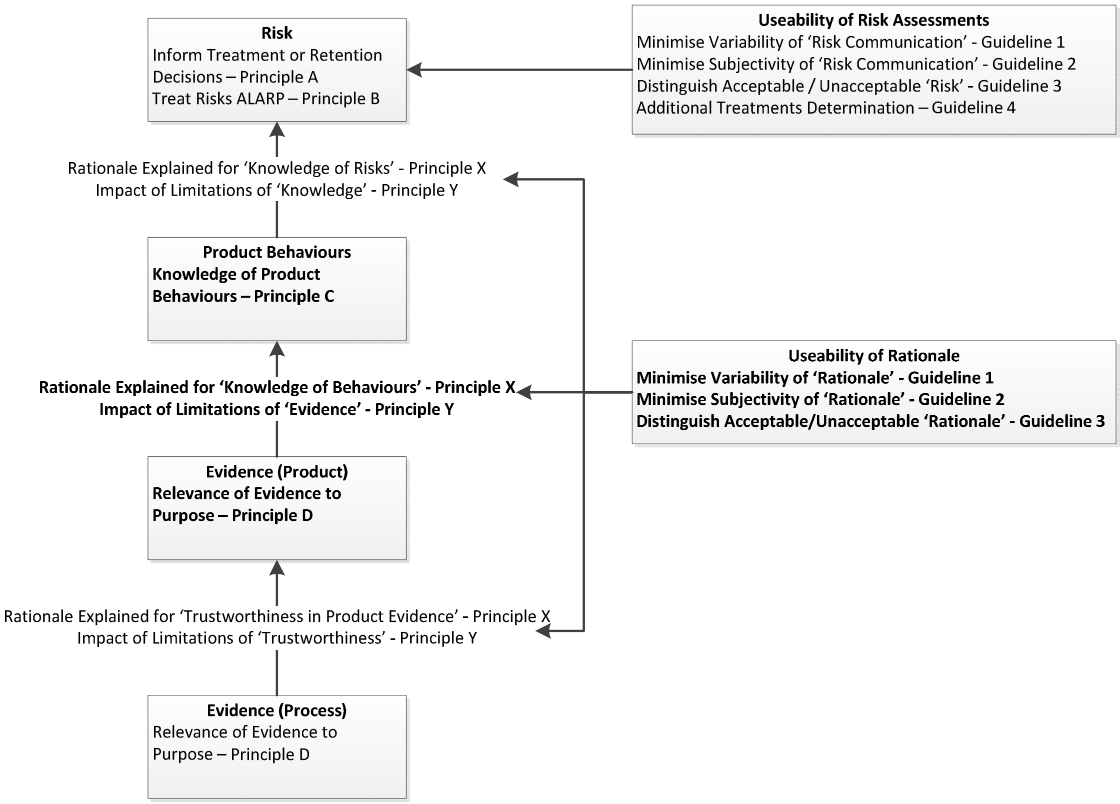
From the outset, it should be acknowledged that the knowledge of product behaviours of a defence will never be absolute. This is because the knowledge of product behaviours is dependent on factors such as:

- the extent to which sources of behavioural influence for the product are examined (i.e. where to look), and
- how systematically each source of behavioural influence for software is examined (i.e. how hard to look).

Where to examine and how hard to examine are influenced not only by the strategy employed, but also the cost and resources available to do so (refer Chapters 2 and 3). However absolute completeness of knowledge of product behaviours is not necessary to achieve safety. Not all behaviours have a bearing on safety, and some behaviours matter more than others. Thus, the goal for knowledge of product behaviours that is less than complete or absolute knowledge, and the emphasis is ‘enough’ knowledge of behaviours that impact safety. Determining the rationale for ‘enough’ knowledge about each defence is the topic of this chapter.

The factors of “where to look” and “how hard to look” are important, but are not in themselves outcomes. The outcomes are the extent of the knowledge of product behaviours, and the uncertainty that that knowledge is valid. This implies that any knowledge is always caveated by the uncertainty of its validity, and the bearing this uncertainty has on the conclusions established about product behaviours. Hence it is important that the uncertainty in knowledge of product behaviours remains explicit. This may be achieved through the concept of assurance, which provides a way of dealing with uncertainty and its inverse, confidence.

In Section 1.2 the concept of assurance was introduced and defined as adequate confidence and evidence that safety requirements have been met. The concept of assurance can be used to provide confidence that the knowledge of product behaviours of a defence is sufficient to reason about their suitability for informing the knowledge of risks. This chapter will examine how assurance of product behavioural knowledge can be achieved. In Chapter 3, a model containing principles and usability criteria was established. This chapter explores the assurance of product behavioural knowledge for defences. This chapter focusses on the principles shown in bold italics in Figure 32.



Risk
Inform Treatment or Retention Decisions – Principle A
Treat Risks ALARP – Principle B
Useability of Risk Assessments
Minimise Variability of 'Risk Communication' - Guideline 1
Minimise Subjectivity of 'Risk Communication' - Guideline 2
Distinguish Acceptable / Unacceptable 'Risk' - Guideline 3
Additional Treatments Determination – Guideline 4
Rationale Explained for 'Knowledge of Risks' - Principle X
Impact of Limitations of 'Knowledge' - Principle Y
Product Behaviours
Knowledge of Product Behaviours – Principle C
Useability of Rationale
Minimise Variability of 'Rationale' - Guideline 1
Minimise Subjectivity of 'Rationale' - Guideline 2
Distinguish Acceptable/Unacceptable 'Rationale' - Guideline 3
Rationale Explained for 'Knowledge of Behaviours' - Principle X
Impact of Limitations of 'Evidence' - Principle Y
Evidence (Product)
Relevance of Evidence to Purpose – Principle D
Rationale Explained for 'Trustworthiness in Product Evidence' - Principle X
Impact of Limitations of 'Trustworthiness' - Principle Y
Evidence (Process)
Relevance of Evidence to Purpose – Principle D

Figure 32: Key Principles/Usability Criteria of Safety Assurance for Product Behavioural Knowledge

Sections 5.1 through 5.3 of this chapter examine how knowledge of behaviours of products can be obtained through examination of a product and its lifecycle products (to satisfy Principle C). The role of lifecycle products is examined and a categorisation and hierarchy of lifecycle products is defined. A set of attributes of lifecycle products is also established (to satisfy Principle D). The rationale for how lifecycle products contribute to knowledge of product behaviours is described using meta-arguments (to satisfy Principle X). The effect on knowledge of limitations is also examined (to satisfy Principle Y). Finally, the rationale for satisfying these principles is instantiated in Sections 5.4 and 5.5, through the definition of an product behavioural assurance framework (to satisfy Guidelines 1, 2 and 3).

5.1 The Rationale for Knowledge of Product Behaviours

Deciding on what claims to make about a product and its evidence is a challenge. Established practice also reveals that there is a lack of consistency in ways to do this, each with differing benefits and limitations (refer to (Weaver, 2003)). In Section 3.5 the role of argumentation for demonstration of rationale was discussed. Specifically, the motivation to pre-constrain parts of the argument capturing rationale was identified. Based on this aspiration to constrain relevant aspects of the argument is it possible to:

- establish a set of behavioural attributes of a product defence, and associated evidence requirements for them?, and
- qualify the extent of knowledge of product behaviours with respect to the defence?

5.1.1 What Should Product Behavioural Claims be About?

Chapter 4 has established that knowledge of product behaviours is required to answer questions about architectural fault prevention and fault tolerance which in turn are used to inform risk assessments. Chapter 4 also provides a means for establishing the collective adequacy of fault prevention and fault tolerance mechanisms, presuming there is knowledge of them. Chapter 4 asserts that layers of fault prevention (absence) assertions and/or fault tolerance (detection and handling) mechanisms can be used to provide assurance that systematic faults do not lead to unacceptable failure conditions.

Thus for each defence it is necessary to examine the product and its evidence to determine if the defence achieves its role in architectural assurance. This chapter examines a means of examining the knowledge of the product behaviours of defences.

5.1.2 The Concept of a Constraint

Whether the specific architectural defence uses fault prevention or tolerance is dependent on the specific fault, its propagation path and the architecture of the overall system. The layered treatment requires that for a given fault, the behaviour of the system will be constrained to either prevent or tolerate the fault.

Using this pattern, each fault prevention (absence) assertion or fault tolerance (detection and handling) mechanisms can be transposed to a specified ‘constraint’ requirement on the behaviour of the system and its software. This transposition of meaning is expressed in GSN in Figure 33.

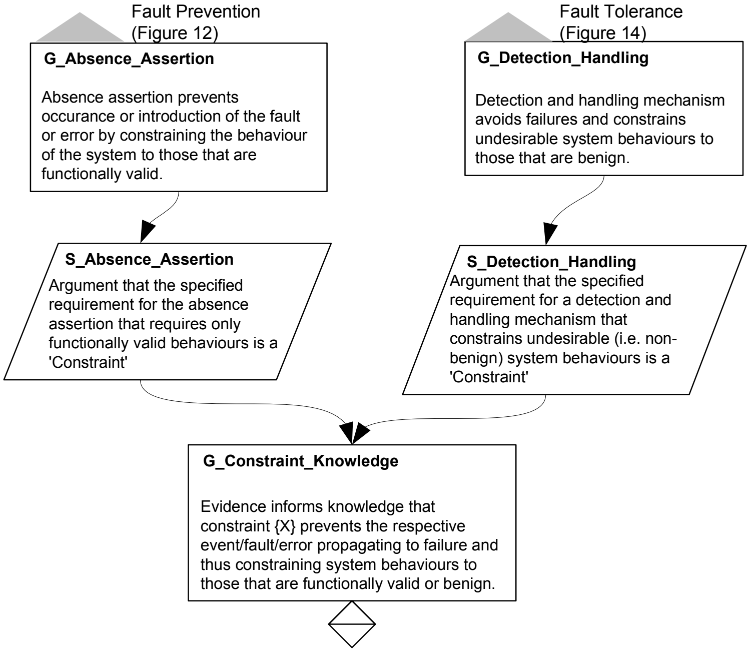

Fault Prevention
(Figure 12)
G_Absence_Assertion
Absence assertion prevents occurance or introduction of the fault or error by constraining the behaviour of the system to those that are functionally valid.
Fault Tolerance
(Figure 14)
G_Detection_Handling
Detection and handling mechanism avoids failures and constrains undesirable system behaviours to those that are benign.
S_Absence_Assertion
Argument that the specified requirement for the absence assertion that requires only functionally valid behaviours is a 'Constraint'
S_Detection_Handling
Argument that the specified requirement for a detection and handling mechanism that constrains undesirable (i.e. non-benign) system behaviours is a 'Constraint'
G_Constraint_Knowledge
Evidence informs knowledge that constraint {X} prevents the respective event/fault/error propagating to failure and thus constraining system behaviours to those that are functionally valid or benign.

Figure 33: Constraints on Behaviours

In general terms, any requirement is a 'constraint' on the behaviour of the system. The focus here is on constraints to do with fault prevention or tolerance.

5.2 Knowledge from Lifecycle Products

There are at least two possible ways to establish knowledge of the behaviours of an engineered product, such as software, electronic hardware or mechanical systems. One is to directly examine (analytically or empirically) the product itself. For example:

- for software: the source code or executable object code,
- for electronic hardware: the physical electrical circuit, and
- for mechanical systems: the interactions between gears, linkages, etc.

However, designers realise that determining the acceptability of behaviours of an engineering product solely on the basis of an examination of the product is difficult. For complex technologies (e.g. software and complex electronic hardware), it remains difficult even with the advent of specialist tools. This is because the behaviours of the product will not be evident without additional information to guide the examination.

Because of these difficulties, most contemporary assurance approaches for complex technologies don't try to reverse engineer a product's behaviours, albeit there are exceptions[27]. Instead they rely on an examination of a product and its lifecycle

[27] Exceptions include the US Nuclear Code, which examines of source code and object code, and UK MoD initial acceptance of C-130J Mission Computer Software, which was subject to static code analysis.

evidence, herein referred to as lifecycle products. The following sub-sections examine how product behaviours can be elicited from lifecycle products and their evidence.

5.2.1 Categorisation of Lifecycle Products

If the typical lifecycle products are examined, such as for real project evidence (refer Chapter 10), it is evident that the typical lifecycle products include:

- requirements,
- refined and detailed design requirements[28],
- human readable logical implementation (e.g. source code) / model[29],
- machine readable logical implementation code (executable object code / binary code),
- physical implementation / product,
- verification results of product with respect to:
    - logical implementation / model,
    - refined and design requirements, and
    - requirements,
- validation results of:
    - product,
    - logical implementation / model,
    - refined and design requirements, and
    - requirements.

Note that it doesn't matter whether the systems engineering development lifecycle is grand design, waterfall, spiral, or certain instantiations of the agile paradigm, because:

- despite good intentions, in practice some lifecycle evidence always lags the product due to project resourcing and scheduling, irrespective of lifecycle model;
- in real world projects there are typically two phases of evidence production: the prototyping phase, and after 'measurement' commences; the identified lifecycle products usually exist once measurement commences as a way of packaging evidence for certification authorities;

[28] at one or more levels of refinement or abstraction; maybe also referred to as technical requirements

[29] e.g. source code in one or more languages, or an abstract implementation language/model such as those used in model based engineering

- the order of production of the lifecycle products is not important to defining attributes of them; and
- holistically, the individual lifecycle products are broadly consistent to these categories irrespective of the paradigm under which they were developed.

There is sentiment, as evidenced by (RTCA Inc., 2012), that methods such as model-based engineering undermine categories based on lifecycles in the way they undermined standards that are based on a specific lifecycle. In these cases, model-based engineering simply changes the sources of evidence for lifecycle products from human centric processes to tools. The evidence should still exist; it just takes a different form depending on the construct of the tool.

Each of the above mentioned lifecycle products are uniquely definable in terms of expected content. While there is evidence of variations in document sets and structures across projects (examined in Chapter 10), it is possible to identify data pertaining to each above mentioned category of lifecycle products irrespective of which physical document it exists in.

Consider a software development, for example:

- System Requirements Data is often documented in a System Specification (SS),
- Sub-System Requirements Data is often documented in a Sub-System Specification (SSS),
- Software Requirements Data (SRD) is often documented in a Software Requirements Specification (SRS),
- refined requirements and design description are often documented in subordinate parts of the SRS, and/or in a Software Design Description (SDD) and/or Interface Design Document (IDD);
- low level requirements and detailed design are often documented in the SDD or as annotations to code functions;
- source code is usually stored in source code files; and
- software verification and validation (V&V) procedures, cases and results are often documented in:
    - o the Software Verification Plan (SVP) or Software Test Plan (STP),
    - o Software Verification Description (SVD) or Software Test Description (STD), and

  - Software Verification Report (SVR) or Software Test Report (STR) respectively;
- Sub-system and system verification and validation produces, cases and results are often documented in:
  - V&V plans, or test plans,
  - V&V descriptions or test descriptions,
  - V&V reports.

The proliferation of developments adhering to the legacy DoD-STD-2167A (United States Department of Defense, 1988), the superseding MIL-STD-498 (US DoD, 1994), or J-STD-016 (IEEE Computer Society, 1995) and ISO/IEC12207 (ISO/IEC, 2008)) results in reasonable consistency between developments, even across different nations.

While there is also variation in the names and structures of these documents, there is consistency in content as illustrated by Section 11 of (RTCA Inc., 1992) which may be exploited in establishing knowledge of product behaviours from lifecycle products. Figure 34 graphically illustrates the hierarchy of lifecycle products. At the top-most level the behaviour is related to a Product Defence and 'constraint' to continue the line of reasoning from Chapter 4 and Section 5.1. Lifecycle product evidence may either pertain to a lifecycle product directly (e.g. analysis evidence of detailed design requirements), or pertain to the relationship between lifecycle products (e.g. verification of the software product against detailed design requirements).

5.2.2 Understanding the Relationship between Knowledge and the Chain of Evidence

From a safety assurance perspective it is important to establish the relationship between the extent to which the hierarchy of evidence preserves the chain of evidence, as this influences the knowledge and uncertainty that may be inferred.

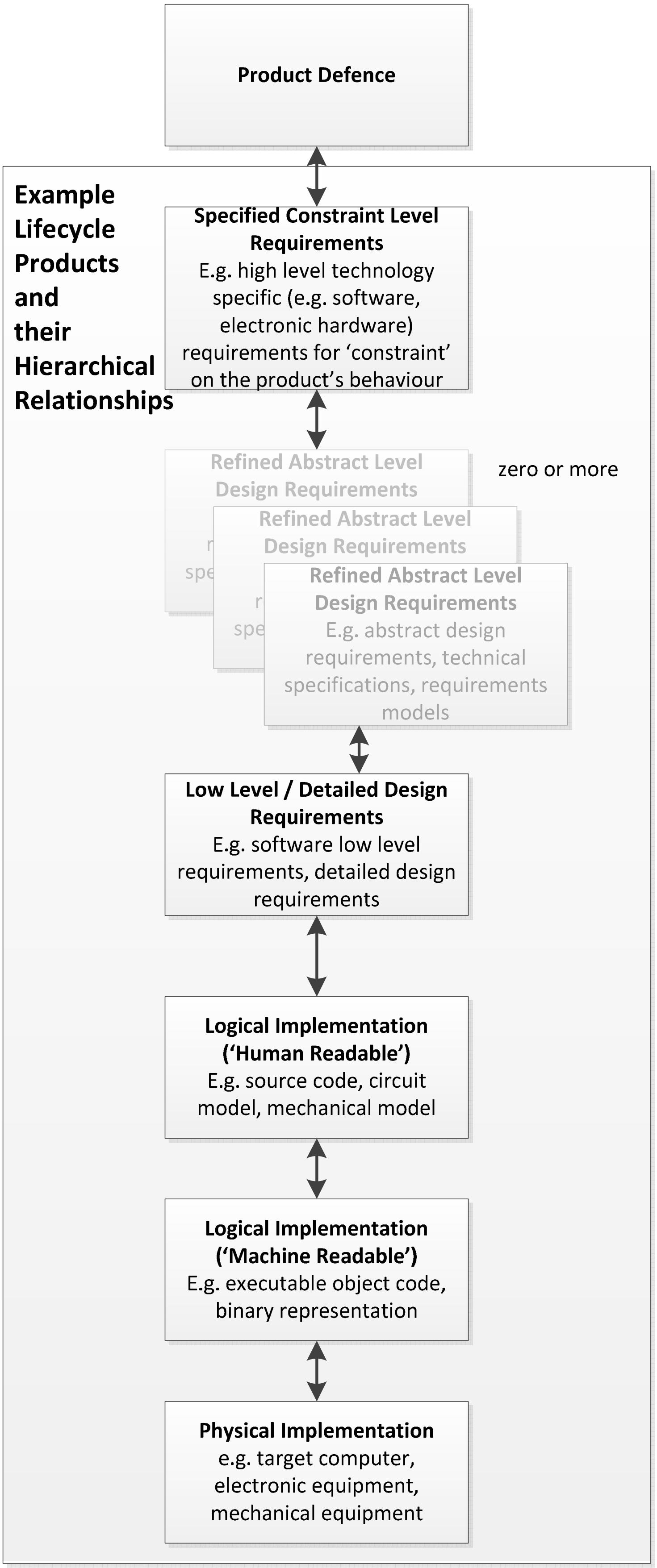

Product Defence
Example Lifecycle Products and their Hierarchical Relationships
Specified Constraint Level Requirements
E.g. high level technology specific (e.g. software, electronic hardware) requirements for 'constraint' on the product's behaviour
Refined Abstract Level Design Requirements
Refined Abstract Level Design Requirements
Refined Abstract Level Design Requirements
E.g. abstract design requirements, technical specifications, requirements models
zero or more
Low Level / Detailed Design Requirements
E.g. software low level requirements, detailed design requirements
Logical Implementation ('Human Readable')
E.g. source code, circuit model, mechanical model
Logical Implementation ('Machine Readable')
E.g. executable object code, binary representation
Physical Implementation
e.g. target computer, electronic equipment, mechanical equipment

Figure 34: Lifecycle Products and their Hierarchical Relationships

Logically it follows that for cases where limitations in the knowledge of behaviours of the product are intolerable, then limitations or discontinuities in the chain of evidence between specified constraint level requirements, and the physical and logical implementation of the constraint can't be tolerated. In these cases it is necessary to present evidence for each lifecycle product and relationship in the hierarchy. This rationale is represented by the left hand branch of Figure 35.

However, for cases where limitations in the knowledge of the behaviours can be tolerated, then there are additional considerations. In cases where limitations in the level of knowledge sought should be constrained such that they don't undermine the chain of evidence, then limitations in the hierarchy of lifecycle products may be tolerated provided they don't introduce a discontinuity into the chain of evidence. In these cases, it is necessary to present evidence for a minimum set of lifecycle products such that discontinuities in the traceability of the chain of evidence are avoided. This rationale is represented by the central branch of Figure 35.

For example, if evidence was presented with the exception that no evidence is presented for any refined or abstract requirements. In this case there would also be no evidence of V&V of the logical and physical implementation of refined abstraction requirements, and there would be no verification of low-level requirements against refined or abstract requirements. Therefore in order to avoid a discontinuity in the chain of evidence, then verification of low level requirements would be required with respect specified constraint level requirements to preserve the chain of evidence. The key factor is that the chain of evidence does not become discontinuous.

Finally, in cases where limitations in the level of knowledge can be tolerated because the safety impact is not severe, then limitations in the hierarchy of lifecycle products may be tolerated. In these cases, it is necessary to at least present evidence that the constraint and implementation exists, else there is no product defence. This rationale is represented by the right hand branch of Figure 35.

In essences Figure 35 suggests that there are three categories of assurance: those where limitations are intolerable, those where the limitations are constrained and those where they are tolerable. This concept will be used as the basis for a product behavioural assurance framework which is described in Section 5.4.

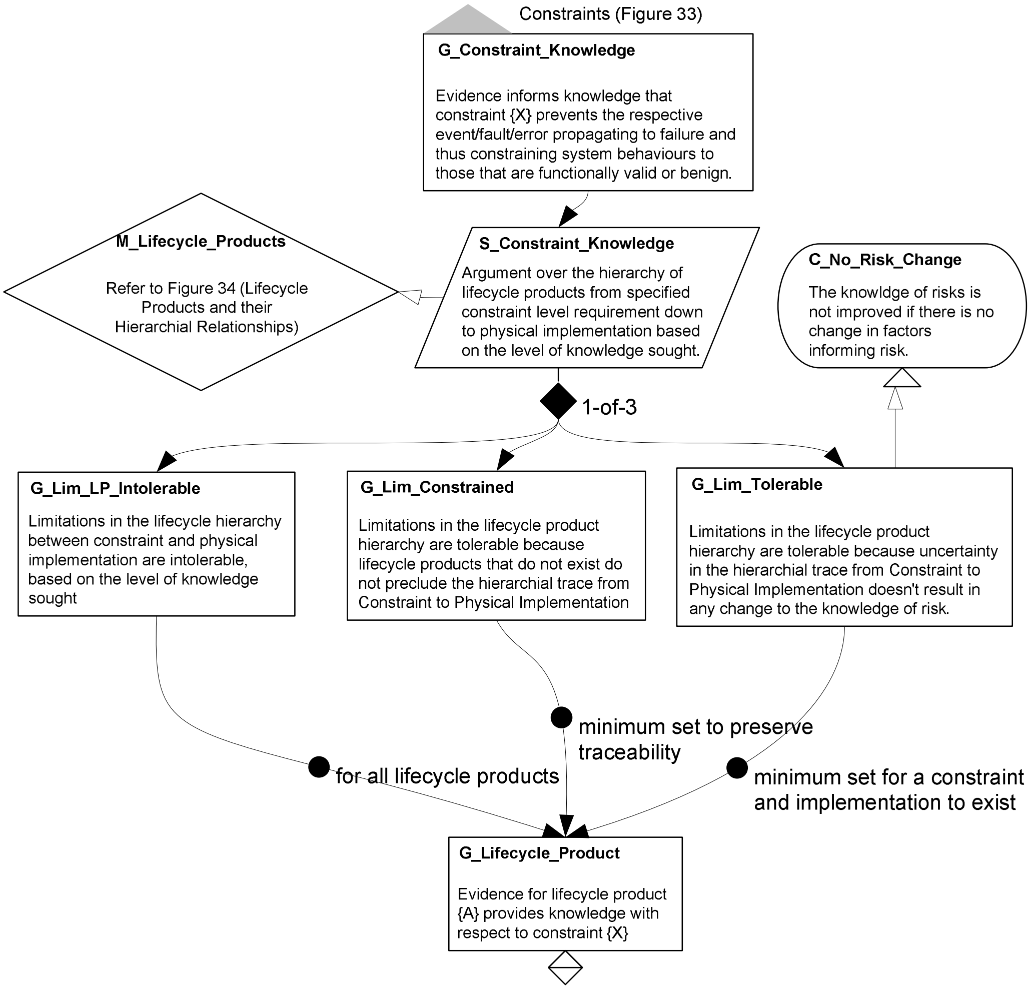

Constraints (Figure 33)
G_Constraint_Knowledge
Evidence informs knowledge that constraint {X} prevents the respective event/fault/error propagating to failure and thus constraining system behaviours to those that are functionally valid or benign.
M_Lifecycle_Products
Refer to Figure 34 (Lifecycle Products and their Hierarchial Relationships)
S_Constraint_Knowledge
Argument over the hierarchy of lifecycle products from specified constraint level requirement down to physical implementation based on the level of knowledge sought.
C_No_Risk_Change
The knowldge of risks is not improved if there is no change in factors informing risk.
1-of-3
G_Lim_LP_Intolerable
Limitations in the lifecycle hierarchy between constraint and physical implementation are intolerable, based on the level of knowledge sought
G_Lim_Constrained
Limitations in the lifecycle product hierarchy are tolerable because lifecycle products that do not exist do not preclude the hierarchial trace from Constraint to Physical Implementation
G_Lim_Tolerable
Limitations in the lifecycle product hierarchy are tolerable because uncertainty in the hierarchial trace from Constraint to Physical Implementation doesn't result in any change to the knowledge of risk.
minimum set to preserve traceability
for all lifecycle products
minimum set for a constraint and implementation to exist
G_Lifecycle_Product
Evidence for lifecycle product {A} provides knowledge with respect to constraint {X}

Figure 35: Knowledge from Lifecycle Product Hierarchy

5.3 Establishing the Attributes of a Lifecycle Products

Section 5.2 requires that evidence be presented about lifecycle products, but what evidence should be presented, how should it be organised, and what should it show? This section examines how attributes of lifecycle products are defined to address this.

5.3.1 Defining Categories of Attributes of Lifecycle Products

Section 2.4 established the concepts of requirements validity, requirements satisfaction and requirements traceability as the core concepts of assurance of safety. While these concepts feature the word requirements, in terms of lifecycle products, they are relevant to all lifecycle products. This is because for lifecycle products not directly about requirements they will provide evidence of the satisfaction or traceability of requirements. Herein, satisfaction means not only the verification of the requirement, but also the means by which the lifecycle product specifies the satisfying behaviour. These concepts can be used to define categories of attributes about lifecycle products.

Figure 36 presents a set of general attributes for lifecycle products. The attributes are grouped as follows:

- *Existence*: attributes that characterise the existence of the lifecycle product;
- *Specification*: attributes that characterise the specification of the behaviour of lifecycle product and how that behaviour related to both high level and low level abstraction of the behaviour;
- *Verification*: attributes that characterise the verification of the behaviour of lifecycle product with respect to higher level abstractions of the behaviour; and
- *Validation*: attributes that characterise the validity of the behaviour both in terms of safety objectives and other behaviours of the system.

Independent of the grouping of the attributes, there is an extra-group attribute which is relevant to each attribute in each group. In Figure 36 it appears as 'Inadequacies Resolved'. Establishment of the other attributes may reveal inadequacies in this lifecycle product, and unless these are resolved, the articulation of the relevant behaviour by this lifecycle product will be incorrect. As development is not an instantaneous activity, there will always be inadequacies to resolve for at least some lifecycle products. This attribute pertains to evidence that the inadequacies in evidence of other attributes has been resolved to the extent necessary to avoid a gap or discontinuity in evidence if one cannot be tolerated.

The attribute groups of each lifecycle product contain two types of attributes: self and inter-relational. Self attributes are attributes which deal with internal properties of the relevant lifecycle product. Inter-relational attributes deal with the relationships within the hierarchy of lifecycle products and are predominantly concerned with traceability, and technical agreement (compliance and robustness) between lifecycle products.

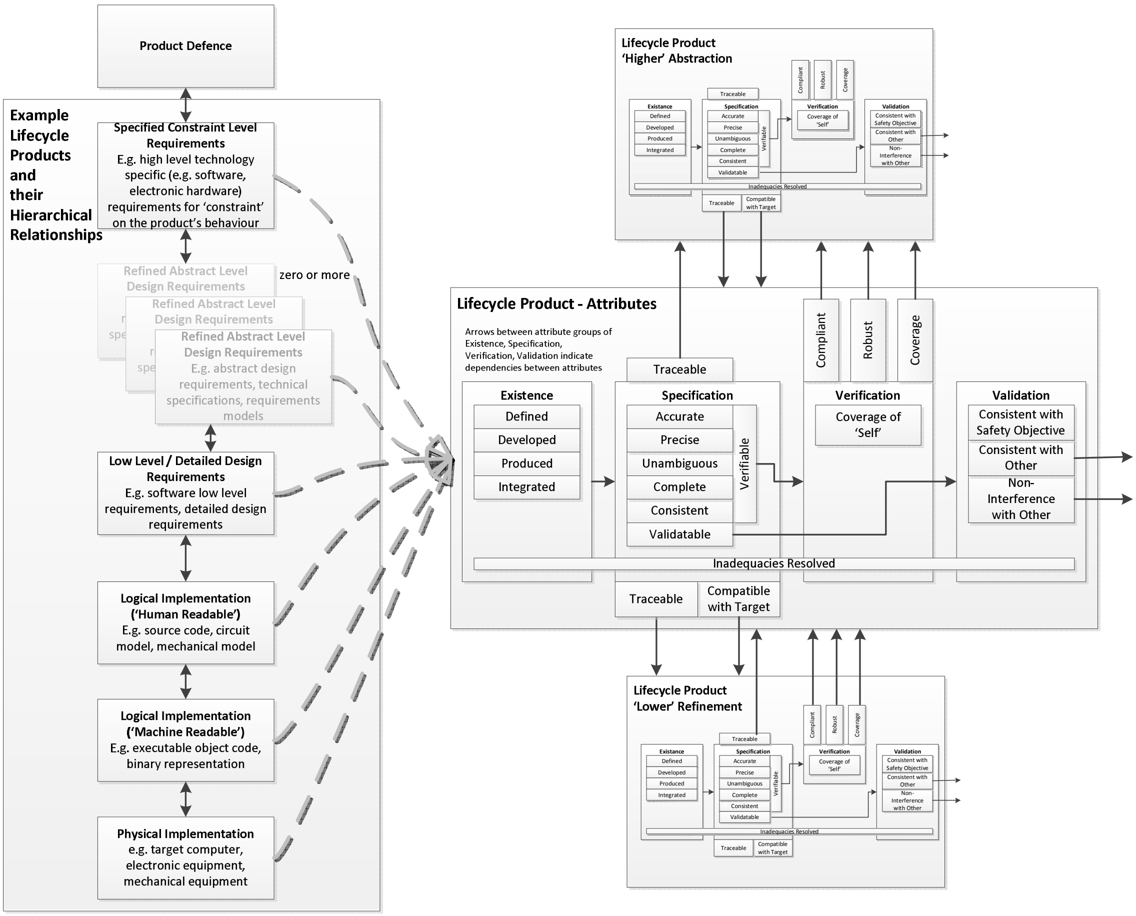

Product Defence
Example Lifecycle Products and their Hierarchical Relationships
Specified Constraint Level Requirements
E.g. high level technology specific (e.g. software, electronic hardware) requirements for 'constraint' on the product's behaviour
zero or more
Refined Abstract Level Design Requirements
Refined Abstract Level Design Requirements
Refined Abstract Level Design Requirements
E.g. abstract design requirements, technical specifications, requirements models
Low Level / Detailed Design Requirements
E.g. software low level requirements, detailed design requirements
Logical Implementation ('Human Readable')
E.g. source code, circuit model, mechanical model
Logical Implementation ('Machine Readable')
E.g. executable object code, binary representation
Physical Implementation
e.g. target computer, electronic equipment, mechanical equipment
Lifecycle Product 'Higher' Abstraction
Lifecycle Product - Attributes
Arrows between attribute groups of Existence, Specification, Verification, Validation indicate dependencies between attributes
Traceable
Compliant
Robust
Coverage
Existence
Defined
Developed
Produced
Integrated
Specification
Accurate
Precise
Unambiguous
Complete
Consistent
Validatable
Verifiable
Verification
Coverage of 'Self'
Validation
Consistent with Safety Objective
Consistent with Other
Non-Interference with Other
Inadequacies Resolved
Traceable
Compatible with Target
Lifecycle Product 'Lower' Refinement

Figure 36: Lifecycle Product – Attributes

Evidence must be presented for each attribute group. Because there are dependencies between attribute groups and lifecycle products, then if evidence is missing there may be a break in the chain of evidence in the hierarchy. This rationale is described by Figure 37, which specifies that evidence is required for each attribute group.

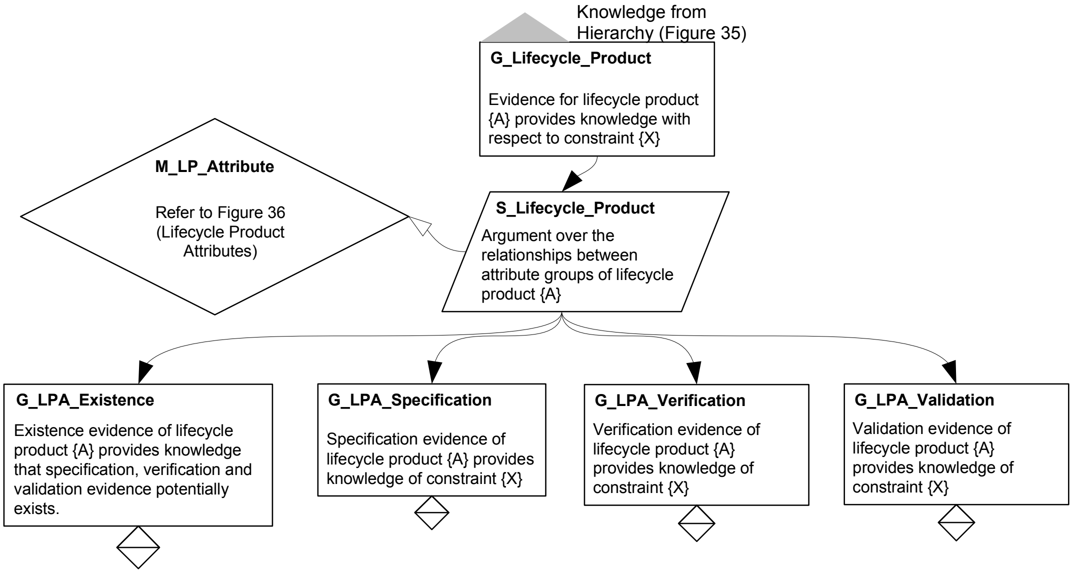

Knowledge from Hierarchy (Figure 35)
G_Lifecycle_Product
Evidence for lifecycle product {A} provides knowledge with respect to constraint {X}
M_LP_Attribute
Refer to Figure 36 (Lifecycle Product Attributes)
S_Lifecycle_Product
Argument over the relationships between attribute groups of lifecycle product {A}
G_LPA_Existence
Existence evidence of lifecycle product {A} provides knowledge that specification, verification and validation evidence potentially exists.
G_LPA_Specification
Specification evidence of lifecycle product {A} provides knowledge of constraint {X}
G_LPA_Verification
Verification evidence of lifecycle product {A} provides knowledge of constraint {X}
G_LPA_Validation
Validation evidence of lifecycle product {A} provides knowledge of constraint {X}

Figure 37: Attribute Groups of Lifecycle Products

The following sub-sections examine each of these attribute groups in greater detail.

5.3.2 Existence Attribute Group

There are four attributes within the existence attribute group, all of which are 'self-attributes': developed, defined, produced and integrated. The attributes are defined as follows:

- ***Defined***: evidence of the definition of the constraint within the lifecycle product;
- ***Developed***: evidence that derived or refined behaviours have been developed appropriate to the abstraction of the lifecycle product;
- ***Produced***: evidence of the implementation of the constraint; and
- ***Integrated***: evidence that the lifecycle product has been integrated into the physical implementation to form part of the product.

Evidence should be presented for each of these attributes as shown in Figure 38.

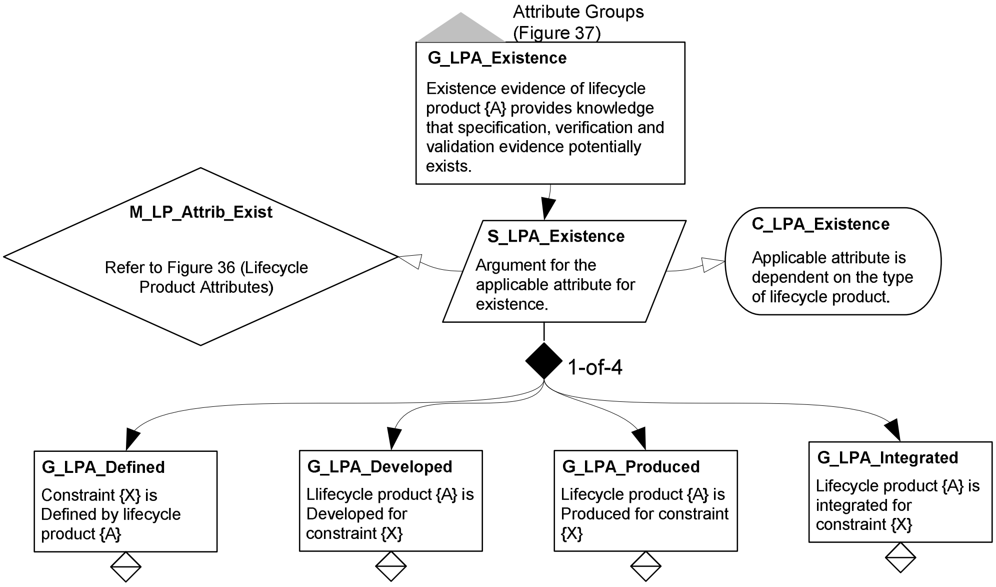

Attribute Groups
(Figure 37)
G_LPA_Existence
Existence evidence of lifecycle product {A} provides knowledge that specification, verification and validation evidence potentially exists.
M_LP_Attrib_Exist
Refer to Figure 36 (Lifecycle Product Attributes)
S_LPA_Existence
Argument for the applicable attribute for existence.
C_LPA_Existence
Applicable attribute is dependent on the type of lifecycle product.
1-of-4
G_LPA_Defined
Constraint {X} is Defined by lifecycle product {A}
G_LPA_Developed
Llifecycle product {A} is Developed for constraint {X}
G_LPA_Produced
Lifecycle product {A} is Produced for constraint {X}
G_LPA_Integrated
Lifecycle product {A} is integrated for constraint {X}

Figure 38: Attributes of Existence

5.3.3 Specification Attribute Group

The specification attribute group is used for presenting evidence of how well the behaviours are specified for the constraint at the level of abstraction of the lifecycle product. The specification attribute group contains both self-attributes as well as inter-relational attributes, as shown by Figure 39.

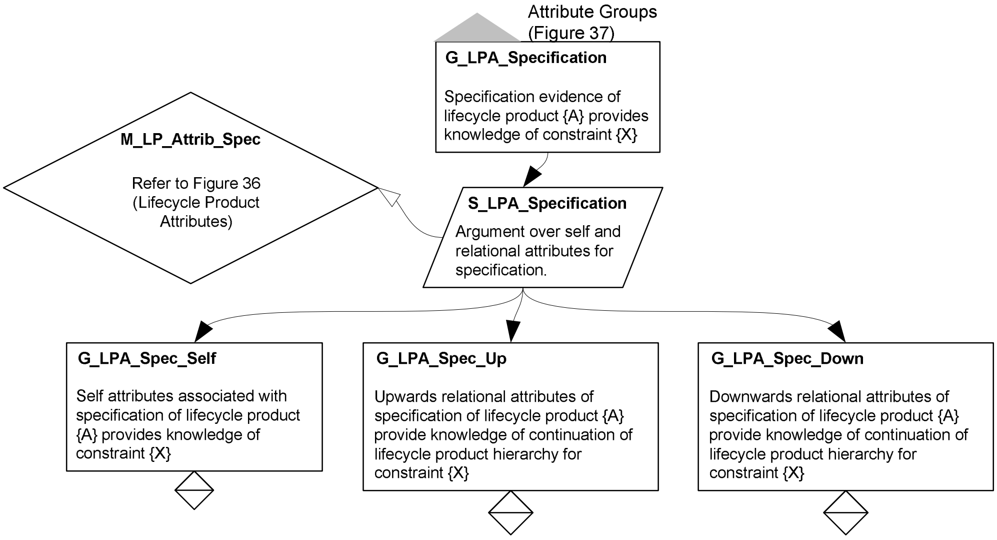

Attribute Groups
(Figure 37)
G_LPA_Specification
Specification evidence of lifecycle product {A} provides knowledge of constraint {X}
M_LP_Attrib_Spec
Refer to Figure 36 (Lifecycle Product Attributes)
S_LPA_Specification
Argument over self and relational attributes for specification.
G_LPA_Spec_Self
Self attributes associated with specification of lifecycle product {A} provides knowledge of constraint {X}
G_LPA_Spec_Up
Upwards relational attributes of specification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}
G_LPA_Spec_Down
Downwards relational attributes of specification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}

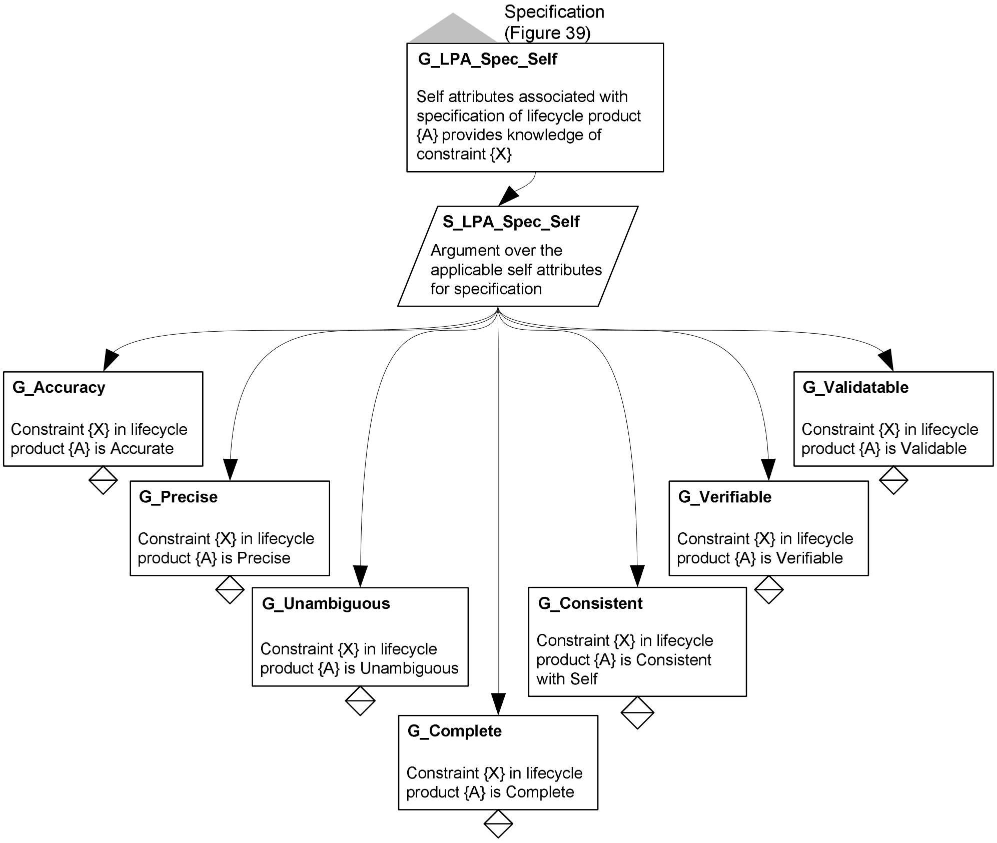
Figure 39: Attributes of Specification

Self Attributes

There are seven self attributes within the specification attribute group, as follows:

- ***Accurate***: evidence of the degree of closeness of specification of the constraint at this level of abstraction to the actual constraint;
- ***Precise***: evidence of the degree of closeness of the representation to physical values at this level of abstraction;
- ***Unambiguous***: evidence of avoidance of misinterpretation due to ambiguity in the specification;
- ***Complete***: evidence that the behavioural specification is not missing part of its specification for this level of abstraction;
- ***Consistent***: evidence that the behavioural specification is internally consistent, such that elements of the specification can be interpreted and evaluated consistently;
- ***Verifiable***: evidence that it is possible to verify the behaviour at this level of abstraction against higher levels of abstraction – if the behaviour can't be verified, then requirements satisfaction cannot be achieved; and
- ***Validatable***: evidence that the behaviour is validated at this level of abstraction against safety objectives and with respect to other behaviours of the system.

Evidence should be presented for each of these attributes as shown in Figure 40.



Specification (Figure 39)
G_LPA_Spec_Self
Self attributes associated with specification of lifecycle product {A} provides knowledge of constraint {X}
S_LPA_Spec_Self
Argument over the applicable self attributes for specification
G_Accuracy
Constraint {X} in lifecycle product {A} is Accurate
G_Precise
Constraint {X} in lifecycle product {A} is Precise
G_Unambiguous
Constraint {X} in lifecycle product {A} is Unambiguous
G_Complete
Constraint {X} in lifecycle product {A} is Complete
G_Consistent
Constraint {X} in lifecycle product {A} is Consistent with Self
G_Verifiable
Constraint {X} in lifecycle product {A} is Verifiable
G_Validatable
Constraint {X} in lifecycle product {A} is Validable

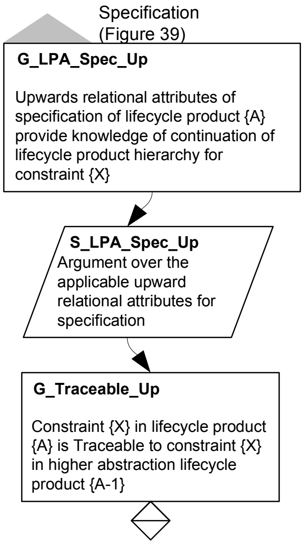
Figure 40: Self Attributes of Specification

Up Attributes

There is one upwards inter-relational attribute within the specification attribute group:

- ***Traceable to 'Higher' Abstractions:*** evidence that the specification is traceable to the relevant specification of next higher abstraction of lifecycle product to continue the chain of evidence.

Evidence should be presented for this attribute as shown in Figure 41.



Specification (Figure 39)
G_LPA_Spec_Up
Upwards relational attributes of specification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}
S_LPA_Spec_Up
Argument over the applicable upward relational attributes for specification
G_Traceable_Up
Constraint {X} in lifecycle product {A} is Traceable to constraint {X} in higher abstraction lifecycle product {A-1}

Figure 41: Up Attributes of Specification

Down Attributes

There are two downwards inter-relational attributes within the specification attribute group:

- ***Traceable to 'Lower' Abstractions:*** evidence that the specification is traceable to the relevant specification of the next lower abstraction of lifecycle product to continue the chain of evidence; and
- ***Compatible with Target:*** evidence that the specification is compatible with the intended logical and physical implementation, and that any behaviours introduced at this level that are inherited from logical or physical implementation are captured as a dependency for low level abstractions.

Evidence should be presented for these attributes as shown in Figure 42.

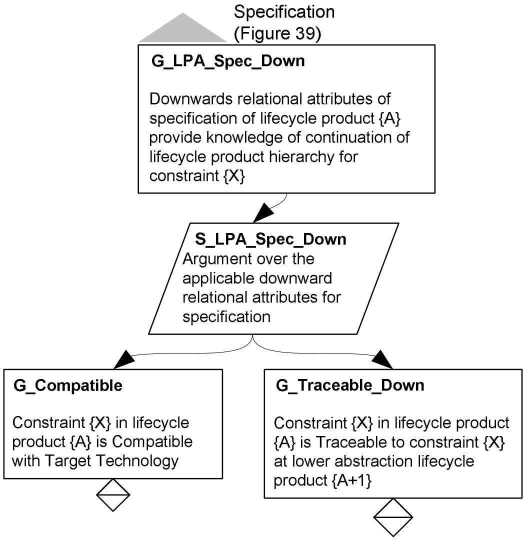

Specification
(Figure 39)
G_LPA_Spec_Down
Downwards relational attributes of specification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}
S_LPA_Spec_Down
Argument over the applicable downward relational attributes for specification
G_Compatible
Constraint {X} in lifecycle product {A} is Compatible with Target Technology
G_Traceable_Down
Constraint {X} in lifecycle product {A} is Traceable to constraint {X} at lower abstraction lifecycle product {A+1}

Figure 42: Down Attributes of Specification

5.3.4 Verification Attribute Group

The verification attribute group is used for presenting evidence as to how well the behaviours specified for the constraint have been verified with respect to high level abstractions. The verification attribute group contains both self-attributes as well as inter-relational attributes, as shown by Figure 43.

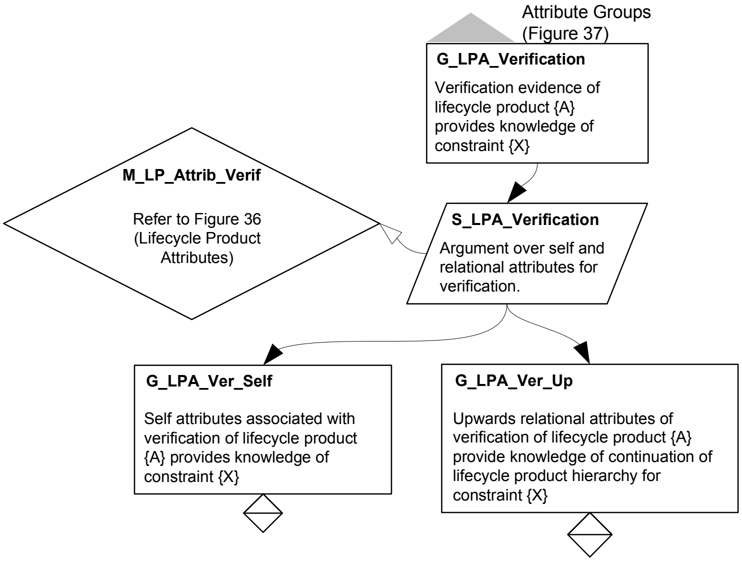

Attribute Groups
(Figure 37)
G_LPA_Verification
Verification evidence of lifecycle product {A} provides knowledge of constraint {X}
M_LP_Attrib_Verif
Refer to Figure 36 (Lifecycle Product Attributes)
S_LPA_Verification
Argument over self and relational attributes for verification.
G_LPA_Ver_Self
Self attributes associated with verification of lifecycle product {A} provides knowledge of constraint {X}
G_LPA_Ver_Up
Upwards relational attributes of verification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}

Figure 43: Attributes of Verification

Self Attributes

There is one self attribute within the verification attribute group:

- ***Coverage of 'Self':*** evidence of verification coverage of the behaviours.

Evidence should be presented for this attribute as shown in Figure 44.

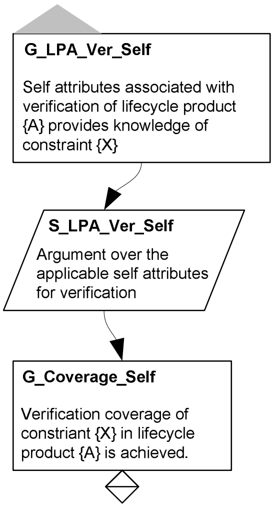

G_LPA_Ver_Self
Self attributes associated with verification of lifecycle product {A} provides knowledge of constraint {X}
S_LPA_Ver_Self
Argument over the applicable self attributes for verification
G_Coverage_Self
Verification coverage of constriant {X} in lifecycle product {A} is achieved.

Figure 44: Self Attributes of Verification

Up Attributes

There are three upwards inter-relational attributes within the verification attribute group, as follows:

- ***Compliant with 'Higher' Abstractions:*** evidence that the constraint at this level of abstraction is in technical agreement with the relevant behaviours at higher abstractions (positive perspective – when faults are absent from input data);

- ***Robust with 'Higher' Abstractions:*** evidence that the constraint at this level of abstraction is in technical agreement with the relevant behaviours at higher abstractions (negative perspective – when faults are present in input data); and
- ***Coverage of 'Higher' Abstraction:*** evidence of verification coverage of the traceable higher abstractions of the constraint by verification in this lifecycle product.

Evidence should be presented for these attributes as shown in Figure 45.

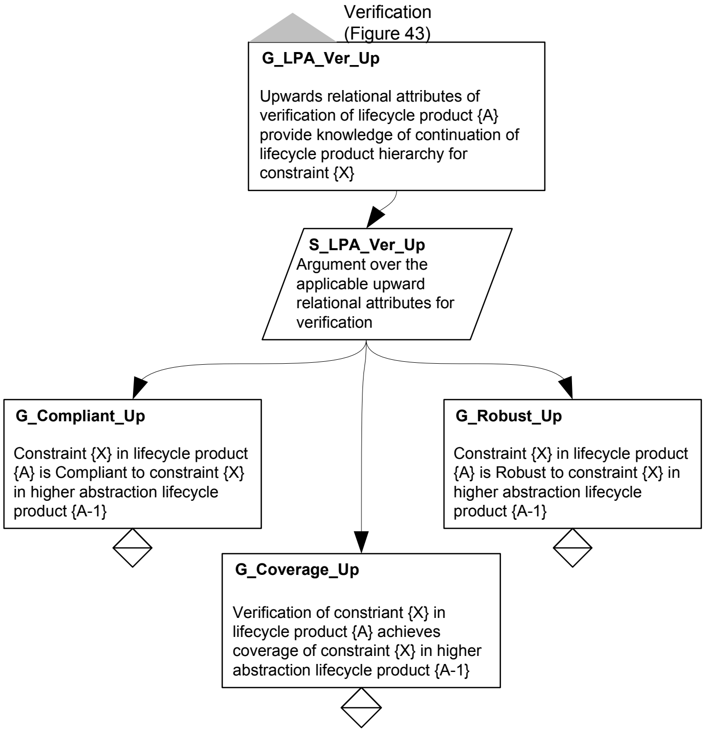

Verification
(Figure 43)
G_LPA_Ver_Up
Upwards relational attributes of verification of lifecycle product {A} provide knowledge of continuation of lifecycle product hierarchy for constraint {X}
S_LPA_Ver_Up
Argument over the applicable upward relational attributes for verification
G_Compliant_Up
Constraint {X} in lifecycle product {A} is Compliant to constraint {X} in higher abstraction lifecycle product {A-1}
G_Robust_Up
Constraint {X} in lifecycle product {A} is Robust to constraint {X} in higher abstraction lifecycle product {A-1}
G_Coverage_Up
Verification of constriant {X} in lifecycle product {A} achieves coverage of constraint {X} in higher abstraction lifecycle product {A-1}

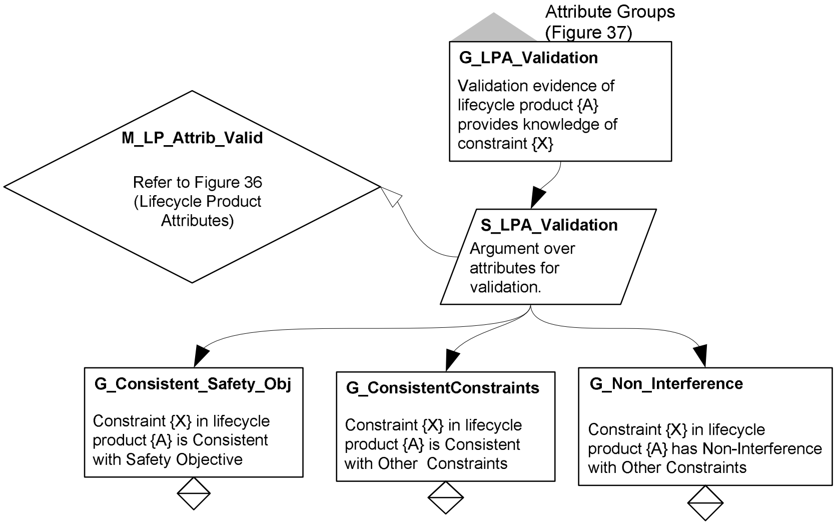
Figure 45: Up Attributes of Verification

5.3.5 Validation Attribute Group

The validation attribute group is used for presenting evidence of the validity of the behaviours specified for the constraint at the applicable level of abstraction of the lifecycle product with respect to both safety objectives and other constraints and behaviours of the system. The validation attribute group contains attributes, as shown by Figure 46.

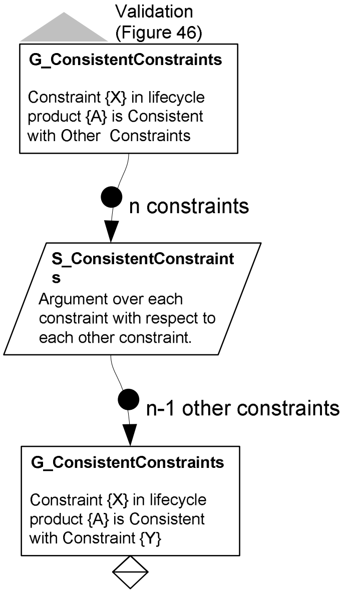
- ***Consistent with Safety Objectives:*** evidence that the behaviours specified in relation to the constraint at the level of abstraction of this lifecycle product are consistent with the safety objectives for the overall system and the behaviours necessary for the system to achieve those safety objectives;

- ***Consistent with Constraints:*** evidence that the behaviours specified in relation to the constraint are consistent with other constraints placed on the system by architectural assurances (refer Figure 47); and
- ***Non-Interference with Other:*** evidence that the behaviours specified in relation to the constraint do not interfere with other constraints placed on the system through physical or logical partitioning, or acceptable interactions (refer Figure 48).



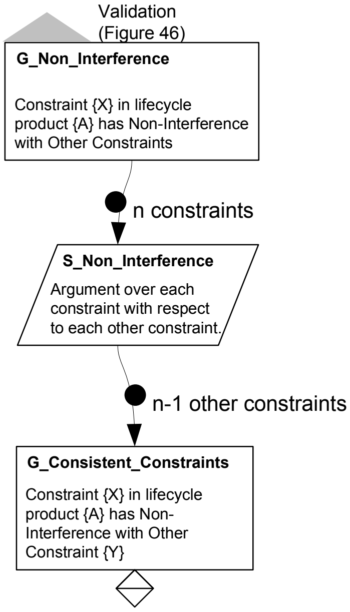
Attribute Groups (Figure 37)
G_LPA_Validation
Validation evidence of lifecycle product {A} provides knowledge of constraint {X}
M_LP_Attrib_Valid
Refer to Figure 36 (Lifecycle Product Attributes)
S_LPA_Validation
Argument over attributes for validation.
G_Consistent_Safety_Obj
Constraint {X} in lifecycle product {A} is Consistent with Safety Objective
G_ConsistentConstraints
Constraint {X} in lifecycle product {A} is Consistent with Other Constraints
G_Non_Interference
Constraint {X} in lifecycle product {A} has Non-Interference with Other Constraints

Figure 46: Attributes of Validation



Validation (Figure 46)
G_ConsistentConstraints
Constraint {X} in lifecycle product {A} is Consistent with Other Constraints
n constraints
S_ConsistentConstraints
Argument over each constraint with respect to each other constraint.
n-1 other constraints
G_ConsistentConstraints
Constraint {X} in lifecycle product {A} is Consistent with Constraint {Y}

Figure 47: Consistency with Constraints



Validation (Figure 46)
G_Non_Interference
Constraint {X} in lifecycle product {A} has Non-Interference with Other Constraints
n constraints
S_Non_Interference
Argument over each constraint with respect to each other constraint.
n-1 other constraints
G_Consistent_Constraints
Constraint {X} in lifecycle product {A} has Non-Interference with Other Constraint {Y}

Figure 48: Non-interference with Constraints

In (Weaver, 2003)'s argument patterns, non-interference was addressed by including elements of the argument that address failures of other software components which could lead to the specific software failure mode. Effectively (Weaver, 2003) was identifying a category of evidence showing other software components didn't cause the failure in question.

If the concept of the constraint used in this chapter is considered, the mechanisms that provide non-interference between constraints and other aspects of the system can also be labelled as constraints and knowledge of them provided as per the lifecycle product hierarchy and attributes discussed above. For example these non-interference 'constraints' may be defined to address non-interference of intended and unintended coupling paths between components.

Defining 'constraints' for intended coupling paths to show they do not lead to a violation of the initiating constraint will usually involve addressing all intended coupling paths such as control and data flows, intentionally shared resources, etc.

Defining 'constraints' for unintended coupling paths to show they also do not lead to a violation of the initiating constraint will usually involve addressing all feasible spatial and temporal coupling paths. For example 'constraints' can be defined that use containment and/or mediation mechanisms for spatial interference paths. Such 'constraints' for software for example might include such mechanisms as the application of protected modes, virtual machines, memory management units, data wrappers, cache management, and software instruction run time evaluation. 'Constraints' for software based mediation mechanisms of temporal interference paths might include execution time monitors, and real time scheduling mechanisms (earliest deadline first, rate monotonic, cyclic executive with interrupts, etc.).

This sub-section has described the attributes that have been defined for lifecycle products. In Section 5.4, these shall be used as the basis for the definition of an assurance framework for product behaviours associated with constraints. However, it is firstly important to examine several observations about the attribute set.

5.3.6 Asymmetry in Inter-relational Attributes

Although it might seem that compliant/robust/coverage should be symmetrical to both 'higher' and 'lower' abstractions, it would be incorrect to treat them as similar. A lower level abstraction typically includes refined information with implementation specifics. Therefore, while a 'lower' abstraction should always remain compliant and robust with a 'higher' abstraction, a 'higher' abstraction may not be in technical agreement (compliant or robust) with a 'lower' abstraction. When it comes to verification, this same factor governs the relationship between verification evidence of lower abstractions with higher abstractions, and thus prevents verification of higher abstractions with lower abstractions. Hence there is no need for attributes relating to compliance, robustness or coverage in the downwards abstraction direction.

5.3.7 Attributes - Binary versus Qualitative

The attributes defined in this Section can be characterised as either binary or qualitative. Binary attributes are those for which the evidence shows a clearly distinguishable outcome. The following are examples of binary attributes:

- Existence attributes – defined, developed, produced, integrated;
- Traceability attributes

The assessment of all other attributes may not be clearly distinguishable and involves qualitative factors. In these cases it is necessary to reason about the suitability and limitations in evidence where the attribute is not clearly established.

Identification of these differences between attributes is useful as it permits guidelines for reasoning about evidence to be tailored based on attribute type. Section 5.4 examines why this is useful.

5.3.8 Completeness of the Attribute Set

The set of attributes describe above is complete because the set of attributes was determined by ensuring that each potential source of violation for lifecycle product has attributes that provide coverage of requirements validity, requirements satisfaction, and requirements traceability, as shown in Table 18.

Comparison to Software Systems Engineering Initiative Model

There are similarities of the lifecycle product hierarchy and attribute set to elements of the GSN patterns proposed by (Menon, et al., 2009). These patterns provide a framework for reasoning about evidence at differing level of abstractions, and are relevant to many of the underlying principles of proposing a framework based on attributes and the evidence assurance describe in Chapter 6. However this thesis differs from (Menon, et al., 2009) in that it identifies properties of the lifecycle hierarchy and the attribute set that can be exploited for setting qualitative benchmarks for reasoning about evidence. This is an enhancement to the (Menon, et al., 2009) work which only provided a categorisation of argument for reasoning about evidence at each level of abstraction.

Attribute	Requirements Validity	Requirements Satisfaction	Requirements Traceability
Existence			
Defined	✓[A]	✓[A]	✓[A]
Developed	✓[A]	✓[A]	✓[A]
Produced	✓[A]	✓[A]	✓[A]
Integrated	✓[A]	✓[A]	✓[A]
Specification			
Accurate	✓		
Precise	✓	✓[B]	
Unambiguous	✓	✓[B]	
Complete	✓		
Consistent	✓		
Verifiable		✓	
Validatable	✓		
Traceable to Higher Abstractions		✓[C]	✓
Traceable to Lower Abstractions		✓[C]	✓
Compatible with Target		✓	
Verification			
Coverage of Self		✓	
Compliant		✓	
Robust		✓	
Coverage		✓	
Validation			
Consistent with Safety Objectives	✓	✓[D]	
Consistent with Other	✓	✓[D]	
Non-interference with Other	✓	✓[D]	
Inadequacies Resolved	✓	✓	✓
Key: *A: Permits this to be possible.* *B: Addresses potential for errors to be introduced into lower level lifecycle products.* *C: Traceability is necessary in order to relate verification evidence.* *D: Satisfaction is only possible when consistency and non-interference are assured.*			

Table 18: Completeness of Attribute Set

Comparison to RTCA/DO-178B

There are similarities between the some attributes and objectives of RTCA/DO- 178B/C (RTCA Inc., 2011). While some attributes have been labelled consistently with established DO-178B/C terminology, the key differences are as follows:

- in this framework they are referred to as attributes of lifecycle products – which differs in interpretation from DO-178B/C objectives;
- each of the defined attributes is with respect to the specific behaviour under consideration (e.g. the 'constraint') being considered – in DO-178B/C the objectives relate to the entirety of the software product;
- each of the attributes is organised in a set with respect to a lifecycle product (i.e. with respect to product evidence) – in DO-178B/C the objectives are organised

around software lifecycle phases and integral processes, not with respect to the type of evidence;

- each of the attributes is focussed on behaviours of the product with respect to the 'constraint' – DO-178B/C has additional objectives related entirely to process, such as the planning objectives and certification liaison; and
- additional attributes have been developed to address behavioural interferences (or non-interference as should be the goal) between constraints.

In many respects the attribute set described within this chapter is far more general than DO-178B/C usage of objectives. The lifecycle product hierarchy described herein pertains to any product be it software, electronic hardware or mechanical systems, and the attribute sets are general in that they do not imply a specific lifecycle process to generate the evidence. The attributes of the lifecycle products have also been specified so they focus on the general self and relational properties of lifecycle products rather than specific properties of any specific lifecycle product.

5.4 Assurance of Product Behaviours

This section proposes a framework for assurance of product behaviours. The framework provides a set of product behavioural assurance requirements based on a specific instantiation of the meta-claims presented in the previous sections of this chapter. The framework is also intended to address the principles and usability criteria of safety assurance pertaining to knowledge of product behaviours.

In Section 4.7.1 the benefits of assurance levels were discussed with respect to adhering to the usability guidelines presented in Figure 32. Additional guidelines were identified in Section 4.7.2 to avoid the limitations of assurance levels and the framework proposed by this section has been developed in response to these guidelines.

5.4.1 Defining the Claims Safety Assurance Level (CSAL) Concept

This thesis proposes a Claims Safety Assurance Level (CSAL) concept. The CSAL qualifies the level of product behavioural knowledge about validity, satisfaction and traceability of each specified 'constraint' level requirement (refer to Section 5.1.2). In essence, the CSAL sets a benchmark for the tolerability of gaps or discontinuities in the lifecycle product hierarchy (refer Section 5.2) and associated attributes (refer Section 5.3) based on the strength of knowledge required given the severity of failure of the product defence established from Chapter 4. The CSALs are defined based on the meta-argument (Figure 35) that distinguishes options for tolerability of limitations in the chain of evidence across the hierarchy of lifecycle products. Five CSAL levels are proposed in Table 19, although CSAL4 sets an upper limit for reference only and is not used in practice.

CSAL	Category	Intended Outcome – Qualitative	Sources of Knowledge from Lifecycle Product Hierarchy and Attributes[30]
CSAL 4 (not used)	Absolute Assurance	Intended and unintended behaviours of the absence or detection and handling constraint are absolutely assured with respect to safety, such that there is no uncertainty in behaviour	Not practicable (or affordable) to provide evidence of absolute assurance – Near Absolute Assurance provides sufficient control of the uncertainty
CSAL 3	Near Absolute Assurance	All reasonably practical and effective steps have been taken to systematically account for the intended and unintended behaviours of the absence or detection and handling constraint with respect to safety, such that the remaining uncertainty would unlikely lead to a violation of the constraint under any credible or foreseeable circumstances.	Limitations and discontinuities in the lifecycle product hierarchy are intolerable: • Specified Constraint Level Requirement • Refined Abstract Level Design Requirements (as necessary to avoid discontinuities) • Low Level / Detailed Design Requirements • Logical Implementation ("Human Readable") • Logical Implementation ("Machine Readable") • Physical Implementation Limitations in evidence of attributes are intolerable.
CSAL 2	Nominal Assurance	Steps have been taken to systematically account for the intended functional behaviours of the absence or detection and handling constraint with respect to safety, such that the remaining uncertainty would only lead to a violation of the constraint under extremely improbable circumstances	Limitations and discontinuities in the lifecycle product hierarchy are tolerable provided they are constrained such that they don't introduce a discontinuity in the chain of evidence: • Specified Constraint Level Requirement • Refined Abstract Level Design Requirements (as necessary to avoid discontinuities) • Low Level / Detailed Design Requirements • Logical Implementation ("Human Readable") • Logical Implementation ("Machine Readable") • Physical Implementation Limitations in evidence of attributes are tolerable, provided they don't introduce a discontinuity in the chain of evidence.

[30] Note that the expression of this column was refined based on the evaluation (Chapter 10), when compared to earlier work, to better express the relationship to limitations and discontinuities in the lifecycle product hierarchy.

CSAL	Category	Intended Outcome – Qualitative	Sources of Knowledge from Lifecycle Product Hierarchy and Attributes[30]
CSAL 1	Limited Assurance	Claims broadly account for the intended functional behaviours of the absence or detection and handling constraint with respect to safety, such that the remaining uncertainty could lead to a violation of the constraint, but this would not be expected under probably operating conditions that would exercise the constraint	Limitations and discontinuities in the lifecycle product hierarchy are tolerable provided a constraint and implementation exists: • Specified Constraint Level Requirement • Refined Abstract Level Design Requirements (optional) • Low Level / Detailed Design Requirements (optional) • Logical Implementation ("Human Readable") • Logical Implementation ("Machine Readable") • Physical Implementation Limitations in evidence of attributes are tolerable, provided there is evidence of the constraint and implementation.
CSAL 0	No Assurance	No evidence exists to assure the absence or detection and handling constraint with respect to safety	Limitations and discontinuities in the lifecycle product hierarchy are tolerable. Limitations in evidence of attributes are tolerable.

Table 19: Claims Safety Assurance Level (CSAL) Definition

5.4.2 Relating ASALs and CSALs

For each 'constraint' it is necessary to establish the degree to which the 'constraint' should be assured, and also what the degree of assurance means with respect to the product defence. The ASAL concept (refer Chapter 4), uses layers of defences (absence assertions or detection/handling mechanisms) to provide assurance that systematic faults do not lead to unacceptable failure circumstances. Each 'constraint' will be associated with a specific layer of fault prevention (absence) assertion or fault tolerance (detection and handling) mechanisms in the context of the system architecture. Therefore, the degree of claims assurance, as expressed by the CSAL is related to the role of the constraint in the architecture, as expressed by the ASAL. Implicitly therefore, CSAL is also related to the severity of failures associated with the system through the ASAL definition.

Determining the CSAL Assignment Approach

The simplest approach that could be taken is to assign the CSAL commensurate to the severity of the failure of the system and the ASAL, noting that the ASAL is already defined in terms of the severity of failures of the system. Therefore, the stronger the architectural necessity for the system to resolve systematic faults, the stronger the motivation for claims assurance and evidence. This is the approach that has been chosen within this thesis however it is acknowledged that it could be performed differently.

One alternative approach would be to assign the CSAL proportionally to the remaining defence in depth for the given fault propagation path, in addition to the severity of the failure of the system. Thus the greater the defence in depth, the lesser the CSAL could be[31]. In essence, the CSAL could be potentially reduced for one defence in response to an increase in CSAL for another defence. In this scenario, the argument is that claims assurance might also be used to provide additional strength for one layer of mechanism to reduce a higher CSAL for one or more requisite layers. While this seems intuitive, and an attractive approach for a number of practical reasons, there is a factor to this approach which violates a key concept outlined in Chapter 4.

Chapter 4 examined the fail safe design criteria and looked at the effect of single and multiple failures on the system. If the CSALs are reduced based on re-proportioning of CSALs for other defences, then potentially the fail safe design concept is impacted regarding our knowledge of behaviours of defences. Claims assurance is about knowledge of behaviours and not necessarily about the likelihood of failure. The architectural benefits of resolving faults at differing layers of abstraction and the impact of this on bounding uncertainty are an important facet of this framework, which should be supported by the knowledge obtained from claims assurance, and not overridden by it. For this reason, the re-proportioning approaches to CSAL assignment have been avoided in this thesis.

The relationship between ASAL and CSALs is described by Table 20.

Dealing with Additional Layers of Defence

Recognising that some systems might include additional layers of defence (over and beyond the requisite layers defined by Chapter 4); Table 20 also defines the CSAL associated with additional layers. The key factor in specifying the CSAL for additional layers is the extent to which the additional layer might potentially interfere with the required layers, and its effect on the propagation path. Careful consideration is required when assigning layers of defences as either the primary layer, or additional layers. Depending on the layer’s role in the architectural hierarchy of defences, some defences

[31] This is essentially what ARP4754A does through the FDAL and IDAL assignment requirements.

might be more suitable to be defined as primary layers (and subject to non-reduced claims assurance) rather than additional layers due to their potential for interference.

ASAL	1st Layer of Defence Fault Prevention (Absence) or Fault Tolerance (Detection and Handling)		2nd Layer of Defence Fault Tolerance (Detection/Handling)		3rd Layer of Defence Fault Tolerance (Detection/Handling)		Additional Layers of Defence	
							Potentially Interfere[1]	Can't Interfere[2]
ASAL3	Direct	CSAL3	Partitioned Direct[#] or Intra-System[*]	CSAL3	Intra-System[*] or Extra-System Level	CSAL3	CSAL2[$]	CSAL0
ASAL2	Direct	CSAL2	Partitioned Direct[#] or Intra-System or Extra-System	CSAL2	Not Required		CSAL2[$]	CSAL0
ASAL1	Direct OR Intra-System OR Extra-System	CSAL1	Not Required				CSAL1	CSAL0

Notes:

1 Potentially interfere with subsequent detection and handling

2 Can't Interfere with subsequent detection and handling

# must be independent of the initiating failure and the 1st Absence / Detection and Handling mechanism (i.e. through a partitioning mechanism

* must be independent of the preceding detection/handling mechanism

$ additional mechanisms behaviour must be assured to reason that it won't interfere with the main mechanisms

Table 20: Relating ASALs and CSALs

5.4.3 CSAL Assignment Methodology

Section 2.4 identified that there are two factors that must be considered in the assignment of an assurance level, regardless of how it is defined. These are:

- what the software level is being assigned to, and
- how the assignment is performed.

The CSAL is intended to be assigned to a specific behavioural 'constraint' used to provide fault prevention or fault tolerance. In some respects this is equivalent to the assigning of assurance levels to safety functions or safety requirements as performed by existing assurance approaches. Each constraint will have an assigned CSAL, and the CSAL assignment may differ across differing 'constraints' within the same product.

CSAL assignment should be performed by the following steps:

1. Identify the applicable fault prevention (absence assertion) or fault tolerance (detection and handling mechanism), as determined from the architectural assurance activities (refer chapter 4).
2. Determine the ‘constraint’ on the behaviour of the product necessary to achieve fault prevention or fault tolerance.
3. Perform CSAL assignment for the ‘constraint’ using Table 20, as follows:
   a. For the applicable ASAL assigned to the system, identify the relevant row from column 1 of Table 20.
   b. For the applicable ‘constraint’ layer, identify the relevant major column from columns 2, 3, 4 or 5 of Table 20.
   c. Provided the ‘constraint’ meets the criteria from the first sub-column for the relevant major column of Table 20, assign the CSAL per the second sub-column for the relevant major column of Table 20 based on the relevant ASAL row.

With the CSAL assigned, the next step is to establish how the CSAL will be achieved.

5.4.4 Specifying the Attributes of Software Lifecycle Products

Having determined the CSAL, it is necessary to specify the measures for evidence based on the level of knowledge required of lifecycle product chain of evidence and the attributes of these lifecycle products. Appendix B presents the complete list of attributes versus CSAL level for each lifecycle product category based on the requirements of Table 19[32]. Appendix B provides:

- a table for each lifecycle product in the hierarchy.
- each of the attributes as it pertains for the specific lifecycle product,
- details the impact of the attribute not being satisfied with evidence for that lifecycle product, including how this breaks down the evidence chain, and
- the relationship to the CSAL by indicating if it is intolerable, constrained tolerability, or tolerable to have a limitation in evidence for the specified attribute.

[32] The Appendix B tables were refined from earlier works based on the evaluation (Chapter 10) with respect to the grouping and attribute definition to better align with the lifecycle product hierarchy and attribute relationships.

Each attribute is annotated for each lifecycle product as intolerable, constrained or tolerable based on:

- the role of the lifecycle product in the chain of evidence, which sets the reference point for the columnar transition between intolerable (CSAL3), constrained (CSAL2) and tolerable (CSAL1);
- the extent to which the specific attribute can introduce a discontinuity into the chain of evidence, which is shown as adjustments to columnar divisions between intolerable and constrained (CSAL3→CSAL2 or CSAL2→CSAL1), and constrained and tolerable (CSAL2→CSAL1 or CSAL1→CSAL0) based on the lifecycle product; and
- determination as to whether the attribute is binary or qualitative as per Section 5.3.7, which is shown at direct transitions from intolerable to tolerable for binary attributes and intolerable to constrained to tolerable for qualitative attributes.

In essence, this provides an explanation for the differences in assurance level requirements. Appendix B is intended to provide a practical instantiation of the rationale described in earlier sections of this chapter. Having set the tolerability of limitations in evidence for each attribute, Chapter 6 addresses how evidence should be assessed based on the level of knowledge sought for each attribute.

5.4.5 Potential Benefits of the CSAL concept

The CSAL concept provides the following perceived benefits to assurance frameworks:

- The ‘constraint’ and associated CSAL are contextualised by the applicable ASAL objectives, and thus the relevant product failure modes and severity – this provides the CSAL traceability to a product meaning.
- The ASAL/CSAL integrated approach ensures that the search for knowledge of product behaviours is commensurate with the seriousness of each specific failure modes and product behaviour – this is advantageous as it ensures the evidence examined is that evidence that is most relevant to safety and risk.
- The CSAL concept overcomes criticism of existing assurance level approaches that assign a level to an entire configuration irrespective of the differing severities of the failure modes of that configuration item.
- The CSAL approach is an instantiation of a set of general principles pertaining to the chain of evidence in the hierarchy of lifecycle products and their attribute set.

Therefore, while specific instantiations might vary, the general principles have been reasoned about to provide the foundation for the framework.

- The CSAL concept provides inherent consistency in safety cases without unduly limiting or constraining the product.
- The CSAL concept places emphasis on generic properties of a product's evidence without burdening the developer with the difficultly of architecting holistically unique top level arguments for safety cases for each development.
- It should be possible to reuse evidence from developments (e.g. RTCA/DO-178B) with the CSAL concept.

The validity of these benefits is evaluated in Chapter 10 of this thesis.

5.4.6 Potential Limitations of the CSAL concept

The CSAL concept has the following potential limitations:

- There are options for the ways CSALs are assigned (refer Section 5.4.2). The approach chosen in this thesis is a simple relationship, which may not cater for every possible architectural trade-off in practice. Further practical validation is necessary to establish the most suitable relationships between ASAL and CSALs.
- The CSAL concept has set benchmarks for the type of evidence required to support attributes of lifecycle products which are necessary to assure the chain of evidence between constraint and implementation, based on the level of knowledge sought. The CSAL concept does not address how evidence should be reasoned about based on the level of knowledge sought for each attribute. Chapter 6 will examine this in further detail.

The impact of these limitations is evaluated in Chapter 10 of this thesis.

5.5 Defining a Process for Applying the CSAL Concept

With the necessary fault prevention (absence) assertions or fault tolerance (detection and handling) mechanisms identified from architectural assurance activities, it is necessary to define an overall lifecycle process for applying claims assurance to each constraint and associated fault prevention assertion or fault tolerance mechanism. Figure 49 provides an overview of the process, which incorporates those sub-processes defined in Section 5.4.3.

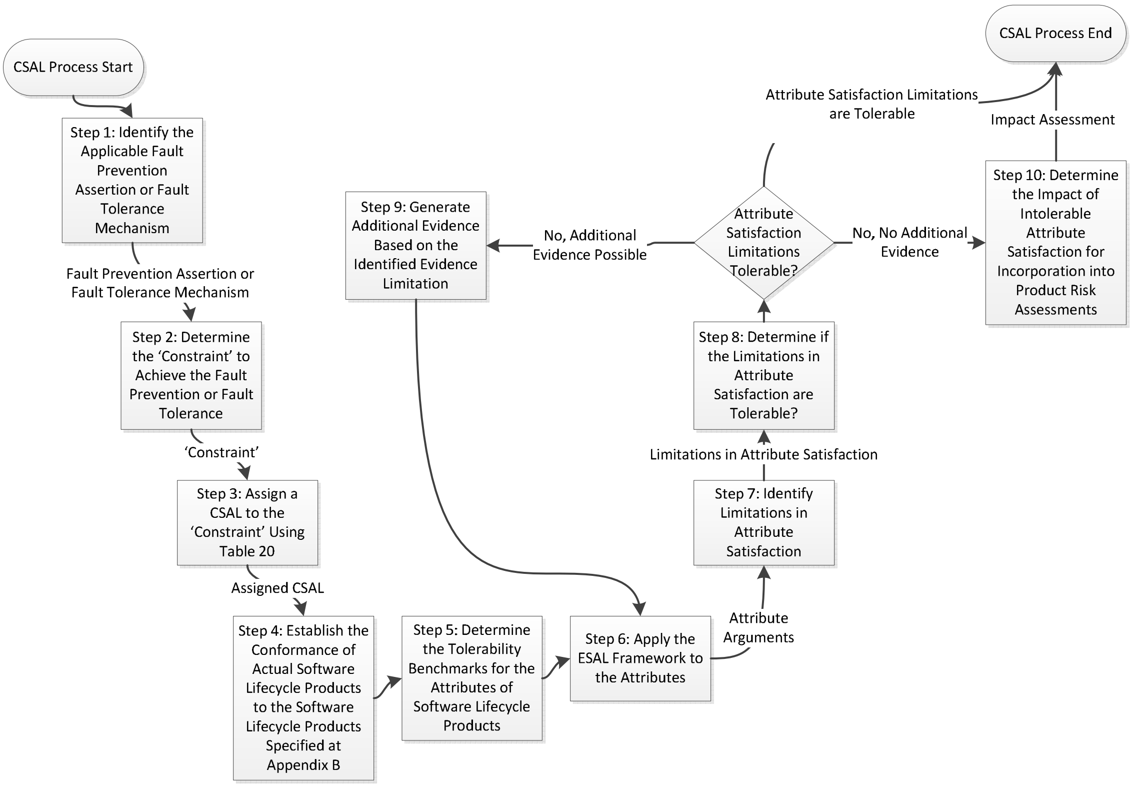

CSAL Process Start
Step 1: Identify the Applicable Fault Prevention Assertion or Fault Tolerance Mechanism
Fault Prevention Assertion or Fault Tolerance Mechanism
Step 2: Determine the 'Constraint' to Achieve the Fault Prevention or Fault Tolerance
'Constraint'
Step 3: Assign a CSAL to the 'Constraint' Using Table 20
Assigned CSAL
Step 4: Establish the Conformance of Actual Software Lifecycle Products to the Software Lifecycle Products Specified at Appendix B
Step 5: Determine the Tolerability Benchmarks for the Attributes of Software Lifecycle Products
Step 6: Apply the ESAL Framework to the Attributes
Attribute Arguments
Step 7: Identify Limitations in Attribute Satisfaction
Limitations in Attribute Satisfaction
Step 8: Determine if the Limitations in Attribute Satisfaction are Tolerable?
Attribute Satisfaction Limitations Tolerable?
No, Additional Evidence Possible
Step 9: Generate Additional Evidence Based on the Identified Evidence Limitation
No, No Additional Evidence
Step 10: Determine the Impact of Intolerable Attribute Satisfaction for Incorporation into Product Risk Assessments
Impact Assessment
Attribute Satisfaction Limitations are Tolerable
CSAL Process End

Figure 49: CSAL Process Overview

The following provides elaboration of each of the CSAL process steps. Each step is illustrated by use of the A-DHC-4 fictitious example from Section 3.8.

5.5.1 Step 1 – Identify the Fault Prevention Assertion or Fault Tolerance Mechanism

a. *Identify the fault prevention assertion or fault prevention mechanism to which claims assurance is to be applied.*

b. *Identify the components within the system on which fault prevention or fault tolerance is to be implemented.*

Example – A-DHC-4

This example considers the fault tolerance mechanism in the SENSOR_DATA_CONDITIONING functional unit of the Primary Command Channel of the Primary Flight Control Computer (refer Figure 31). This mechanism provides a defence against a value failure of sensor_data.attitude#1.

5.5.2 Step 2 – Determine the 'Constraint' to Achieve Fault Prevention or Tolerance

a. *Define a specified 'constraint' level requirement for achieving the fault prevention assertion or fault tolerance mechanisms on the relevant system components. For 'constraints' to be implemented in software, the 'constraint' will be a high level software requirement.*

Example – A-DHC-4

The constraint to achieve the aforementioned defence is *"Value failures of type fixed of sensor_data.attitude#1 shall be detected using reasonability checking against expected attitude data based on sensor_data.attitude#2, aircraft_motion and lateral_mode. Handling shall set the sensor_data.attitude#1.valid flag to invalid."*

This specified constraint level requirement is allocated to the Primary Command Channel software.

5.5.3 Step 3 – Assign a CSAL to the 'Constraint'

a. *Assign a CSAL to the 'constraint' using the process defined in Section 5.4.3 and using Table 20.*

Example – A-DHC-4

Section 4.8.3 detailed that the architecture must be assured to ASAL3. Based on this ASAL assignment, the aforementioned 'constraint' associated with the direct defence should be assured to CSAL3 (in accordance with Table 20).

5.5.4 Step 4 – Establish the Conformance of Software Lifecycle Products to Appendix B

a. *As the lifecycle products generated by the developer may not exactly correspond to the categories of lifecycle products defined by Appendix B, establish a mapping between the categories of Appendix B and the actual lifecycle products being used.*

b. *Where one or more abstractions of Refined Abstract Level Requirements are employed, identify the abstractions to which the Refined Abstract Level Requirements will be applied. Ensure these abstractions are distinct from the Specified 'Constraint' Level Requirements and the Low Level Requirements.*

Example – A-DHC-4

The following summarised mapping is established between the developer lifecycle product documentation and the lifecycle product hierarchy from Appendix B:

- Specified Constraint Level Requirements are documented in the PFCC SSS and SRS, which are stored in the Requirements Management Database. Traceability between the SRS and SSS is documented as links between sub-system level requirements (prefix PFCC) and software requirements (prefix PFCC.PCC).
- One level of Refined Abstract Level Requirements is documented in the Model-Based Development Tool in the form of a functional and implementation model defined by an abstract definition language. Traceability between model elements and software requirements is documented in the Requirements Database.
- Low Level / Detailed Design Requirements are documented in a Software Design Description, which is an export of the model definition from the Model-Based Development Tool supplemented with additional low level requirements pertaining to software architecture and other infrastructure related behaviours such as fault and error management and built in test.
- Logical Implementation ("Human Readable") exists as source code which has been generated from the Model-Based Development Tool. The code has been manually integrated with additional hand coded units necessary to add a subset of SPARK annotations and integrate the source code with the board support and executive/scheduling source code. The Logical Implementation was subject to code reviews (walkthrough and peer reviews), has been checked using the SPARK Examiner, as documented in the Software Test Description.
- Logical Implementation ("Machine Readable") exists as executable object code which has been compiled from the source code using an Ada compiler, including appropriate compile time checking, and integrated with the minimal Ada run time machine. The executable object code was tested on a target based software test bench with simulated and emulated inputs, as documented in the Software Test Description.
- Physical Implementation exists as single board computer hosted on the backplane within the PFCC. The PFCC underwent testing on a target based software test bench as well as system in the systems integration laboratory and 'iron bird' test rig as documented in the Sub-System Test Description and System Test Description.

5.5.5 Step 5 – Determine the Tolerability Benchmarks

a. *Using Appendix B and the assigned CSAL, identify the tolerability benchmark for each attribute for each software lifecycle product.*

Example – A-DHC-4

For CSAL 3 the tolerability benchmark for attributes is Intolerable as per Appendix B. For the sake of brevity of this example, the focus is on the Low Level / Detailed Design Requirements Lifecycle Product only.

5.5.6 Step 6 – Apply the ESAL Framework to the Attributes

a. *For each attribute of each lifecycle product, assign an ESAL based on the tolerability benchmark determined at Step 5.*

b. *Apply the ESAL framework as per Chapter 6.*

Example – A-DHC-4

Consider the following attributes of requirements applicable to the aforementioned constraint in the Low Level / Detailed Design Requirements lifecycle product:

- Specification Group: Accuracy, and
- Verification Group: Robust with Higher Abstraction.

The evidence presented in support of each of these attributes is described in Table 21.

Low Level / Detailed Design Requirements Lifecycle Product	**Attribute**	**Evidence Provided**
	Specification Group: Accuracy	• Software Design Description • Model-Based Development Analysis • Inspections by Peer Review • Configuration Management Records for SDD
	Verification Group: Robust with Higher Abstraction	• Requirements Management Database containing Software Requirements and Traceability Data • Software Design Description • Software Test Description • Software Test Procedures and Cases • Software Test Results • Configuration Management Records

Table 21: Examples of Initial Evidence Supporting Attributes

Chapter 6 provides guidance and continuation of the A-DHC-4 example on providing evidence assurance, using the evidence articles identified above.

5.5.7 Step 7 – Identify Limitations in Attribute Satisfaction

a. *For each argument of tolerability for each attribute of each software lifecycle product, identify limitations in evidence.*

Example – A-DHC-4

Chapter 6 provides guidance and continuation of the A-DHC-4 example on providing evidence assurance. As an example, assume that Chapter 6 identifies the following two untreated limitations in evidence:

- There are limitations in trustworthiness of the evidence supporting the Accuracy attribute because of limitations in independence in the inspection of requirements data and also a lack of conformity review by quality assurance of the processes pertaining to inspections and inspection record management.
- There are limitations in relevance to purpose of the evidence supporting the Robust with Higher Abstraction attribute because of limitations in the comprehensives of robustness test cases. Limitations in executing certain robustness cases on the target hardware means that evidence from host based testing and analysis is required to overcome these limitations.

5.5.8 Step 8 – Determine if the Limitations in Attribute Satisfaction are Tolerable

a. *For each argument of tolerability for each attribute of each software lifecycle product, identify if the limitations in attribute satisfaction are tolerable or intolerable.*

Example – A-DHC-4

The aforementioned limitations are assessed to be intolerable based on the level of knowledge sought and reasoning about evidence assurance described in Chapter 6.

5.5.9 Step 9 – Generate Additional Evidence to Address Limitations

a. *Generate additional evidence to resolve the limitations in attribute satisfaction.*

b. *Revise the arguments established in Step 6 to take into account the additional evidence.*

Example – A-DHC-4

The revised evidence presented in support of each of these attributes is described in Table 21. Additional evidence is shown in italics.

	Attribute	**Evidence Provided**
Low Level / Detailed Design Requirements Lifecycle Product	Specification Group: Accuracy	• Software Design Description • Model-Based Development Analysis • *Inspections by Walkthrough and Peer Review* • *Quality Assurance Records for Inspections* • Configuration Management Records for SDD *and Inspection Records*
	Verification Group: Robust with Higher Abstraction	• Requirements Management Database containing Software Requirements and Traceability Data • *Model-Based Development Tool analysis results* • Software Design Description • Software Test Description • Software Test Procedures and Cases • Software Test Results • Configuration Management Records • *SPARK Analyser Results*

Table 22: Examples of Revised Evidence Supporting Attributes

Chapter 6 provides guidance and continuation of the A-DHC-4 example on providing evidence assurance given the revised evidence. For the purposes of this chapter, assume that the additional evidence resolves the evidence shortfall and that the limitation in evidence with respect to attributes is now tolerable.

5.5.10 Step 10 – Determine the Risk of Intolerable Attribute Satisfaction Limitations

a. *Determine the impact of intolerable attribute satisfaction limitations for communication to higher level product risk assessments, which will include consideration of:*

   i. *the attribute against which the evidence shortfalls exists,*

   ii. *the applicable 'constraint' to which it relates and the corresponding CSAL assignment for that 'constraint', and*

   iii. *the other fault prevention or fault tolerance mechanisms employed by the architecture to treat the source of fault.*

Example – A-DHC-4

Assume that an evidence shortfall has also been identified against the Traceable to Higher Abstraction attribute for the Low Level / Detailed Design Requirements Lifecycle Product. Chapter 8 provides guidance and continuation of the A-DHC-4 example on revising risk assessments based on counter evidence.

5.6 Summary

This chapter has examined how knowledge of behaviours of products can be obtained through examination of a product and its lifecycle products with respect to constraints associated with defences defined in Chapter 4. Specifically the role of lifecycle products has been examined and a categorisation and hierarchy of lifecycle products has been defined. A set of attributes of lifecycle products has also been established to determine evidence requirements for product defences.

A means of defining the knowledge of behaviours of products has been described that examines a product and its lifecycle products, thus providing a means to satisfy Principle C of Figure 32. A set of attributes of lifecycle products was established to determine evidence requirements for product defences, providing a means to satisfy Principle D of Figure 32. Through examining the properties of the categorisation, hierarchy and attribute set of lifecycle products; the rational for how lifecycle products contribute to knowledge of product behaviours have been documented using meta-arguments, thus providing a means to satisfy Principle X of Figure 32. The effect on knowledge of limitations in the evidence supporting attributes and the lifecycle product hierarchy has also been examined and expressed within the meta-arguments, thus providing a means to satisfy Principle Y of Figure 32.

Using the identified categorisation, hierarchy and attributes of lifecycle products, the CSAL framework provides assurance of product behavioural knowledge with respect to constraints for product defences. The assurance framework qualifies the knowledge obtained from evidence based on the tolerability of limitations in knowledge. An example was used to illustrate the process of applying the CSAL framework.

The CSAL framework has been developed to also adhere to the usability guidelines identified in Figure 32. The CSAL framework minimises variability (adhering to Guideline 1) by specifying deterministic requirements for presenting evidence for lifecycle products and attributes thereof. These requirements are derived from general properties of lifecycle products and their hierarchical relationships. The CSAL framework minimises subjectivity (adhering to Guideline 2) with respect to product behavioural assessment by ensuring that the CSAL requirements are explicitly defined based on a mutually exclusive and measurable attribute set. This measurability to minimise subjectivity also permits adherence to Guideline 3 as assessors can distinguish acceptable from non-acceptable. Chapter 6 provides more detail on reasoning about the adequacy of evidence for each attribute.

6 Assurance of Evidence

Evidence is the foundation of numerous safety assurance frameworks, and there is growing consensus that an evidence-based approach to safety assurance should be advocated (Committee on Certifiably Dependable Software Systems, 2007). In generic terms, evidence is information that can be used to substantiate whether a belief or proposition may be true or false (Oxford University Press, 2010). For safety assurance, evidence is the information that should be used to substantiate the claims about knowledge of product behaviours, and the suitability of those behaviours with respect to safety. In Chapter 5, a set of attributes of lifecycle products was established as a means of structuring claims about behaviours of products and the chain of evidence between behaviour and implementation. Chapter 5 also introduced the concept of tolerability of the attribute not being satisfied based on the lifecycle products role in the chain of evidence. This chapter will examine how assurance of evidence for attributes of lifecycle products may be achieved.

In Chapter 3, a model containing principles and usability guidelines was established. This chapter explores the assurance of evidence with respect to attributes. Hence, this chapter focusses on addressing the principles shown in bold italics within Figure 50.

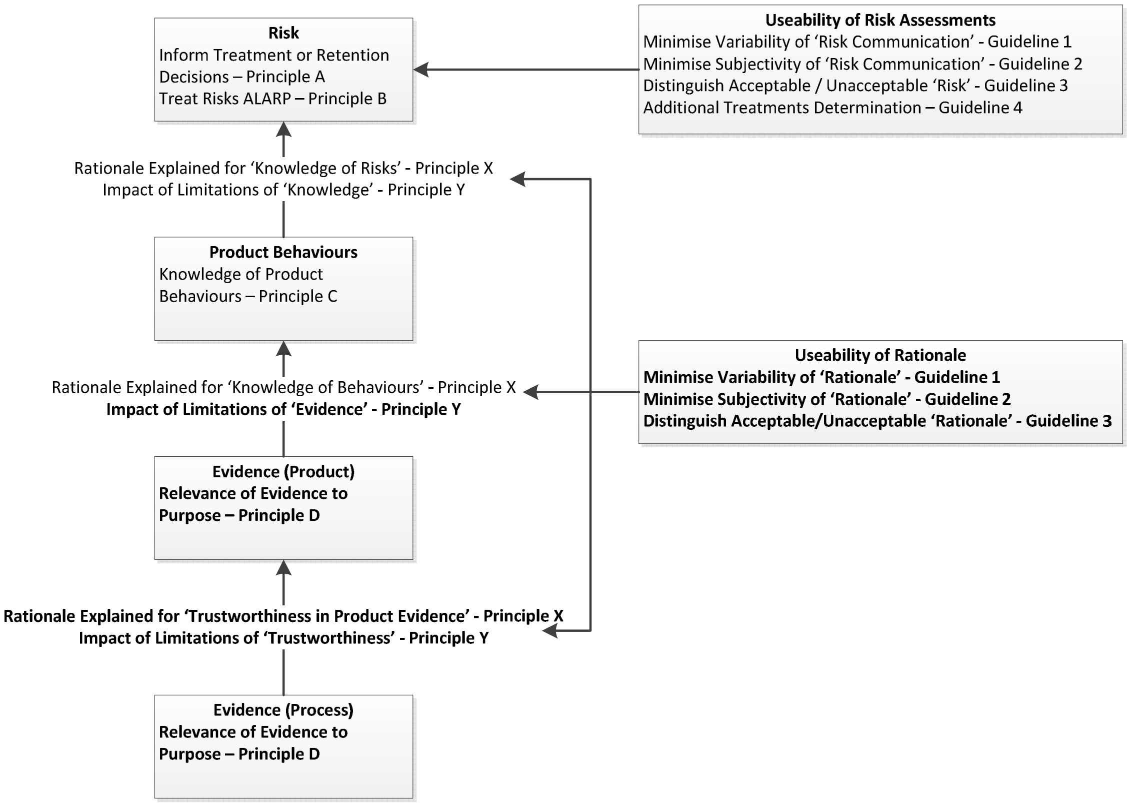

Risk
Inform Treatment or Retention Decisions – Principle A
Treat Risks ALARP – Principle B
Useability of Risk Assessments
Minimise Variability of 'Risk Communication' - Guideline 1
Minimise Subjectivity of 'Risk Communication' - Guideline 2
Distinguish Acceptable / Unacceptable 'Risk' - Guideline 3
Additional Treatments Determination – Guideline 4
Rationale Explained for 'Knowledge of Risks' - Principle X
Impact of Limitations of 'Knowledge' - Principle Y
Product Behaviours
Knowledge of Product Behaviours – Principle C
Useability of Rationale
Minimise Variability of 'Rationale' - Guideline 1
Minimise Subjectivity of 'Rationale' - Guideline 2
Distinguish Acceptable/Unacceptable 'Rationale' - Guideline 3
Rationale Explained for 'Knowledge of Behaviours' - Principle X
Impact of Limitations of 'Evidence' - Principle Y
Evidence (Product)
Relevance of Evidence to Purpose – Principle D
Rationale Explained for 'Trustworthiness in Product Evidence' - Principle X
Impact of Limitations of 'Trustworthiness' - Principle Y
Evidence (Process)
Relevance of Evidence to Purpose – Principle D

Figure 50: Implementing Key Principles/Usability Criteria for Assurance of Evidence

6.1 Exploring the Role of Evidence

Chapter 2 illustrated that evidence plays a prominent role in safety assurance, irrespective of whether the approach involves tasks, objectives, design assurance levels, safety integrity levels, evidence assurance levels or safety arguments. All of these approaches rely on provision of evidence; with a major source of criticism of some frameworks being the type of evidence required and the role of this evidence. For example, design assurance levels and safety integrity levels have been criticised for favouring process evidence over product evidence (McDermid, 2001). However, despite the criticisms, these approaches still involve the generation of evidence, much of which is relevant to making claims relevant to safety objectives. Safety argument methodologies also imply the provision of evidence categorised based on the claims being made, as do evidence assurance levels. The important point that this reveals is to understand the role of different types of evidence. To do this, it is necessary to understand how evidence can be categorised.

6.1.1 Categorisation of Evidence Types

Evidence may be generated from different types of activities, and exists in different forms and formats. Evidence may come from previous products and their development lifecycles, from operation of systems and from the development of systems. However despite its apparently eclectic nature, evidence can be categorised in ways that help in understanding the role of the evidence, and the claims that can be made from it.

(Toulmin, 1958) has suggested that evidence may be considered as Direct or Backing evidence, which is based on the degree of directness by which the evidence supports the arguments being made. (Weaver, 2003) has refined this categorisation based on the evidence's role in supporting requirements validation, requirements satisfaction and requirements traceability. At a more detailed level, (Weaver, 2003) also categorised evidence as pertaining to absence arguments or handling arguments in relation to failure modes. In essence, (Weaver, 2003) categorises evidence by the structure of the argument patterns being defined.

At the lowest level, (Weaver, 2003) categorises for the suitability of evidence based on *Relevance, Trustworthiness and Independence,* which are defined as follows:

Relevance

The extent to which an item of evidence entails[33] *the requirements for evidence.*

(Weaver, 2003)

Trustworthiness

The perceived ability to rely on the character, ability, strength or truth of the evidence.

(Weaver, 2003)

Independence

The extent to which complementary items of evidence follow diverse approaches in fulfilling the requirement for evidence.

(Weaver, 2003)

6.1.2 Relevance

Relevance has two properties (*Directness* and *Coverage*) which support the relationship between the evidence and the requirement for evidence. These are defined as follows:

Directness

The extent to which an item of evidence directly fulfils the requirement for evidence.

(Weaver, 2003)

Coverage

The proportion of the requirement for evidence which the evidence addresses.

(Weaver, 2003)

Coverage suggests that more than one piece of evidence may be required for a given claim, and this it will be necessary to establish how evidence combines or is complementary. Therefore, ways of measuring coverage are required to establish if coverage is achieved, and what any gaps in coverage means in terms of safety impact.

[33] To involve, or logically necessitate.

At an evidence level, *Relevance* can be argued distinctly from *Trustworthiness*. This is because *Relevance* is to do with the strength of the result of the method/s with respect to an associated attribute (and ultimately the claim being made) of the lifecycle product, whereas *Trustworthiness* is the extent to which the results of the evidence are correct.

6.1.3 Trustworthiness

(Weaver, 2003) suggests that *Trustworthiness* may be affected by factors such as:

- "buggy-ness" – how many faults there are in the evidence;
- level of review;
- for tool derived evidence: tool qualification and assurance evidence; and
- experience and competence of the personnel.

In more generic terms, *Trustworthiness* is concerned with understanding the role of:

- competency: a function of qualifications, training and experience;
- scope and level of review/inspection;
- the method or approach used to generate the evidence; and
- the level of independence in generating the evidence or reviewing the evidence.

6.1.4 Independence

(Weaver, 2003)'s final category of suitability of evidence is *Independence*. In defining *Independence*, (Weaver, 2003) introduced the concept that *Mechanistic Independence* or *Conceptual Independence* could be used to improve the assurance of a claim. These concepts are defined as follows:

Mechanistic Independence

Mechanistic Independence is achieved through applying the same underlying principles in different ways. For example, the same testing technique performed by two different testing teams is mechanistically independent.

(Weaver, 2003)

Conceptual Independence

Conceptual Independence is achieved through applying different approaches based on difference underlying principles. For example, (dynamic) testing and static analysis are conceptually different approaches as testing involves executing the program whereas static analysis does not.

(Weaver, 2003)

From their definition, it is evident that *Mechanistic* and *Conceptual Independence* affect *Relevance* and *Trustworthiness*. Therefore, despite being identified as a distinct category of evidence by (Weaver, 2003), *Independence* is actually a property of evidence used to support specific claims about the *Relevance* and *Trustworthiness* of evidence to a claim. This is revealed by examining the potential origins of evidence and the types of claims that can be made based on that context. Hence, *Independence* should be considered a subordinate property of evidence, and not a distinct category as suggested by (Weaver, 2003).

6.1.5 An Alternative Perspective on Evidence Categorisation

In Chapter 3 (Figure 3), the concept of product behaviours being informed by evidence was introduced. In essence, the 'direct' evident is the evidence that informs product behaviours, and thus there is a category of evidence in this context that informs knowledge about product behaviours. In Chapter 3, a set of principles and usability guidelines was also established that distinguishes evidence that directly informs knowledge of product behaviour from other types of evidence that inform knowledge about the product evidence (i.e. the backing evidence). Based on this distinction evidence can also be classified as:

- *product evidence*[34], which can be used to provide knowledge of product behaviours – i.e. the evidence says something directly about the product, e.g. the result of a test case that provides information on the behaviour of the system under the specified test conditions; and
- *process evidence*[35], which can be used to provide knowledge of the trustworthiness or confidence of knowledge of product behaviours – i.e. the evidence says something about the rigour behind another piece of evidence

[34] Product evidence is the safety assurance instantiation of the generic concept of Direct Evidence.

[35] Process evidence is the safety assurance instantiation of the generic concept of Backing Evidence.

(usually product evidence), e.g. evidence of review and inspection of the test procedures and cases used to show that the test procedure and test case was appropriate.

The following sub-sections elaborate product and process evidence.

6.1.6 Product Evidence

Examining product evidence further reveals that product evidence can be categorised based on what knowledge the evidence contributes about product behaviours. Product evidence may exist as:

- *product defining information*, which presents information about the product behaviour (i.e. what is the product behaviour?), examples of which are:
  - specification information, such as requirements and detailed design;
  - implementation information, such as source code and other implementation language code;
  - verification of specification information, such as analysis outcomes or test cases and results; and
  - verification of implementation information, such as analysis outcomes or test cases and results;
- *rationale for product behaviour*, which provides information about the rationale for the product behaviour (i.e. why does the product have this behaviour), examples of which are:
  - analysis / modelling / simulation of specification information;
  - analysis / modelling / simulation of implementation information;
  - analysis of verification information;
  - validation (analytical or empirical) of specification information; and
  - validation (analytical or empirical) of implementation information

For example, worst case execution time analysis / testing are *product defining information* because they provide information as to 'what' the timing behaviour of software is. Whereas the analysis / modelling / simulation that establishes the bounds on what are acceptable worst case execution times because of hard real time deadlines and temporal partition is *rationale for product behaviour*. This example emphasises two points: what the evidence is trying to show and what the method used to produce the evidence is.

6.1.7 Process Evidence

Process evidence also lends itself to further categorisation based on what the evidence contributes to knowledge about trustworthiness or confidence. Process evidence may be categorised as shown in Table 23 and elaborated below:

	Product Evidence Generation	***Review/Inspection of Product Evidence***
Competency	Domain, Method	Domain, Method
Method	Suitability, Rigour	Suitability, Rigour
Independence	Complementary	Review/Inspection, Complementary

Table 23: Process Evidence Categories

- *competency*:
  - *for product evidence generation*:
    - *competency in product domain* provides information on the domain competency of the staff involved in the production of the evidence (e.g. for flight control system development the evidence that staff understand the flight control systems domain, as measured by recognised competency frameworks based on qualifications, training and experience);
    - *competency with method* provides information on the development/verification method competency of the staff involved in the production of the product evidence using that method (e.g. the evidence that the staff understand the method used, as measured by evidence of training and experience using that method);
  - for *review/inspection of product evidence*:
    - *competency in product domain*, as for product generation evidence;
    - *competency with method*, as for product generation evidence but with respect to review/inspection method;
- *method:*
  - for *product evidence generation*:
    - *suitability of method* provides information on the suitability of the method of evidence generation with respect to the role of the product evidence (e.g. a UML class diagram is not suitable for providing information on behaviours pertaining to data flow, whereas a MASCOT model is suitable for providing information on data flow between functional units);

        - *coverage of application of method,* which provides information on the coverage of the method for product evidence generation (e.g. a MASCOT model which only models data flows associated with the call tree between selected function units is not as rigorous as a MASCOT model which models all data flows, including those related to call tree, shared resources, and external dependencies);
    - for *review/inspection of product evidence*:
        - *suitability of method*, which provides information on the suitability of the approach used for review/inspection of product evidence (e.g. a walkthrough style inspection may be suitable for assessing the accuracy of textual requirements but less suitable for assessing the accuracy of executable object code, and proofs against hypothesis is more suitable for checking the accuracy of a formal model, than for checking the correctness of test procedures and test cases);
        - *rigour of application of method,* which provides information on the degree to which the review/inspection was systematic or adhoc (e.g. a formal inspection, is more rigorous than a walkthrough, than a guided desktop review, than unguided peer review);
- *independence*:
    - for *product evidence generation*:
        - *complementary product evidence generation*, which provides information about whether mechanistic or conceptual independence in product generation which may have been used to improve the knowledge about trustworthiness (e.g. the usage of two independent teams, using similar (mechanistic) or diverse (conceptual) approaches to product evidence generation);
    - for *review/inspection of product evidence*:
        - *independence of review/inspection activity*, which provides information on the degree to which the review/inspection might have been undermined by project bias or lack of independent mind-set (e.g. review from another member of the development team, versus review from another team such as a test team or quality assurance team, versus review from another organisation);
        - *complementary review/inspection evidence generation*, which provides information about whether mechanistic or conceptual

independence in review process may have been used to increase the knowledge about trustworthiness (e.g. multiple layers of review by reviewers from different roles or backgrounds).

It is important that evidence is used for the right purposes, and that the level of knowledge provided by the evidence can be assessed. Depending on the category of evidence, the existence of evidence affects the knowledge differently. Therefore, it is important to establish how the sufficiency of evidence can be evaluated.

6.1.8 Establishing What the Evidence Does or Doesn't Confirm

In Chapter 3, it was identified that the body of evidence will never be infinite or absolute. In practice, there is never enough time or resources, and all systems are fielded with limitations in evidence. Hence, it is important to focus on:

- what is known? – i.e. what the available evidence can confirm about knowledge of product behaviours, or the trustworthiness of the knowledge therein?,
- what is unknown? – i.e. what the available evidence can't confirm about knowledge of product behaviours or the trustworthiness therein?,
- what could be known? – i.e. what uncertainty results from evidence which does not exist (but could be given more time and resources)?, and
- what should be known? – i.e. what additional evidence (should it be produced) would resolve such uncertainty?.

Across each of these points, a major factor is the limitations in evidence, and how it contributes to uncertainty. If the uncertainty fundamentally undermines the knowledge of product behaviours, then the limitation in evidence may be intolerable. Safety assurance will only be achieved when the limitations in evidence are tolerable.

6.1.9 Understanding the Origins of Limitations in Evidence

There can be limitations in evidence because of the following:

- insufficient methods were applied (so a particular type of evidence is missing) – e.g. claims of accuracy of the requirement is made based on review/inspection evidence only, and there has been no comparison to the results of previous designs, experiments or modelling analysis;
- inappropriate methods or techniques were applied (so the wrong type of evidence is being proposed for assuring a specific attribute) – e.g. formal methods proofs evidence can't be used to make claims about the validity of inputs and outputs of

a system with respect to the operational environment, but formal methods proofs may be used to show that the precision of values are consistent throughout the design model;

- the method or technique was applied incorrectly or non-rigorously (so the evidence may have errors in it) – e.g. the evidence was produced by a developer that has no qualification, training or experience, has not been subject to any level of review by competent supervisors, or there has been no conformity review to established that the developer applied the method properly; and
- the results of application of the method or technique are contrary to the objectives of the method or technique – e.g. a test case shows that the low level design requirement doesn't fully implement the higher level requirement, and that the behaviour in these cases may be undesirable.

Results of Evidence

The above bullet points also suggest an additional category of sufficiency of evidence not covered by (Weaver, 2003)'s definitions – the results[36] of the evidence. For example, even if the evidence type is relevant to the attribute (or claim type) and trustworthy, it is still important that the evidence indicates success or failure against the objective for the evidence. Results that indicate failures, inappropriate behaviours or anything contrary to what is being claimed are all counter evidence to the claim. Therefore, in addition to relevance and trustworthiness, the results of evidence should be considered a distinct category of sufficiency of evidence. The results of the evidence are important because they:

- provide positive indication of the behaviour of the software being appropriate with respect to the constraint and the safety of the system,
- reveal direct counter evidence of a behaviour of the software that would violate the constraint with respect to safety; or
- disclose uncertainty based on counter evidence which may raise questions with respect to the relevance and trustworthiness arguments.

Acknowledging the results of evidence also avoids a common misconception that an absence of evidence infers evidence of absence of faults in a system. Two types of counter evidence have been identified above – direct counter evidence or uncertainty

[36] 'Results' encompasses the outcome, meaning, interpretation and/or consequences of the evidence.

based counter evidence. For systems with severe failure modes, uncertainty based counter evidence is equally important as direct counter evidence, as either are not positive confirmation of appropriate behaviours.

No Single Method

Rarely will a single method address each attribute defined in Chapter 5. All methods have limitations that will impact the relevance of the evidence, and depending on how well they are applied, there may be limitations in the trustworthiness of the evidence. These limitations exist because almost all methods are defined based on a model of the problem they are intended to solve, and almost invariably, this model has limitations.

One example of this is the application of formal methods to proving behaviours about software. Formal methods are good at showing the correctness and internal consistency of a formally defined model. However, to make the model manageable, associated behaviours (e.g. target computer behaviours, operational environment) are almost always simplified, or even excluded. For this reason, formal models may be used to complement testing on the target computer. Likewise there are limitations to testing. It is impractical to exhaustively test all combinations of input and output data, or states for problems that suffer state explosion, and thus complementary approaches (such as formal methods and static code analysis) are usually necessary to overcome the limitations of testing. Many such examples can be provided for a large range of methods, and each development needs to reason about how the totality of methods overcomes the limitations of each method.

As all methods have limitations, it is necessary to ensure an evidence framework that requires arguments about limitations of evidence is explicit in identifying, evaluating and resolving limitations where necessary.

6.1.10 Relating Evidence Assurance to Knowledge of Behaviours

In Chapter 5, attributes of lifecycle products were defined in order to guide the consolidation of evidence with respect to constraints on the behaviour of the system. This chapter now examines how evidence can be allocated and measured against each of the attributes defined in Chapter 5, and a measure of knowledge established from the evidence provided. Figure 51 provides the linkage between Chapter 5 and the knowledge of attributes sought from evidence which is described by this chapter.

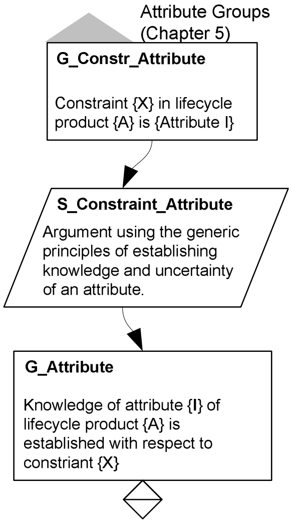

Attribute Groups (Chapter 5)
G_Constr_Attribute
Constraint {X} in lifecycle product {A} is {Attribute I}
S_Constraint_Attribute
Argument using the generic principles of establishing knowledge and uncertainty of an attribute.
G_Attribute
Knowledge of attribute {I} of lifecycle product {A} is established with respect to constriant {X}

Figure 51: Knowledge of Attributes

6.2 The Tolerability of Limitations Concept

Section 6.1 has identified evidence and limitations in evidence as the key sources of knowledge and uncertainty respectively. Where the goal for knowledge is high, then limitations in evidence may be intolerable, whereas if the goal for knowledge is lesser, because the attribute can't undermine the chain of evidence, then the tolerability of limitations maybe greater. This relationship is represented in Figure 52.

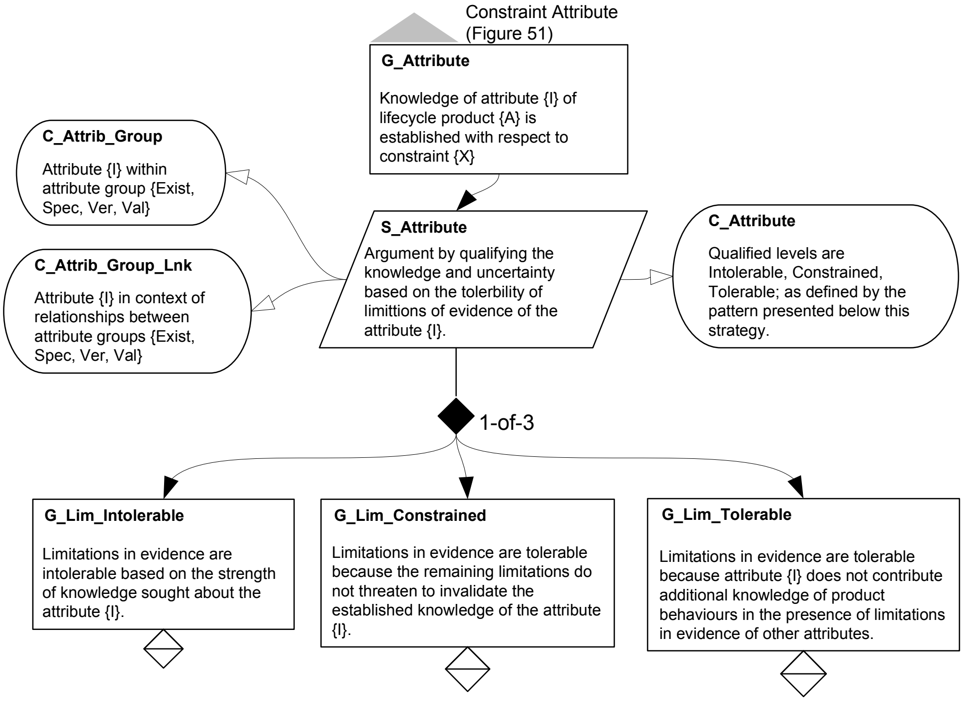

Constraint Attribute (Figure 51)
G_Attribute
Knowledge of attribute {I} of lifecycle product {A} is established with respect to constraint {X}
C_Attrib_Group
Attribute {I} within attribute group {Exist, Spec, Ver, Val}
C_Attrib_Group_Lnk
Attribute {I} in context of relationships between attribute groups {Exist, Spec, Ver, Val}
S_Attribute
Argument by qualifying the knowledge and uncertainty based on the tolerbility of limittions of evidence of the attribute {I}.
C_Attribute
Qualified levels are Intolerable, Constrained, Tolerable; as defined by the pattern presented below this strategy.
1-of-3
G_Lim_Intolerable
Limitations in evidence are intolerable based on the strength of knowledge sought about the attribute {I}.
G_Lim_Constrained
Limitations in evidence are tolerable because the remaining limitations do not threaten to invalidate the established knowledge of the attribute {I}.
G_Lim_Tolerable
Limitations in evidence are tolerable because attribute {I} does not contribute additional knowledge of product behaviours in the presence of limitations in evidence of other attributes.

Figure 52: Tolerability of Limitations in Evidence

In Section 5.2 the relationship between the chain of evidence of lifecycle products was examined and used to frame three qualitative levels of tolerability in the chain of evidence: Intolerable, Constrained and Tolerable, as follows:

- *Intolerable*: limitations in evidence for assuring the attribute are intolerable based on the strength of claim about knowledge of product behaviours.
- *Constrained*: limitations in evidence for assuring the attribute are tolerable provided those limitations don't undermine the satisfaction of the attribute and the role of the lifecycle product in the chain of evidence.
- *Tolerable*: limitations in evidence for assuring the attribute are tolerable because the attribute does not contribute knowledge of product behaviours in the presence of limitations in evidence for other attributes, and because of the role of the lifecycle product in the chain of evidence.

Because of the judgmental nature of attributes, the developer should provide arguments or rationale about the 'tolerability of limitations' with respect to the specific attribute.

6.2.1 Categories for Tolerability of Limitations

If the developer is going to express arguments about the 'tolerability of limitations' in evidence, what should these arguments be about? Section 6.1 has suggested that evidence may be categorised based on the requirement for evidence (relevance, trustworthiness, results). Evidence is also categorised based on the origin of the evidence (product or process evidence). While arguments about tolerability of limitations should be made with respect to what is claimed (relevance, trustworthiness, results), the sources of limitations are most recognisable with respect to the origin of evidence (product or process evidence). In essence, evidence assurance needs to articulate the effect of limitations in product and process evidence categories of relevance, trustworthiness and results (refer Figure 53).

Figure 53 provides a point in the top down argument developed from Chapters 4 and 5, where arguments become specific to solutions. Therefore, Figure 53 provides a junction (S_Attribute_Know) between the top-down argument, and the bottom up arguments about evidence.

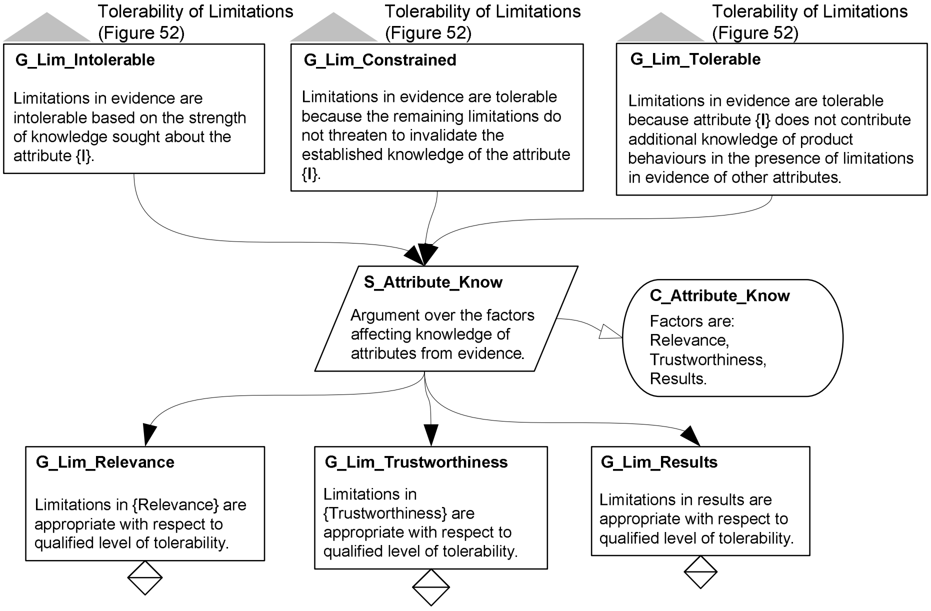

Tolerability of Limitations (Figure 52)
G_Lim_Intolerable
Limitations in evidence are intolerable based on the strength of knowledge sought about the attribute {I}.
Tolerability of Limitations (Figure 52)
G_Lim_Constrained
Limitations in evidence are tolerable because the remaining limitations do not threaten to invalidate the established knowledge of the attribute {I}.
Tolerability of Limitations (Figure 52)
G_Lim_Tolerable
Limitations in evidence are tolerable because attribute {I} does not contribute additional knowledge of product behaviours in the presence of limitations in evidence of other attributes.
S_Attribute_Know
Argument over the factors affecting knowledge of attributes from evidence.
C_Attribute_Know
Factors are: Relevance, Trustworthiness, Results.
G_Lim_Relevance
Limitations in {Relevance} are appropriate with respect to qualified level of tolerability.
G_Lim_Trustworthiness
Limitations in {Trustworthiness} are appropriate with respect to qualified level of tolerability.
G_Lim_Results
Limitations in results are appropriate with respect to qualified level of tolerability.

Figure 53: Relevance, Trustworthiness and Results of Evidence

The undeveloped goals are developed for relevance, trustworthiness and results in Section 6.3, 6.4, and 6.5 respectively.

6.3 Relevance of Evidence

Arguments based on the tolerability of limitations (i.e. intolerable, constrained, or tolerable) should be presented for the relevance of evidence, as shown in Figure 54. These arguments are with respect to the specific attribute of the lifecycle product with respect to the constraint.

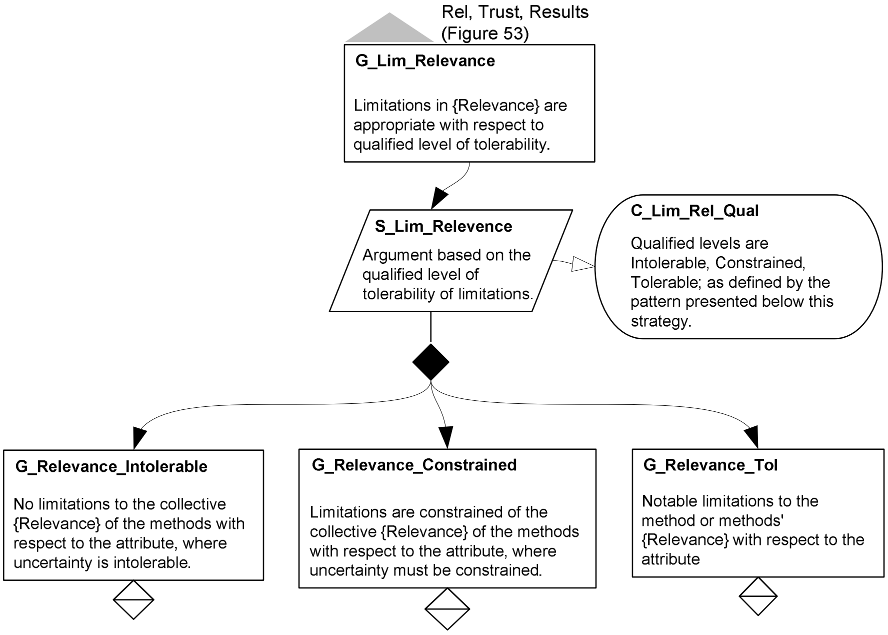

Rel, Trust, Results (Figure 53)
G_Lim_Relevance
Limitations in {Relevance} are appropriate with respect to qualified level of tolerability.
S_Lim_Relevence
Argument based on the qualified level of tolerability of limitations.
C_Lim_Rel_Qual
Qualified levels are Intolerable, Constrained, Tolerable; as defined by the pattern presented below this strategy.
G_Relevance_Intolerable
No limitations to the collective {Relevance} of the methods with respect to the attribute, where uncertainty is intolerable.
G_Relevance_Constrained
Limitations are constrained of the collective {Relevance} of the methods with respect to the attribute, where uncertainty must be constrained.
G_Relevance_Tol
Notable limitations to the method or methods' {Relevance} with respect to the attribute

Figure 54: Relevance of Evidence

6.3.1 Relevance – Intolerable Limitations

For cases where limitations in the relevance of evidence are intolerable, it is necessary to make an argument that there are no limitations to the collective relevance of the methods used for product evidence generation with respect to the attribute of the lifecycle product. This is achieved by arguing over the systematic identification and treatment of all limitations of relevance of evidence. This argument is expressed in Figure 55.

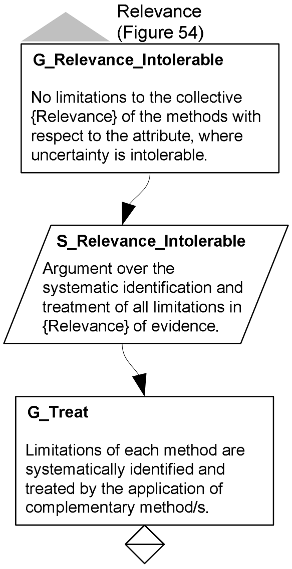

Relevance
(Figure 54)
G_Relevance_Intolerable
No limitations to the collective {Relevance} of the methods with respect to the attribute, where uncertainty is intolerable.
S_Relevance_Intolerable
Argument over the systematic identification and treatment of all limitations in {Relevance} of evidence.
G_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s.

Figure 55: Relevance of Evidence – Intolerable Limitations

Section 6.3.4 develops how the limitations of each method are systematically identified and treated by the application of complementary methods.

6.3.2 Relevance – Constrained Limitations

For cases where limitations in the relevance of evidence are constrained, it is necessary to adapt the argument used for intolerable to only require treatment of limitations where they are practical to treat. The means of establishing reasonable practicality of treatment is discussed in Section 6.3.4, albeit the principles of the legal tests of reasonability are intended to apply.

Therefore, the argument is that limitations of the collective relevance of the methods used for product evidence generation are constrained with respect to the attribute of the lifecycle product. This is achieved by arguing over the systematic identification and treatment, where practical, of all limitations of relevance of evidence. This argument is expressed in Figure 56.

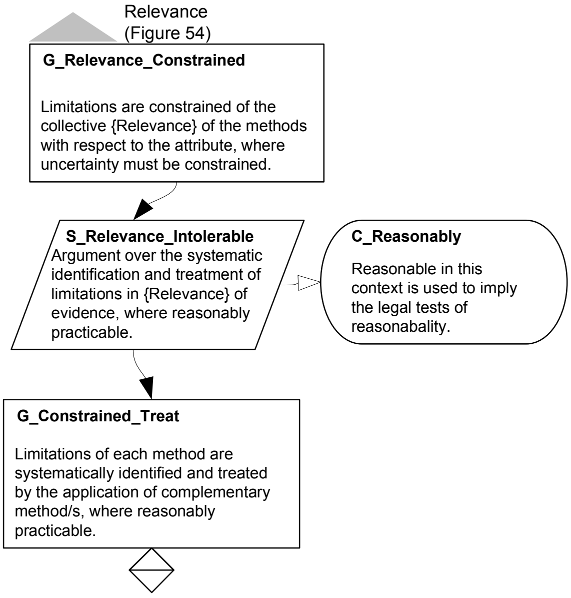

Relevance
(Figure 54)
G_Relevance_Constrained
Limitations are constrained of the collective {Relevance} of the methods with respect to the attribute, where uncertainty must be constrained.
S_Relevance_Intolerable
Argument over the systematic identification and treatment of limitations in {Relevance} of evidence, where reasonably practicable.
C_Reasonably
Reasonable in this context is used to imply the legal tests of reasonabality.
G_Constrained_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s, where reasonably practicable.

Figure 56: Relevance of Evidence – Constrained Limitations

6.3.3 Relevance – Tolerable Limitations

For cases where limitations in the relevance of evidence are tolerable, it is necessary to adapt the argument used for constrained to indicate that treatment of limitations may not have to be undertaken. Therefore, although there are notable limitations of the relevance of evidence with respect to the attribute of the lifecycle, this can be tolerated. This is achieved by reasoning about limitations that may not be identified or treated. This argument is expressed in Figure 57.

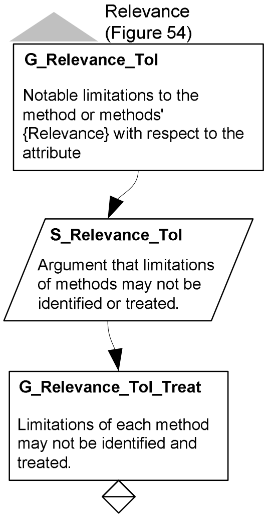

Relevance
(Figure 54)
G_Relevance_Tol
Notable limitations to the method or methods' {Relevance} with respect to the attribute
S_Relevance_Tol
Argument that limitations of methods may not be identified or treated.
G_Relevance_Tol_Treat
Limitations of each method may not be identified and treated.

Figure 57: Relevance of Evidence – Tolerable Limitations

6.3.4 Treating Limitations in Evidence

To argue that the limitations of each method are systematically identified and treated by the application of complementary methods it is necessary to argue in specific terms about the limitations of each method, as shown in Figure 58.

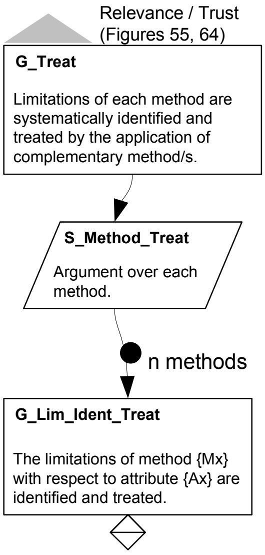

Relevance / Trust
(Figures 55, 64)
G_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s.
S_Method_Treat
Argument over each method.
n methods
G_Lim_Ident_Treat
The limitations of method {Mx} with respect to attribute {Ax} are identified and treated.

Figure 58: Treatment of Limitations

For each method, limitations are identified against the claim being made from the evidence. For each limitation it is then necessary to identify one or more methods which will treat the identified limitations and complement the evidence already provided, as shown in Figure 59.

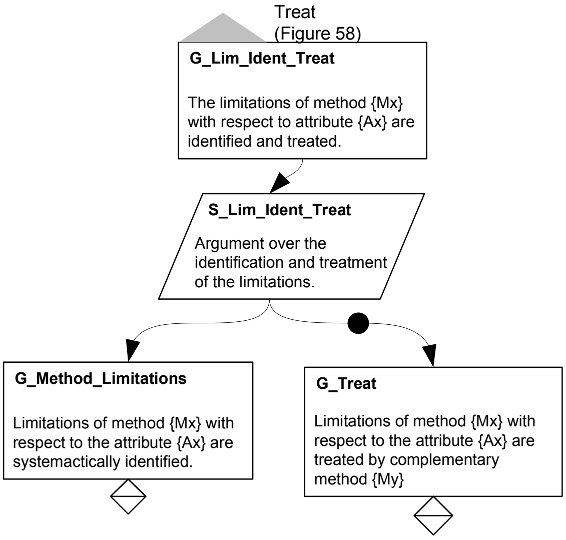
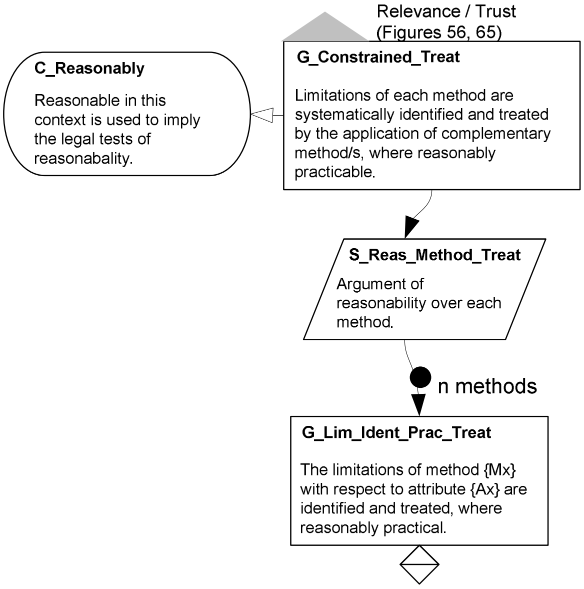

Treat
(Figure 58)
G_Lim_Ident_Treat
The limitations of method {Mx} with respect to attribute {Ax} are identified and treated.
S_Lim_Ident_Treat
Argument over the identification and treatment of the limitations.
G_Method_Limitations
Limitations of method {Mx} with respect to the attribute {Ax} are systemactically identified.
G_Treat
Limitations of method {Mx} with respect to the attribute {Ax} are treated by complementary method {My}

Figure 59: Identification and Treatment of Limitations

6.3.5 Treating Limitations Where Practical

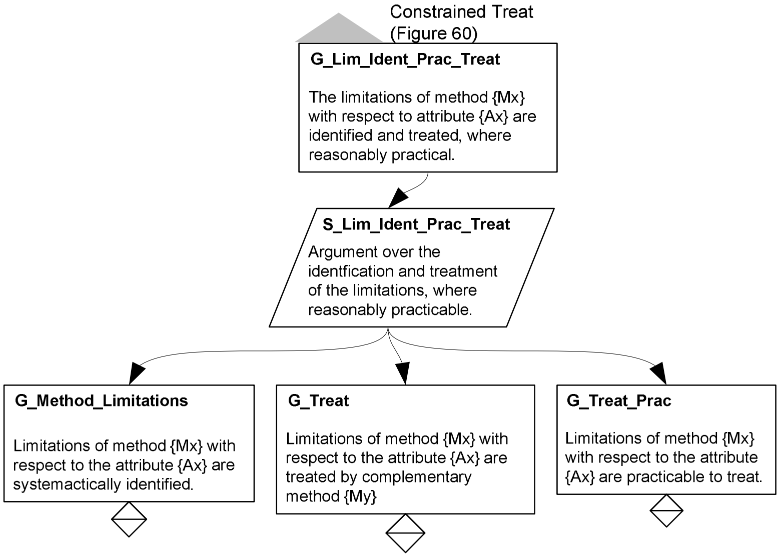
There are cases where the argument that the limitation of each method are systematically identified and treated by the application of complementary methods, should only be made where it is reasonable practical to treat the limitation. In these cases it is necessary to argue in specific terms about the limitations of each method (or application of method), as well as the reasonability, as shown in Figure 60.



Relevance / Trust
(Figures 56, 65)
C_Reasonably
Reasonable in this context is used to imply the legal tests of reasonabality.
G_Constrained_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s, where reasonably practicable.
S_Reas_Method_Treat
Argument of reasonability over each method.
n methods
G_Lim_Ident_Prac_Treat
The limitations of method {Mx} with respect to attribute {Ax} are identified and treated, where reasonably practical.

Figure 60: Constrained Treatment of Limitations

For each method and attribute claim, limitations are identified. For each limitation it is then necessary to identify one or more methods which will treat the identified limitation and complement the evidence already provided. Further, a justification should be made about the practicality of treating limitations, as shown in Figure 61.



Constrained Treat
(Figure 60)
G_Lim_Ident_Prac_Treat
The limitations of method {Mx} with respect to attribute {Ax} are identified and treated, where reasonably practical.
S_Lim_Ident_Prac_Treat
Argument over the identfication and treatment of the limitations, where reasonably practicable.
G_Method_Limitations
Limitations of method {Mx} with respect to the attribute {Ax} are systemactically identified.
G_Treat
Limitations of method {Mx} with respect to the attribute {Ax} are treated by complementary method {My}
G_Treat_Prac
Limitations of method {Mx} with respect to the attribute {Ax} are practicable to treat.

Figure 61: Constrained Identification and Treatment of Limitations

Reasonable practicality of treatment involves an argument over the factors affecting reasonable practicality which are cost and safety benefit. An argument is required that the cost of treating the limitation is not disproportionate to the benefit of resolving the limitation, and the benefit must provide reasonable improvement of the knowledge of the attribute, as shown in Figure 62. Chapter 8 provides further elaboration for how this information impact safety.

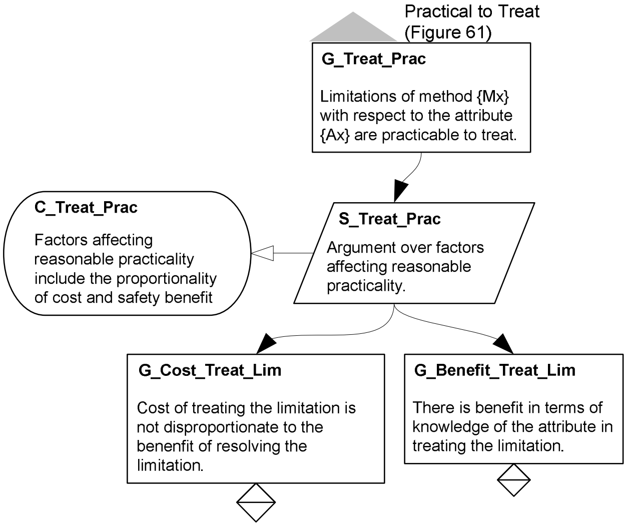

Practical to Treat
(Figure 61)
G_Treat_Prac
Limitations of method {Mx} with respect to the attribute {Ax} are practicable to treat.
C_Treat_Prac
Factors affecting reasonable practicality include the proportionality of cost and safety benefit
S_Treat_Prac
Argument over factors affecting reasonable practicality.
G_Cost_Treat_Lim
Cost of treating the limitation is not disproportionate to the benenfit of resolving the limitation.
G_Benefit_Treat_Lim
There is benefit in terms of knowledge of the attribute in treating the limitation.

Figure 62: Where Practicable Treatment of Limitations

6.4 Trustworthiness of Evidence

Along the same lines as relevance, arguments based on the tolerability (i.e. intolerable, constrained, or tolerable) of limitations should be presented for the trustworthiness of evidence, as shown in Figure 63. As these argument patterns mirror the argument structures for relevance from Section 6.3, the explanation has not been repeated in this section. For completeness the GSN argument patterns have been included.

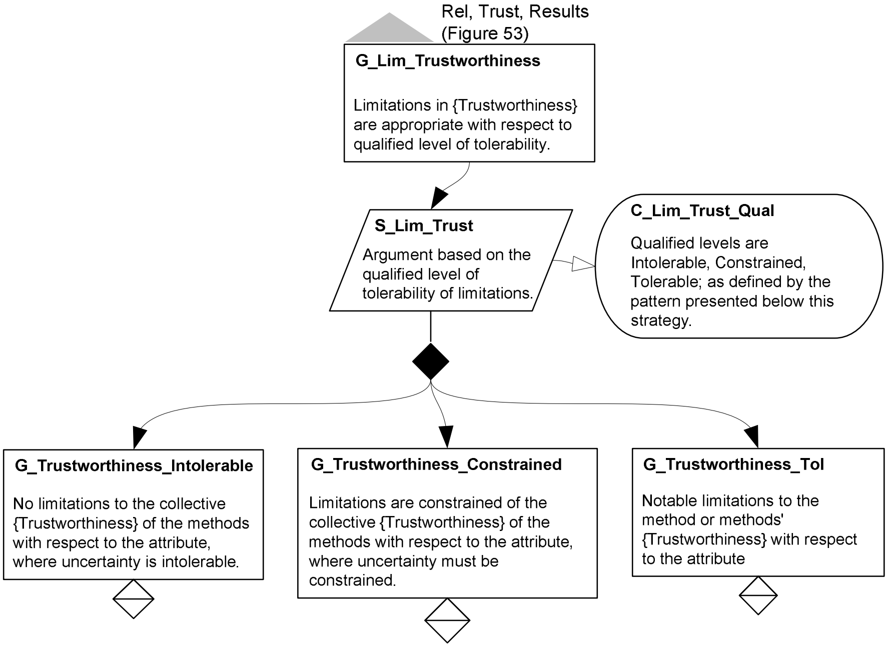

Rel, Trust, Results
(Figure 53)
G_Lim_Trustworthiness
Limitations in {Trustworthiness} are appropriate with respect to qualified level of tolerability.
S_Lim_Trust
Argument based on the qualified level of tolerability of limitations.
C_Lim_Trust_Qual
Qualified levels are Intolerable, Constrained, Tolerable; as defined by the pattern presented below this strategy.
G_Trustworthiness_Intolerable
No limitations to the collective {Trustworthiness} of the methods with respect to the attribute, where uncertainty is intolerable.
G_Trustworthiness_Constrained
Limitations are constrained of the collective {Trustworthiness} of the methods with respect to the attribute, where uncertainty must be constrained.
G_Trustworthiness_Tol
Notable limitations to the method or methods' {Trustworthiness} with respect to the attribute

Figure 63: Trustworthiness of Evidence

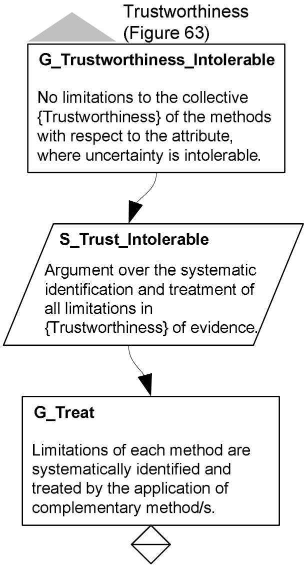

Trustworthiness
(Figure 63)
G_Trustworthiness_Intolerable
No limitations to the collective {Trustworthiness} of the methods with respect to the attribute, where uncertainty is intolerable.
S_Trust_Intolerable
Argument over the systematic identification and treatment of all limitations in {Trustworthiness} of evidence.
G_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s.

Figure 64: Trustworthiness– Intolerable Limitations

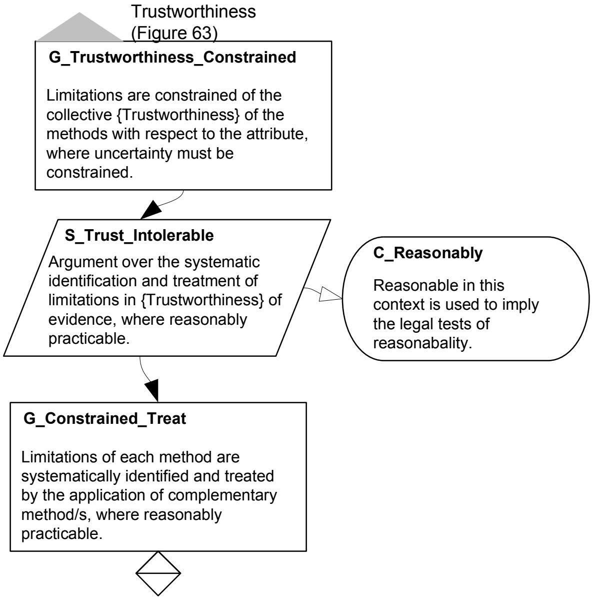

Trustworthiness
(Figure 63)
G_Trustworthiness_Constrained
Limitations are constrained of the collective {Trustworthiness} of the methods with respect to the attribute, where uncertainty must be constrained.
S_Trust_Intolerable
Argument over the systematic identification and treatment of limitations in {Trustworthiness} of evidence, where reasonably practicable.
C_Reasonably
Reasonable in this context is used to imply the legal tests of reasonabality.
G_Constrained_Treat
Limitations of each method are systematically identified and treated by the application of complementary method/s, where reasonably practicable.

Figure 65: Trustworthiness– Constrained Limitations

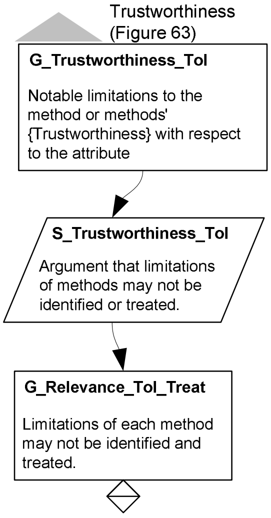

Trustworthiness
(Figure 63)
G_Trustworthiness_Tol
Notable limitations to the method or methods' {Trustworthiness} with respect to the attribute
S_Trustworthiness_Tol
Argument that limitations of methods may not be identified or treated.
G_Relevance_Tol_Treat
Limitations of each method may not be identified and treated.

Figure 66: Trustworthiness of Evidence – Tolerable Limitations

6.5 Results of Evidence

Arguments based on the tolerability (i.e. intolerable, constrained, or tolerable) of limitations should be presented for the results of evidence, as shown in Figure 67. These arguments are with respect to the specific attribute of the lifecycle product with respect to the constraint.

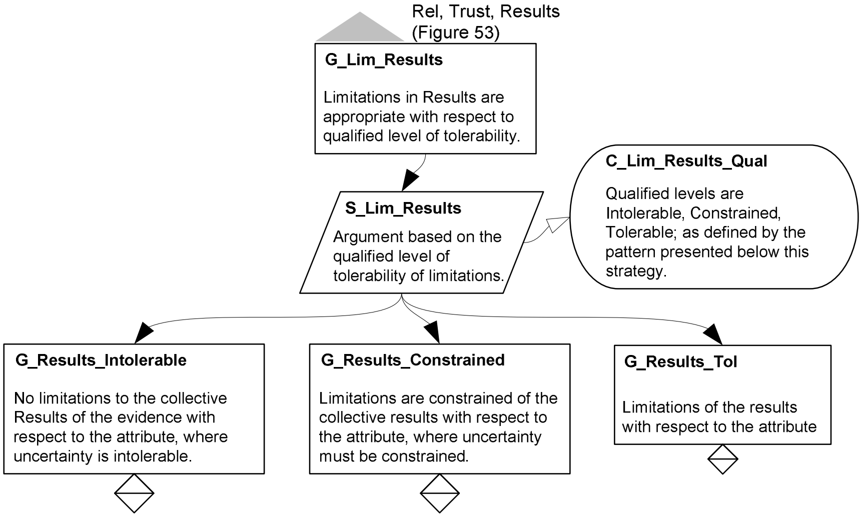

Rel, Trust, Results
(Figure 53)
G_Lim_Results
Limitations in Results are appropriate with respect to qualified level of tolerability.
S_Lim_Results
Argument based on the qualified level of tolerability of limitations.
C_Lim_Results_Qual
Qualified levels are Intolerable, Constrained, Tolerable; as defined by the pattern presented below this strategy.
G_Results_Intolerable
No limitations to the collective Results of the evidence with respect to the attribute, where uncertainty is intolerable.
G_Results_Constrained
Limitations are constrained of the collective results with respect to the attribute, where uncertainty must be constrained.
G_Results_Tol
Limitations of the results with respect to the attribute

Figure 67: Results of Evidence

6.5.1 Results – Intolerable Limitations

For cases where limitations in the results of evidence would be intolerable, it is necessary to make three arguments about results, as follows:

- the results of the evidence satisfy the attribute of the constraint (i.e. no additional results are needed to satisfy the attribute);
- the results of the evidence contain no counter evidence (i.e. there is no evidence of faults or errors, or false results); and
- there are no potential sources of counter evidence for which evidence is not available (i.e. there are no results missing that if present could be a source of counter evidence).

These arguments are expressed in Figure 68.

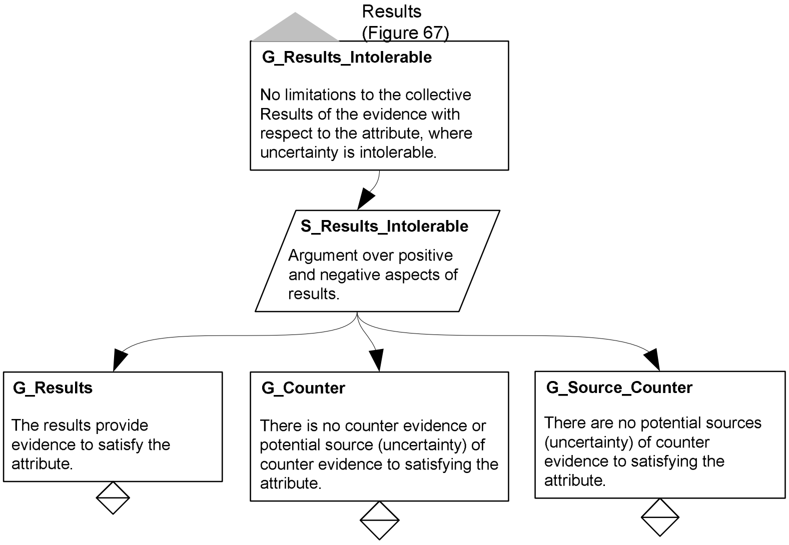

Results
(Figure 67)
G_Results_Intolerable
No limitations to the collective Results of the evidence with respect to the attribute, where uncertainty is intolerable.
S_Results_Intolerable
Argument over positive and negative aspects of results.
G_Results
The results provide evidence to satisfy the attribute.
G_Counter
There is no counter evidence or potential source (uncertainty) of counter evidence to satisfying the attribute.
G_Source_Counter
There are no potential sources (uncertainty) of counter evidence to satisfying the attribute.

Figure 68: Results of Evidence – Intolerable Limitations

6.5.2 Results – Constrained Limitations

For cases where limitations in the results of evidence would be constrained, it is necessary to adapt the three arguments from those used for intolerable to be permissive of counter evidence, but in way that it is constrained, as follows:

- the results of the evidence contribute towards satisfying the attribute of the constraint, but it is not possible to claim that the results are complete;
- the results may contain counter evidence, but the counter evidence is limited such that it does not invalidate the established results; and
- the potential sources of counter evidence, which exist because the results are not complete, are limited such that they do not threaten the established results.

These arguments are expressed in Figure 69.

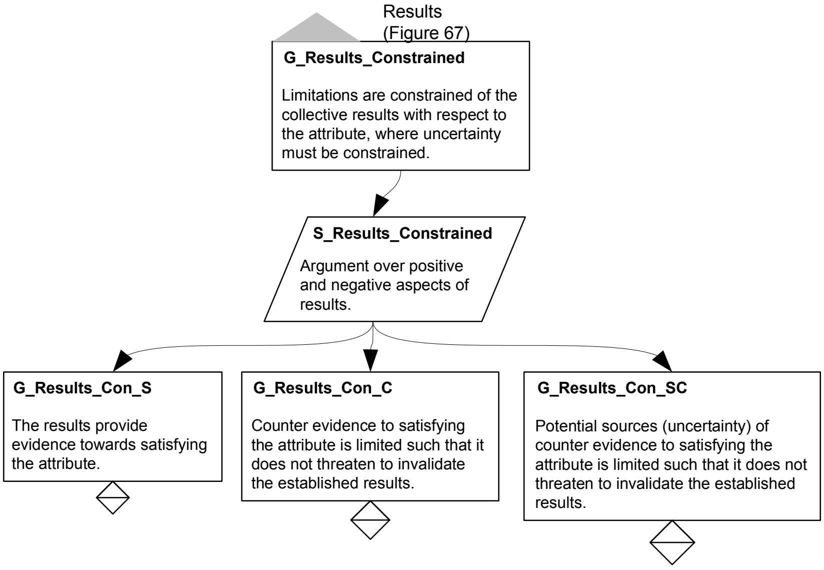

Results (Figure 67)
G_Results_Constrained
Limitations are constrained of the collective results with respect to the attribute, where uncertainty must be constrained.
S_Results_Constrained
Argument over positive and negative aspects of results.
G_Results_Con_S
The results provide evidence towards satisfying the attribute.
G_Results_Con_C
Counter evidence to satisfying the attribute is limited such that it does not threaten to invalidate the established results.
G_Results_Con_SC
Potential sources (uncertainty) of counter evidence to satisfying the attribute is limited such that it does not threaten to invalidate the established results.

Figure 69: Results of Evidence – Constrained Limitations

6.5.3 Results – Tolerable Limitations

For cases where limitations in the results of evidence are tolerable, it is necessary to adapt the three arguments from those used for constrained to be permissive of counter evidence in way that may not be constrained, as follows:

- the results of the evidence contribute towards satisfying the attribute of the constraint, but it is not possible to claim that the results are complete;
- the results may contain counter evidence, but the counter evidence is limited such that it does not invalidate the established results; and

- the potential sources of counter evidence, which exist because the results are not complete, may not be limited and as such they may threaten the established results.

These arguments are expressed in Figure 70.

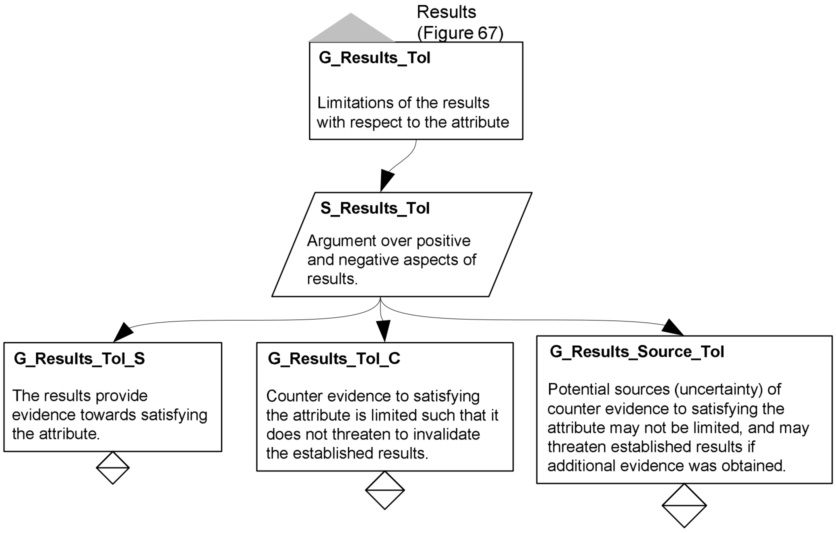

Results
(Figure 67)
G_Results_Tol
Limitations of the results with respect to the attribute
S_Results_Tol
Argument over positive and negative aspects of results.
G_Results_Tol_S
The results provide evidence towards satisfying the attribute.
G_Results_Tol_C
Counter evidence to satisfying the attribute is limited such that it does not threaten to invalidate the established results.
G_Results_Source_Tol
Potential sources (uncertainty) of counter evidence to satisfying the attribute may not be limited, and may threaten established results if additional evidence was obtained.

Figure 70: Results of Evidence – Tolerable Limitations

6.6 Assurance of Evidence

6.6.1 Defining the Evidence Safety Assurance Level (ESAL) Concept

This thesis proposes a framework that includes the concept of an Evidence Safety Assurance Level (ESAL) for determining the requirements for arguments about the 'tolerability of limitations' of evidence. The ESAL provides an implementation of the meta-arguments for 'tolerability of limitations' for assuring the applicable attribute of the software lifecycle product. The ESAL serves two functions. The first is to set benchmarks for the importance (i.e. relationship to the CSAL) of specific attributes in assuring the specific 'constraint'. The second is to set benchmarks for argument construction for:

- relevance of evidence (and the combination of methods or techniques from which evidence is generated) with respect to the attribute of the software lifecycle product in the context of the 'constraint',
- trustworthiness of the evidence (i.e. to what extent can the results of the evidence be tolerated to be incorrect?), and
- results of the evidence (i.e. what the evidence actually shows?) to ensure that the presence of counter evidence is appropriately understood.

Three ESALS are proposed as presented in Table 24 (see over).

Tolerability of Limitations to Assuring Attribute	**Relevance of Evidence**	**Trustworthiness of Evidence**	**Results of Evidence**
Intolerable (ESAL3) – *limitations in evidence would be intolerable*	No limitations to the collective relevance of the method or methods' with respect to the attribute Limitations of each method are systematically identified and treated by the application of complementary methods.	No limitations to the evidence's trustworthiness with respect to the attribute. Limitations of the trustworthiness of evidence are systematically identified and treated by the application of appropriate competencies, reviews and inspections, and independence.	The results of the method or methods provides evidence of satisfying the attribute AND there is no counter evidence or potential source (uncertainty) of counter evidence to satisfying the attribute
Constrained (ESAL2) – *limitations in evidence would be tolerable provided those limitations are constrained with respect to relevance, trustworthiness and results*	Constrained limitations to the method/s relevance with respect to the attribute Limitations of each method are systematically identified and treated where practicable by the application of complementary methods. Non-treatment of a limitation should not introduce uncertainty grossly disproportionate to the limitation such that it would likely lead to a violation of the constraint	Constrained limitations to the evidence's trustworthiness with respect to the attribute. Limitations of the trustworthiness of evidence are systematically identified and where practicable treated by the application of appropriate competencies, reviews and inspections, and independence. Non-treatment of a limitation should not introduce uncertainty grossly disproportionate to the limitation such that it would like lead to a violation of the constraint	Results of the method or methods provides evidence of satisfying the attribute AND counter evidence to satisfying the attribute is limited such that it would not likely lead to violation of the constraint Uncertainty is constrained such that counter evidence is unlikely.
Tolerable (ESAL1) – *limitations in evidence would be tolerable*	Notable limitations to the method or method's relevance with respect to the attribute. Limitations of each method may not be systematically identified and treated where practicable by the application of complementary methods.	Notable limitations to the evidence's trustworthiness with respect to the attribute.	Results of the method or methods may provide evidence of non-satisfaction of the attribute and/or violation of the constraint OR counter evidence indicates possible violation of the constraint OR uncertainty may be substantial

Table 24: Evidence Safety Assurance Level (ESAL) Definitions

6.6.2 ESAL Assignment Methodology

Section 2.4 identified that there are two factors that must be considered in the assignment of an assurance level, regardless of how it is defined. These are:

- what the software level is being assigned to, and
- how the assignment is performed.

Section 2.4 also identified that there are established instances where assurance levels are assigned to safety functions, configuration items, safety requirements or safety objectives.

The ESAL is intended to be assigned to a specific attribute of a lifecycle product and its associated tolerability. In some respects this is equivalent to the assigning of evidence assurance levels based on the importance of the evidence in the argument as performed by existing evidence assurance approaches. Each attribute will have an assigned tolerability and ESAL, and the ESAL assignment may differ across differing attributes within the same software product, depending on the importance of lifecycle product and the attribute in preserving the chain of evidence.

ESAL assignment should be performed by:

1. Identifying the applicable attribute of the software lifecycle product and the associated evidence tolerability level (i.e. Intolerable, Constrained, Tolerable), as determined from the architectural and claims assurance activities (refer Chapters 4 and 5 respectively).
2. Performing ESAL assignment for the attribute of the software lifecycle product with respect to the ‘constraint’ using Table 24, as follows:
   a. For the identified tolerability level (i.e. Intolerable, Constrained, Tolerable), identify the corresponding row from column 1 of Table 24.
   b. Set the requirements for the Relevance argument using column 2 of Table 24 for the corresponding row assigned in step 2a.
   c. Set the requirements for the Trustworthiness argument using column 3 of Table 24 for the corresponding row assigned in step 2a.
   d. Set the requirements for the Results argument using column 3 of Table 24 for the corresponding row assigned in step 2a.

Once the ESAL has been assigned, it is necessary to establish if/how the ESAL will be achieved. Section 6.9 explains how this is done, but first we consider some related topics.

6.7 Dealing with the Human Element in Trustworthiness

The generation of much evidence is highly dependent on human involvement, and this introduces a highly subjective element. While evidence that is known to be incorrect is easy to classify as not trustworthy, deciding if evidence which is presumed correct is a more difficult proposition. Furthermore, the typical methods or techniques that improve a piece of evidence's trustworthiness (e.g. reviews and inspections) are also subjective and dependent on human involvement. Hence compared with relevance and results of evidence, arguments about trustworthiness will be more subjective. Therefore there are difficulties in implementing the approach that parallels the approach for 'relevance'.

Competency frameworks, such as (The IET, 1999), and its later evolution (The IET, 2007) provide a means of measuring the human element, although adherence to the categories of competencies in these frameworks is still subjective.

In paradigms such as model based development, there is motivation to utilise a larger number of tools in the development of software to reduce the opportunity for humans to introduce errors, however these tools still have to be built by someone, usually a human. So the challenge of trustworthiness does not go away, it simply moves somewhere else in the overall argument.

Therefore, there are difficulties in implementing the approach described by this thesis that parallels the approach for 'relevance' which reasons about systematic identification and treatment of limitations. While it is possible to reason about the limitations of human involvement in developing evidence, in reviews and inspections, and the impact of independence being systematically identified and treated, in the more specific context, that approach is less practical. This is because the limitations might vary significantly depending on the competencies of the specific people involved throughout, their state of mind and mental condition or endurance throughout the activity, and the inevitable human error factor. It will also not be possible to benchmark competencies between different system developments because arguments in this context are entirely flexible. Developers may argue that they are competent, and that the reviews they carried out the development were effective. In practice this may not be the case.

Hence limitations will be potentially difficult to use as any basis of comparison with benchmarks. Therefore, trustworthiness of evidence may benefit from an increased level of prescription over other parts of the framework, and this shall be examined.

Table 25 presents an example approach as to how the regulator might set benchmarks for measures of trustworthiness. This approach has been derived from an analysis of evidence trends from real world systems, such as those analysed in the evaluation discussed in Chapter 4 and Chapter 10. The approach is intended to set benchmarks that take into account the variability of human involvement and thus avoid the need to systematically model the resultant limitations of human involvement, which are more difficult to reason about in practice. Conceptually, this approach is similar to what current assurance standards prescribe. However it does provide a focus to ensure the trustworthiness of evidence is considered within the appropriate parts of the framework.

6.7.1 Competency in Trustworthiness

It has already been suggested that competency assessments are subjective. However, the competency element cannot be ignored when establishing the trustworthiness of evidence, both in terms of the generation of the evidence and the review or inspection of the evidence. Competency assessment frameworks are prominent within the established professions. Since establishing competency through competency frameworks by professional bodies is established practice across the known professions (Mason & Friedman, 2004), including but not limited to surveying, medicine, actuarial science, law, dentistry, engineering, architecture, and pilots; then, despite the subjectivity with these approaches, it is possible to conclude there is consensus on such as approach in professions where human safety can be at risk. In safety assurance, at least one such competency framework exists (refer to (The IET, 2007)) which can be the basis of competency evidence for safety assurance.

However, in any engineering organisation there will often be a wide range of staff, some apprentices, some un-qualified, usually being supervised by a smaller number of staff with recognised competencies. Hence evidence may often be produced by personnel that don’t have recognised competencies. Any such evidence framework relating to competency evidence has to recognise that this occurs. At the same time though, any such framework must address questions such as is it tolerable for a non-expert to be responsible for the generation or review/inspection evidence for the most critical systems? Should there be evidence that the generator of important trustworthiness evidence be recognised by their peers as being an expert? These are important questions, but questions for which responses will vary across society.

Trustworthiness	Develop Competency (Minimum)^	Reviews and Inspections (Minimum)			Comparative Evidence (Minimum)	
		Approach	Competency	Independence	Mechanistic Independence	Conceptual Independence
ESAL3 – Intolerable	Expert	Systematic Inspection OR Criteria Review%	Expert	Organisational OR Intellectual	None OR Applied (Expert, Organisation)*	None OR Applied (Expert, Organisational)*%
ESAL2 – Constrained	Practitioner	Criteria Review OR Adhoc Review#	Expert OR Practitioner #	Peer	None OR Applied (Expert, Intellectual)#	None OR Applied (Expert, Intellectual)#
ESAL1 – Tolerable	Supervised Practitioner	Adhoc Review	Practitioner	None	None	None
ESAL0 – No Assurance	No more than Supervised Practitioner	None	N/A	None	None	None

^ - Competency categorisations used from (The IET, 1999) and (The IET, 2007)

% - Conceptual Independence de-obligates the requirement for the review and inspection to be a Systematic Inspection (which inherently contains conceptual independence)

* - Organisational Independence of Mechanistic Independence or Conceptual Independence de-obligates the requirement for the review and inspection to have Organisational Independence (as organisational independence is achieved mechanistically or conceptually).

# - Intellectual Independence of Mechanistic Independence or Conceptual Independence de-obligates the requirement for the review and inspection to have Intellectual Independence (as intellectual independence is achieved mechanistically or conceptually).

Developer Competency – Expert, Practitioner, Supervised Practitioner

Reviews and Inspections – Systematic Inspection, Criteria Review, Adhoc Review

Competency – Expert, Practitioner, Supervised Practitioner

Independence – Organisational Independence, Intellectual Independence, Peer Independence, None

Mechanistic Independence – Applied, None

Conceptual Independence – Applied, None

Note – organisational independence assumes intellectual independence

No independent approach (review and inspection, mechanistic, or conceptual) is ever applied by a lesser competency.

Table 25: A Prescriptive Approach to Measuring Trustworthiness of Evidence

Table 25 addresses these questions by setting benchmarks for ‘recognised’ experts only where limitations in evidence (or trustworthiness of evidence) are intolerable. For attributes where a limitation in evidence may be tolerable, then the competency benchmarks set may be less strict. In the event that the evidence generation or review/inspection does not meet these benchmarks, then this should prompt the presentation of arguments relating to what additional evidence will be provided to resolve such a limitation. Thus Table 25 suggests a way that allows argument for tolerability of limitations to be presented only where the evidence differs from what

Table 25 benchmarks. Evidence of trustworthiness will be required regardless of whether Table 25 is applied or not.

6.7.2 Benchmarks for Reviews and Inspections

Reviews and inspections are heavily human centric activities, and thus are also very subjective. However there are several things which contemporary assurance approaches recognise as improving the trustworthiness of review and inspection activities.

Approach

The first factor that might improve the trustworthiness of a human review or inspection is the approach used to undertake the review or inspection. For example, many different types of structured reviews have been described in the literature. Broadly these types of review can be classified in decreasing order of resulting confidence as:

- *systematic inspection*, which uses a defined set of criteria, review/inspection format, and review/inspection conduct control to systematically review/inspect a piece of evidence (e.g. a Fagan inspection);
- *criteria review*, which uses a defined set of criteria, but without the stricter format and conduct controls used for a systematic inspection (e.g. a checklist based desktop review, or walkthrough); and
- *adhoc review*, which doesn't place any controls on review/inspection criteria, format or conduct (e.g. a peer review).

Table 25 uses this decreasing scale of review/inspection effectiveness to set benchmarks for evidence trustworthiness based on the idea that if a limitation is intolerable, then only a systematic inspection is compelling enough review or inspection evidence.

Competency in Reviews and Inspections

The role of competency in trustworthiness has been discussed; however there are a couple of additional points to make when establishing competency benchmarks for reviews/inspections. The most pertinent question is to understand if a higher competency person can make up for limitations in the competency of the person that developed the piece of evidence? The traditional supervisory model that exists in many engineering businesses and that is advocated by professional bodies such as Registered Professional Engineers of Queensland (RPEQ) (Board of Professional Engineers Queensland, 2013) suggests that it does. Hence Table 25 acknowledges this by permitting non-experts to develop evidence for circumstances where limitations in evidence are tolerable, provided sufficient competency is applied to the review/inspection.

Independence in Reviews and Inspection

The final factor in trustworthiness of a review and inspection is the extent of intellectual independence of the reviewer/inspector from the generation of the evidence. Contemporary assurance practices, such as (RTCA Inc., 2011) and (Ministry of Defence, 1997) usually require either organisational (i.e. the independent safety assessor) or intellectual (i.e. independent from the evidence generating activity) for the most critical software to avoid undue bias from members of the development team. Such a concept is also utilised in Table 25 for setting benchmarks for the independence of a review. For circumstances where a limitation in evidence is intolerable, then contemporary practice suggests that it wouldn't be compelling if the review/inspection did not achieve organisational or intellectual independence.

6.7.3 Benchmarks for Mechanistic and Conceptual Independence

There are cases where a developer may elect that a review or inspection is not sufficient on its own, and that greater trustworthiness can be achieved by using mechanistic or conceptual independence in the conduct of activities. Reviews and inspections will still be relevant, as the developer will still need to establish some trustworthiness in the evidence from each activity, and the resulting comparison of results that will be performed, however, it may be possible to tolerate some reduction in the rigour of individual reviews/inspections in the presence of comparative evidence. Table 25 uses this concept to set benchmarks for evidence trustworthiness when mechanistic or conceptual independence is used as a source of comparative evidence.

This section illustrates one possible way that trustworthiness of evidence may benefit from an increased level of prescription over other parts of the framework.

6.8 Relationship to the Assurance Deficit

The ‘tolerability of limitations’ concept proposed in this thesis has similarities to the *Assurance Deficit* concept proposed by (Menon, et al., 2009), albeit developed independently. However, there are a number of differences. The ‘tolerability of limitations’ approach is concerned with presenting arguments about the impact of limitations of evidence on preserving the chain of evidence between constraints and implementation through the hierarchy of lifecycle products. However, the *Assurance Deficit* is a more general acknowledgement of a shortfall in assurance. The ‘tolerability of limitations’ approach recognises that there is never absolute assurance (i.e. perfect assurance is never achievable), and that arguments will always be required to justify the limitations in the specific context. The necessity for the strength of the argument will come from the importance of the limitation.

The *Assurance Deficit*, by definition, is with reference to a defined level of assurance or benchmark. There needs to be a benchmark, else there is nothing to measure the deficit against. However, the supporting guidance for the assurance deficit work, mostly avoids defining the benchmark, and instead focusses on describing the generic properties of evidence generation and usage in arguments (Menon, et al., 2009). In making this observation, it is noted that the *Assurance Deficit* approach is intended to be completely general (i.e. independent of domain or application), whereas the ‘tolerability of limitations’ concept been developed of focusing more narrowly on avionics systems with well-defined architectural approaches. However, it is evident from the presentation of the ‘tolerability of limitations’ concept in this chapter that it is also possible to use this concept generically.

Because of these differences, the ‘tolerability of limitations’ approach has several key advantages approach over the *Assurance Deficit* approach. Namely the ‘tolerability of limitations’ approach addresses several key limitations of the *Assurance Deficit* concept (as it is described in (Menon, et al., 2009)). These are as follows:

- Absolute assurance is never attainable, so there is always an assurance deficit. What is more important is determining if it impacts safety, and the value in providing additional evidence to address the limitation.
- ‘tolerability of limitations’ sets benchmarks for where an assurance deficit would be tolerable or intolerable – and is explicit in the rationale behind the tolerability;

- ‘tolerability of limitations’ provides fidelity of assurance claims and attributes of lifecycle products at a level that is sufficiently detailed to provide a clear taxonomy of evidence; and
- ‘tolerability of limitations’ clearly distinguishes between lifecycle product attributes with binary attributes and those where there is greater potential for justified tolerability in satisfaction.

6.9 Defining a Process for Applying the ESAL Concept

With the attribute identified, the level of tolerability of limitations in evidence identified, and the ESAL assigned, it is necessary to define an overall lifecycle process for using these concepts. Figure 71 provides an overview of the process, which incorporates those sub-processes defined in this chapter.

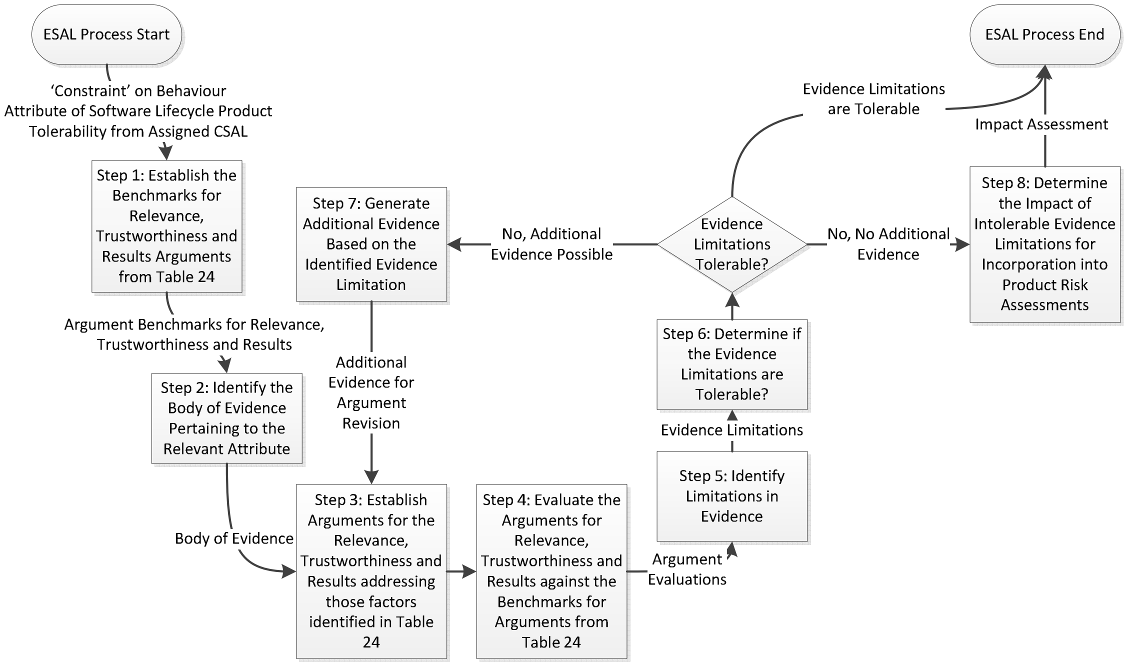

ESAL Process Start
‘Constraint’ on Behaviour
Attribute of Software Lifecycle Product
Tolerability from Assigned CSAL
Step 1: Establish the Benchmarks for Relevance, Trustworthiness and Results Arguments from Table 24
Argument Benchmarks for Relevance, Trustworthiness and Results
Step 2: Identify the Body of Evidence Pertaining to the Relevant Attribute
Body of Evidence
Step 3: Establish Arguments for the Relevance, Trustworthiness and Results addressing those factors identified in Table 24
Step 4: Evaluate the Arguments for Relevance, Trustworthiness and Results against the Benchmarks for Arguments from Table 24
Argument Evaluations
Step 5: Identify Limitations in Evidence
Evidence Limitations
Step 6: Determine if the Evidence Limitations are Tolerable?
Evidence Limitations Tolerable?
No, Additional Evidence Possible
Step 7: Generate Additional Evidence Based on the Identified Evidence Limitation
Additional Evidence for Argument Revision
No, No Additional Evidence
Step 8: Determine the Impact of Intolerable Evidence Limitations for Incorporation into Product Risk Assessments
Impact Assessment
Evidence Limitations are Tolerable
ESAL Process End

Figure 71: ESAL Process Overview

The following provides elaboration of each of the ESAL process steps:

6.9.1 Step 1 – Establish Benchmarks

a. *Establish the benchmark for Relevance of evidence from the corresponding row of Column 2 of Table 24.*

b. *Establish the benchmark for Trustworthiness of evidence from the corresponding row of Column 3 of Table 24.*

c. *Establish the benchmark for Results of evidence from the corresponding row of column 4 of Table 24*

Example – A-DHC-4

In Chapter 5 the A-DHC-4 example considered the constraint *"Value failures of type fixed of sensor_data.attitude#1 shall be detected using reasonability checking against expected attitude data based on sensor_data.attitude#2, aircraft motion and lateral mode. Handling shall set the sensor_data.attitude#1.valid flag to invalid".* Specifically the Accuracy and Robust with Higher Abstraction attributes of the Low Level / Detailed Design Requirements lifecycle product were examined.

For CSAL 3, the tolerability benchmark for both these attributes is ESAL 3 – Intolerable, as per Appendix B and Table 24. Therefore, the benchmarks are as follows:

- Relevance: *No limitations to the collective relevance of the method or methods' with respect to the attribute. Limitations of each method are systematically identified and treated by the application of complementary methods.*
- Trustworthiness: *No limitations to the evidence's trustworthiness with respect to the attribute. Limitations of the trustworthiness of evidence are systematically identified and treated by the application of appropriate competencies, reviews and inspections, and independence.*

In this example, Relevance is assessed for the Robust with Higher attribute and Trustworthiness is assessed for Accuracy. In practice Relevance, Trustworthiness and Results would be assessed for each attribute. For the purposes of this example, let's also consider the results category for the attribute Compliance with Higher at ESAL 3.

- Results: *The results of the method or methods provide evidence of satisfying the attribute AND there is no counter evidence or potential source (uncertainty) of counter evidence to satisfying the attribute.*

These are the benchmarks our arguments about evidence need to achieve.

6.9.2 Step 2 – Identify the body of Evidence pertaining to the relevant Attribute

a. *Identify the body of evidence pertaining to the relevant attribute.*

b. *Assemble the evidence ready for evaluation.*

c. *Categorise the evidence based on both the attribute and the categories of evidence identified in Section 6.1.*

Example – A-DHC-4

The evidence has been identified and categorised as per Table 26.

6.9.3 Step 3 – Establish Arguments

a. *Establish the Relevance argument that relates the applicable product evidence to the attribute, and that adheres to the benchmark for the argument specified by column 2 of Table 24.*

b. *Establish the Trustworthiness argument that relates the applicable process evidence to the attribute, and that adheres to the benchmark for the argument specified by column 3 of Table 24.*

c. *Establish the Results argument that relates the applicable evidence to the attribute, and that adheres to the benchmark for the argument specified by column 4 of Table 24.*

<u>Example – A-DHC-4</u>

The arguments for relevance, trustworthiness and results are summarised in textual form in Table 27.

6.9.4 Step 4 – Evaluate the Arguments

a. *Evaluate the Relevance, Trustworthiness, and Results arguments against the benchmark for the argument specified by columns 2, 3 and 4 of Table 24 respectively.*

<u>Example – A-DHC-4</u>

The benchmarks for relevance, trustworthiness and results for the attributes considered in this example, are intolerable, thus implying no limitations are permissible. Step 5 identifies the limitations that have been identified through the evaluation.

Lifecycle Product	**Attribute**	**Evidence Category**	**Evidence Type**	**Evidence**
Low Level / Detailed Design Requirements Lifecycle Product	Verification Group: Robust with Higher	Relevance	Elaborating Rationale	Software Design Description Software Development Plan Software Test Plan Software Test Description Software Test Procedures and Cases
	Specification Group: Accuracy	Trustworthiness	Competency (Domain)	Design Authority Letter of Engineering Authority Qualification, Training and Experience Records Competency Assessment by Design Authority
			Competency (Method)	As for domain competency, but for method instead.
			Method (Suitability)	Software Development Plan Model-Based Development Procedure Model-Based Development Reports Review and Inspections Procedure Peer Review Inspection Records
			Method (Rigour)	As for method suitability, but for rigour instead.
			Independence (Review/Inspection)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records
			Independence (Complementary)	No Evidence
	Verification Group: Compliance with Higher	Results	Product Defining Information	Requirements Management Database containing Software Requirements and Traceability Data Software Design Description Software Test Description Software Test Procedures and Cases Software Test Results

Table 26: Examples of Evidence Categorisation for Selected Attributes

Attribute	Evidence		Evidence	Assessment	Limitations and Treatments
	Cat	Type			
Verification Group: Robust	Relevance	Elaborating Rationale	Software Design Description Software Development Plan Software Test Plan Software Test Description Software Test Procedures and Cases	Method is robustness demonstration by rig testing. Criteria for robustness cases are defined in SDP and meets RTCA/DO-178B robustness criteria. Testing carried on target hardware, except for code for hardware exceptions that the test rig can't replicate.	Robustness of Code for hardware exceptions can't be demonstrated on the rig. Evidence from another method is required.
Specification Group: Accuracy	Trustworthiness	Competency (Domain)	Design Authority Letter of Engineering Authority (LEA) Qualification, Training and Experience Records Competency Assessment by Design Authority	Regulator has authorised design authority based on competency assessment. Design authority has evaluated requirements staff QTE against flight control system domain competency requirements. All developers and reviewers are 'Experts, except 'Bloggs' who isn't allowed to generate evidence on this project.	No limitations. All staff are assessed as competent. Staff have undergone refresher training on flight control system fundamentals.
		Competency (Method)	Design Authority Letter of Engineering Authority Qualification, Training and Experience Records Competency Assessment by Design Authority	Regulator has authorised design authority based on competency assessment. Design authority has evaluated requirements staff QTE against Modelling and simulation tool competency requirements. All developers and reviewers are 'Experts'.	No limitations. All staff are assessed as competent. Staff have undergone refresher training on control system design using Modelling tool.
		Method (Suitability)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records	Method is trustworthiness by adhoc peer review. Adhoc peer review is not systematic enough to assure high level of trustworthiness.	Adhoc peer review is not systematic. A systematic review is required.
		Method (Rigour)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records	No evidence that adhoc peer review was carried out in accordance with the review and inspection procedure, otherwise a more rigorous review method would have been used. No quality assurance records to confirm conformity to review processes was assessed.	No conformity review evidence by quality assurance pertaining to reviews/inspections and review/inspection record management.
		Independence (Review / Inspection)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records	Peer review is undertaken by another member of the requirements team. No organisational independence and no intellectual independence.	No organisational or intellectual independence in review/inspection of requirements data.
		Independence (Complementary)	No Evidence	No complementary method using mechanistic or conceptual methods undertaken.	No mechanistic or conceptual independence in review/ inspection of requirements data
Verification Group: Compliance with Higher	Results	Product Defining Information	Requirements Management Database containing Software Requirements and Traceability Data Software Design Description Software Test Description Software Test Procedures and Cases Software Test Results Software Problem Report Database	Test descriptions cover the criteria for completeness of requirements based testing for 'constraint'. Test procedures and cases are correct for 'constraint'. Test results are all passes. No additional testing is required. No test failures identified. No software problem reports are open.	No limitations identified.

Table 27: Example Assessments/Arguments of Limitations

6.9.5 Step 5 – Identify Limitations in Evidence

a. *Identify limitations in evidence with respect to Relevance*

b. *Identify limitations in evidence with respect to Trustworthiness*

c. *Identify limitations in evidence with respect to Results*

<u>Example – A-DHC-4</u>

Limitations in evidence for Relevance and Trustworthiness are identified in Table 27 for the attributes Robust with Higher and Accuracy respectively. There are no limitations in evidence for Results of Compliant with Higher.

6.9.6 Step 6 – Determine if the Evidence Limitations are Tolerable

a. *If the evidence limitations are intolerable, and the generation of additional evidence is possible, then go to Step 7.*

b. *If the evidence limitations are intolerable, and the generation of additional evidence is not possible, then go to Step 8.*

c. *If the evidence limitations are tolerable, then go to ESAL Process End.*

<u>Example – A-DHC-4</u>

The limitations of the relevance of evidence with respect to the Robust with Higher attribute are assessed to be intolerable based on the requirements of ESAL3. Refer to Step 7 for their resolution.

The limitation of the trustworthiness of evidence with respect to the Accuracy attribute is also assessed to be intolerable based on the requirements of ESAL3. Refer to Step 7 for their resolution.

There are no limitations of results of evidence with respect to the Compliance with Higher attribute, and thus this attribute is considered satisfied.

6.9.7 Step 7 – Generate Additional Evidence Based on the Identified Evidence Limitation

a. *Generate additional evidence to resolve the evidence limitation with respect to relevance, trustworthiness or result respectively.*

b. *Revise the arguments established in Step 3 to take into account the additional evidence.*

Example – A-DHC-4

The revised evidence provided in support of each of these attributes is described in Table 28. Additional evidence is shown in bold italics.

Lifecycle Product	**Attribute**	**Evidence Category**	**Evidence Type**	**Evidence**
Low Level / Detailed Design Requirements Lifecycle Product	Robust with Higher	Relevance	Elaborating Rationale	Software Design Description Software Development Plan Software Test Plan Software Test Description Software Test Procedures and Cases ***Additional Host Based Test Descriptions, Procedures, and Cases*** ***Model-Based Development Tool Analysis Results*** ***SPARK Analyser Procedure and Results***
	Accuracy	Trustworthiness	Competency (Domain)	Design Authority Letter of Engineering Authority Qualification, Training and Experience Records Competency Assessment by Design Authority
			Competency (Method)	Design Authority Letter of Engineering Authority Qualification, Training and Experience Records Competency Assessment by Design Authority
			Method (Suitability)	Software Development Plan Model-Based Development Procedure Model-Based Development Reports Review and Inspections Procedure Peer Review Inspection Records ***Inspections by Walkthrough Records***
			Method (Rigour)	Software Development Plan Model-Based Development Procedure Model-Based Development Reports Review and Inspections Procedure Peer Review Inspection Records ***Quality Assurance Records for Inspections.*** ***CM records for Inspection Records.*** ***Problem Reporting Records***
			Independence (Review / Inspection)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records ***Walkthrough Evidence by IV&V Team***
			Independence (Complementary)	***Walkthrough Evidence by IV&V Team*** ***Model-Based Development Simulation Results***
	Compliance with Higher	Results	Product Defining Information	Requirements Management Database containing Software Requirements and Traceability Data Software Design Description Software Test Description Software Test Procedures and Cases Software Test Results

Table 28: Examples of Revised Evidence Categorisation for Selected Attributes

The revised tolerability of limitations arguments are shown in Table 29.

Attribute	Evidence		Evidence	Assessment	Limitations
	Cat	Type			
Verification Group: Robust with Higher	Relevance	Elaborating Rationale	Software Design Description Software Development Plan Software Test Plan Software Test Description Software Test Procedures and Cases ***Additional Host Based Test Descriptions, Procedures, and Cases*** ***Model-Based Development Tool Analysis Results*** ***SPARK Analyser Procedure and Results***	Method is robustness demonstration by rig testing. Criteria for robustness cases are defined in SDP and meets RTCA/DO-178B robustness criteria. Testing carried on target hardware, except for code for hardware exceptions that the test rig can't replicate. ***Exception related code robustness established via model-based development methodology and SPARK analysis. These methods don't address target computer behaviour, but this limitation is addressed by the analysing similar results obtained from robustness testing on the target.***	***No remaining limitations.***
Specification Group: Accuracy	Trustworthiness	Competency (Domain)	Design Authority Letter of Engineering Authority (LEA) Qualification, Training and Experience Records Competency Assessment by Design Authority	Regulator has authorised design authority based on competency assessment. Design authority has evaluated requirements staff QTE against flight control system domain competency requirements. All developers and reviewers are 'Experts, except 'Bloggs' who isn't allowed to generate evidence on this project. Staff have undergone refresher training on flight control system fundamentals.	No limitations. All staff are assessed as competent.
		Competency (Method)	Design Authority Letter of Engineering Authority Qualification, Training and Experience Records Competency Assessment by Design Authority	Regulator has authorised design authority based on competency assessment. Design authority has evaluated requirements staff QTE against Modelling and simulation tool competency requirements. All developers and reviewers are 'Experts'. Staff have undergone refresher training on control system design using Modelling tool.	No limitations. All staff are assessed as competent.
		Method (Suitability)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records ***Inspections by Walkthrough Records***	Method is trustworthiness by adhoc peer review. Adhoc peer review is not systematic enough to assure high level of trustworthiness. ***Additional method walkthrough applied. Walkthrough is systematic.***	***No remaining limitations.***
		Method (Rigour)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records ***Quality Assurance Records for Inspections.*** ***CM records for Inspection Records.*** ***Problem Reporting Records***	***Walkthrough carried out in accordance with walkthrough procedure. Walkthrough is systematic.*** ***Quality assurance conformity review on walkthrough records confirms to process.***	***No remaining limitations.***
		Independence (Review / Inspection)	Software Development Plan Review and Inspections Procedure Peer Review Inspection Records ***Walkthrough Evidence by IV&V Team***	***Walkthrough carried out by IV&V team, with development team participation. IV&V achieve both organisation and intellectual independence.***	***No remaining limitations.***
		Independence (Complementary)	***Walkthrough Evidence by IV&V Team*** ***Model-Based Development Simulation Results***	***Model-Based Development Model Analysis/Simulation of Requirements provides conceptual independence to review/inspection approach.***	***No remaining limitations***
Verification Group: Compliance with	Results	Product Defining Information	Requirements Management Database containing Software Requirements and Traceability Data Software Design Description Software Test Description Software Test Procedures and Cases Software Test Results Software Problem Report Database	Test descriptions cover the criteria for completeness of requirements based testing for 'constraint'. Test procedures and cases are correct for 'constraint'. Test results are all passes. No additional testing is required. No test failures identified. No software problem reports are open.	No limitations identified

Table 29: Example Revised Assessments/Arguments of Limitations

6.9.8 Step 8 – Determine the Risk Impact of Intolerable Evidence Limitations

a. *Determine the impact of intolerable evidence limitations for communication to higher level product risk assessments, which will include consideration of:*

   i. *the attribute against which the evidence shortfalls exists,*

   ii. *the applicable 'constraint' to which it relates and the corresponding CSAL assignment for that 'constraint', and*

   iii. *the other fault prevention or fault tolerance mechanisms employed by the architecture to treat the source of fault.*

<u>Example – A-DHC-4</u>

In addition to the attributes presented in the example above, assume that an evidence shortfall has also been identified against the Traceable to High Abstraction attribute for the Low Level / Detailed Design Requirements Lifecycle Product. Chapter 8 provides guidance and continuation of the A-DHC-4 example on revising risk assessments based on counter evidence.

6.10 Summary

This chapter has examined how knowledge of tolerability of limitations can be obtained through evaluation of evidence, based on categorisation and type of evidence with respect to the attributes of lifecycle products defined in Chapter 5. Specifically, several categorisations of evidence based on claim, and based on the type of the evidence have been examined and categorisations of evidence established. This chapter then illustrates how the categorisations of evidence can be used to reason about the suitability of evidence with respect to attributes.

The role of relevance with respect to product and process evidence has been examined and expressed within meta-arguments, thus providing a means to satisfy Principle D of Figure 50 for product and process evidence respectively. The role of trustworthiness in relation to both product and process evidence has also been explained (satisfying Principle X of Figure 50). A means of evaluating the impact of limitation in both product and process evidence with respect to both relevance and trustworthiness has been expressed using meta-arguments (satisfying Principle Y of Figure 50).

Using the identified categorisations of evidence, the ESAL framework has been proposed for evaluating the tolerability of limitation in evidence with respect to attributes of lifecycle products. The sources of limitations in evidence are categorised based on relevance, trustworthiness and results, and the type of evidence used to support these claim types. Arguments are required for each of these categories to relate the evidence to the attribute, and to reason about limitations in evidence. For trustworthiness, which may not be well suited to the argumentation approach, an alternative approach has also been suggested that provides a greater level of prescription to minimise the need for subjective arguments.

A process has been defined for applying the evidence assurance and assigning ESALs. An example has been presented which illustrates how the evidence assurance can be applied. The example illustrates that it is feasible to construct arguments about 'tolerability of limitations' in evidence with respect to attributes of software lifecycle products.

The ESAL framework has been developed to also adhere to the usability guidelines identified in Figure 50. The ESAL framework minimises variability (adhering to Guideline 1) by specifying deterministic requirements for evaluating evidence against attributes based on both category and type of evidence. The ESAL framework minimises subjectivity (adhering to Guideline 2) by ensuring that the evidence categorises are mutually exclusive and through traceability to a specific attribute of the lifecycle product. Through exposing limitations of methods with respect to the evidence categories and type categorisations, it is also feasible that subjectivity may be further reduced once limitations of method become systematically documented and widely acknowledged by the industry. Finally, the articulation of differences between the ESAL levels helps assessors distinguish between tolerable cases for limitations and intolerable cases.

This chapter completes the final element of the architectural (Chapter 4), product behavioural knowledge and claims (Chapter 5), and evidence aspects of the framework proposed by this thesis. Chapter 7 examines how this framework could be contracted for in the military aviation environment, and Chapter 8 examines how the impact on safety risk can be evaluated as a result of limitations in evidence.

7 Contracting for Architectural, Claims and Evidence Assurance

Chapter 2 established that contracts are used to achieve the regulatory and safety assurance outcomes for military programs. Military contracts achieve this by referencing the applicable regulations and safety standards. However, Section 2.3 established that this does not guarantee that safety assurance will be successful.

Section 2.3.1 indicated that a value for money and on-time/on-budget contract will only be possible when both the acquirer's and supplier's expectations are aligned. This implies that the acquirer and supplier must align their expectations of the product and evidence requirements prior to contract signature. When there is ambiguity in a contract, a contract dispute will often find in favour of the organisation that didn't draft the contract (i.e. the supplier). Hence supplier understanding is predicated by the clarity of communication of these expectations by the contract. The contract must communicate certification requirements, include activities and controls for evidence provision, incentives for suppliers to comply, and provide mechanisms for enforcement when suppliers don't comply. This chapter examines how this might be achieved using the approach described in Chapters 4 through 6.

7.1 Integrating Safety Assurance and Tender/Contract

How a safety assurance standard integrates with the contractual lifecycle is an important factor in achieving and demonstrating safety. A safety assurance standard should reduce uncertainty about the delivered product, argument and evidence prior to the establishment of a contract. This is important because acquirer and supplier will seek confidence that the contract will be successful. Similarly, the standard should assist during contract execution. Should safety issues emerge during the contract, then timely and cost-effective resolution will be a goal for both supplier and acquirer. The contract and standard should support the resolution of safety issues, and not hinder it by contributing to a dispute. There is evidence in historical projects that standards, particularly those where product and/or evidence requirements are less prescriptive, actually increase contractual dispute in projects (refer Chapter 10).

An inspection of contemporary safety standards reveals that integration between the standard's lifecycle and contract lifecycle varies significantly between standards. The following sub-sections examine these variations.

7.1.1 ARP4754 and DO-178B

ARP4754 and RTCA/DO-178B don’t mention integration with contracts. This is understandable because they are used where there is legal enforcement of certification requirements. However, these standards can be used to achieve elements of contract integration through the certification authority liaison and artefact requirements within these standards. It is necessary to supplement them with contract requirements in order to interface the standards’ certification environment assumptions to the certification environment of the acquirer. Guidance on a means of doing this is provided in (Directorate General Technical Airworthiness, 2010).

7.1.2 UK Defence Standard 00-56 Issue 4

UK Defence Standard 00-56 Issue 4 mentions the contractor, and defines requirements on contractors with respect to safety. However, limited guidance is provided with respect to how to prepare Statement of Requirement (SOR) and Statement of Work (SOW) clauses for the standard, and the standard doesn’t provide requirements for the provision of arguments or evidence across the contracting lifecycle. Hence, one factor that has limited the effectives of this standard in practice is the lack of contractual implementation guidance (McDermid, 2010).

7.1.3 MIL-STD-882

MIL-STD-882C/D/E includes contract integration. There are references to recommended contract clauses, tender processes and data requirements, although this guidance is not always adhered to by project authorities (Joint Software Systems Safety Engineering Workshop, 2010). The standards don’t address how safety or evidence limitations should be resolved, other than via contractual dispute. They also don’t include information provision required to inform the tender process of architectural and evidence limitations prior to contract signature.

7.1.4 Integrating Safety Standards and Contracts

The requirements of the standards have a substantial effect on the integration of the standard across the tender/contract lifecycle. Therefore, it should be understood what elements of standards and their implementation in contracts, provides appropriate certainty (regarding product and assurance evidence) for acquirers and suppliers? Is it possible to define requirements for safety and assurance standards to achieve effective contract process integration?

Ultimately, it is vital that the regulatory and safety assurance standards used be compatible with the contracts used for military acquisitions, without impairing or detracting from the achievement and demonstration of safety. This chapter investigates an approach to answering the questions from the previous paragraph. Based on the discussion at Section 2.3.1, the focus is on fixed-price contracts.

7.2 Roles for Military System Contracts

Sections 2.2 and 2.3 establish that military system contracts are used for regulation of safety assurance. Their importance is examined in the following sub-sections.

7.2.1 Enforcement of Design Requirements

In military aviation, the airworthiness design requirements (or requirement to establish and agree them) must be included in the contract if they are to apply to the development (refer to Section 2.2.4). This means that the SOR should include or reference applicable airworthiness design requirements, including product safety and safety assurance requirements. In addition, the SOW should include activities to ensure elicitation and agreement of any additional airworthiness or design requirements relevant to the design.

Hence an important role for achieving safety assurance through a contract is to ensure that product design requirements pertaining to safety can be communicated, established and agreed through the contracting process.

7.2.2 Obtaining Assurance Evidence through the Contract

In military aviation, the regulator (as part of the acquirer organisation) obtains evidence required for certification via the contract. This means that the SOW must include applicable activities for the generation of evidence. Delivery versus access to evidence is usually dictated by intellectual property and export control considerations (which are outside the scope of this thesis). Whether delivery of evidence is sought is usually evident from the artefacts listed in the Contract Data Requirements List (CDRL), and supporting Data Item Descriptions (DIDs) which describe content requirements.

Hence an important role for achieving safety assurance is to ensure that the requirements for access and delivery of assurance evidence are explicit. This is more challenging than preparing a CDRL, as it should articulate benchmarks that will assure evidence sufficiency, but without constraining the design solution unnecessarily.

7.2.3 Resolving Shortfalls in Product and Evidence

In the military aviation circumstance, resolving a shortfall in product or evidence will depend on whether it is in or out of scope of the contract. If the issue is within scope, then the onus is on the supplier, but if there is any ambiguity regarding scope of the contract pertaining to the issue, then the onus for resolution is shared by the acquirer. If the supplier and acquirer can't agree that it is wholly within the scope of the contract, then the issue may be the subject of contractual dispute.

The ramifications of a contractual dispute can include cost and schedule implications while the dispute takes place, a requirement to elevate beyond project staff, a requirement to negotiate over contractual interpretation and compliance, etc. These issues potentially degrade the effectiveness of safety regulation achieved through the contract, particularly where projects must seek additional funding from Government (an onerous process) to resolve the safety shortfalls via contract change proposals.

Hence an important role for achieving safety assurance through a contract is to ensure that arrangements for resolving shortfalls in product and evidence are explicit in the contract and meet both acquirer and supplier expectations.

7.3 Contract-based Acquisition Paradigms

Before examining how to contract for architectural, claims and evidence assurance, it is worthwhile clarifying the terminology related to contracts. The three most common acquisition paradigms (Defence Materiel Organisation , 2012) are the:

- Open Tender,
- Restricted Tender, or
- Sole Source Acquisition.

The paradigms pursued by the contracting authority depends on the extent to which:

- the solution will be developmental or off-the-shelf;
- a supplier or suppliers are known prior to the acquisition;
- engaging a larger market improves competition and value for money; and
- engaging a narrower market improves contractual response times.

The following sub-sections summarise the three different paradigms, and emphasises implications for contracting for architectural, claims and evidence assurance.

7.3.1 Open Tender

The Open Tender involves the release of a Request for Tender (RFT) to the whole market. This will be a large number of prospective tenderers, and include a cross section of maturity across the market. Hence it is important within this paradigm that 'strong' tenders can be distinguished from 'weaker' tenders during tender evaluation. A 'strong' tender would include forecast compliance and demonstration with safety objectives, whereas a weaker tender may include substantial uncertainty.

The RFT typically contains a version of the Tender Statement of Requirement (SOR) and a Statement of Work (SOW) which includes:

- the envisaged contract requirements and scope of work (i.e. what the tender is bidding against from a product and evidence perspective), and
- the tender submission requirements and scope of work (i.e. what information the tenderer has to provide as part of the tender for the purposes of tender evaluation).

The RFT responses would then be evaluated and a preferred tenderer identified, with whom contract negotiations would commence. At the time of contract negotiations a draft contract is refined based on the original tender documents, and amended (as necessary) based on any limitations in the preferred tenderers RFT response. Presuming the contract negotiations are successful, contract signature would be achieved.

Note that some tender processes involve an initial release of a Call for Expressions of Interest (EOI) to identify the market, and then release of the RFT to only suitable responses to the EOI. This approach is really a hybrid of the Open Tender and Restricted Tender (refer Section 7.3.2), but with the luxury that the actual tender SOR and SOW can be refined based on the initial look to the market under the Call for EOI.

Where the acquisition or modification is of substantial complexity, then the single phase tendering process may not incentivise suppliers to invest a level of effort to develop their solution to a level that permits effective evaluation. This is often the case for a new aircraft development. In this case a two-phase tender may be more suitable, such as those used in the JSF selection process (JSF Program, 2013). The first phase would identify solutions that accord with the program objectives and use a normal tender construct. The second would be a partially funded tender phase, where funding is provided to a restricted set of tenderers to further develop the tender artefacts supporting evaluation. The second phase is synonymous with a Restricted Tender, but includes funding so that tenderers can invest a level of effort which they are compensated for.

Such options are available where the acquirer is not satisfied that the tenderer is incentivised to offer competitive solutions, or to resolve the uncertainty to a level consistent with the constraints on acquirer funding.

7.3.2 Restricted Tender

Restricted Tender involves the release of the Request For Tender (RFT) to a restricted number of market participants. This subset of market participants will have been predetermined either by a market selection activity (such as a Call for Expressions of Interest, Request for Proposals, etc.), or through market research.

The key factor that distinguishes this approach from the Open Tender is that the tender is restricted to a nominated number of tenders. Otherwise the processes are very similar to the Open Tender. The goal of identifying 'strong' and 'weaker' tenders still remains with respect to safety assurance compliance.

7.3.3 Sole Source

Sole Source Acquisition involves confining the acquisition to a single supplier, because the supplier has been predetermined to provide an off-the-shelf solution, or because the supplier has been assessed as the most suitable. Examples of common circumstances include rapid acquisitions due to operational imperatives, and standing intellectual property restrictions that prevent the work being contracted to another supplier.

For Sole Source Acquisition, the Request For Tender (RFT) is usually replaced by a Request For Quote (RFQ) or Request for Proposal (RFP) to reflect the definite nature of the acquisition. In some cases the proposal request is similar in nature to an RFT, as much of the same information is needed. This step is sometimes overlooked because of perceptions that the project scope is already defined by the solution. While this perception may be true for physical tangibles, it is less applicable to the body of evidence needed to form the safety case. For Sole Source Acquisition, the proposal evaluation and contract negotiations phase usually has a greater burden for establishing evidence requirements into the contract SOW and SOR. If overlooked then the contract will likely be inadequate and result in certification challenges. Once on contract, there little difference between Sole Source and the Open and Restricted Tender approaches.

7.3.4 What Do the Paradigms Mean for Contracts

From Sections 7.3.1 through 7.3.3, it is evident that the acquisition paradigm changes the focus of the contract. In an Open Tender the acquirer has the opportunity to eliminate tenders whose safety objectives or evidence is not to the acquirer's satisfaction. However, in a Sole Source acquisition, the product may already be predetermined, and as such the contract has to predominantly inform the acquirer of the potential risks and evidence shortfalls of the solution to inform decisions on treatment or retention, depending on the capability or operational imperative.

Hence it can be seen that the balance between roles for contracts identified throughout this thesis so far may alter depending on the acquisition paradigm and that the construction of the tender and contract needs to be flexible to accommodate this.

7.4 Impact of Uncertainty at Contract Signature

Sections 2.3 and 7.2 identified several responsibilities of contracts if achievement and demonstration of safety is to be effective. There is an increased risk of a contract being unsuccessful if there is uncertainty with respect to:

- communication and enforcement of design requirements,
- generation and access to assurance evidence, and
- expectations for resolving shortfalls in assurance evidence.

7.4.1 The Gamble of Entering Into Contract

Signing a contract involves a gamble. It is a wager for supplier and acquirer that the supplier can provide a system that meets the acquirer's requirements within the cost and schedule of the contract. Contract success risk is a function of the uncertainty at contract signature. Lots of uncertainty and the odds may be against success; lesser uncertainty and the odds might favour success. Fortunately, the project definition and tender phases provide the contract authority with a way of seeking important information prior to contract signature. This information, if sought and used effectively, can reduce uncertainty, and thus reduce potential contract risks.

How to seek the right information and effectively evaluate it with respect to safety is a challenge. The existing standards and contracting approaches offer limited guidance on how this might be achieved. Industrial examples (refer Chapter 1 and 2, and also Chapter 10) involving project overruns and cancellations due to safety assurance concerns suggests that the current approaches are also insufficient.

7.4.2 Potential Sources of Uncertainty at Contract Signature

To further understand the implications of uncertainty at contract signature for safety it is necessary to establish where it might exist. Uncertainty may exist with the following:

- Will the design requirements proposed by the acquirer be adequate to achieve the safety objectives? Specifically, from an architectural (refer Chapter 4) safety assurance perspective, will:
    - the software and system architecture, including the use of redundancy, diversity, and fault avoidance/tolerance likely permit achievement of the safety objectives?
    - the architecture provide adequate protection against systematic faults?
- Will compliance with the design requirements and safety objectives be compelling based on the evidence provided? Specifically, from a claims and evidence perspective (refer Chapters 5 and 6), will:
    - the behaviours of the system and its software be sufficiently understood and valid under both normal and failure circumstances?
    - these behaviours be appropriate with respect to safety?
    - the evidence support the safety assurance claims made by the supplier about these behaviours?
    - any limitations in evidence be tolerable?
- Will limitations in evidence be resolvable within the scope of the contract? Specifically, what is:
    - within scope?
    - out of scope, requiring a contract change and additional funding?

Whenever there is uncertainty with respect to these questions, then these manifest as contract risks. Uncertainty might undermine the acquirer's aspiration to establish if the system will likely achieve safety. Thus the supplier might be eliminated during the tender evaluation based on perceived uncertainty in suitability of product and evidence (when the product may achieve safety). More seriously, the design solution may be contracted for, yet have unsuitable behaviours. In this case the acquirer may not be able to complete safety certification within the contract. Worse still, it may require the acquirer to retain risks and these risks prove to be intolerable in practice. No acquirer enters into a contract with an aspiration to retain safety risks at the time of delivery.

If these factors are extrapolated, then the result is obvious: have the supplier provide full disclosure to the acquirer during the tender process. However, the realities of the commercial business quickly make this impractical. In domains where highly developmental systems are common-place, it is uneconomical to require suppliers to complete their development lifecycle to the point that answers to the above bullet point questions become certain during the tender process. As only a small percentage of tender responses are actually successful, and tenderers invest substantial resources in preparing them, the acquirer must avoid deterring potentially suitable tenderers due to the level of effort required to tender. Therefore, the tender response must provide for sufficient disclosure and understanding, but while ensuring the minimum imposition on tenderers. This is a difficult balance.

7.4.3 Acquirer and Supplier Motivations

Acquirers and suppliers have motivations, aspirations and perspectives which are a unique contrast between goals for project success, mixed with broader commercial goals and commercial restrictions. Each of these will vary between every acquirer, supplier and circumstance. The most obvious motivations for the acquirer and supplier are that the solution will achieve the safety objectives, and that the evidence will show this. But the additional motivations vary the perspective on achievement. Acquirer motivators include:

- satisfying capability requirements,
- credibility of supplier cost and schedule forecasting,
- avoiding contract changes (because they are onerous to get approved),
- costs of solutions falling within notional budgets (because getting additional funding often involves going back to government, which is difficult), and
- delivery within capability fielding/scheduling requirements.

Supplier motivators include:

- providing a competitive tender cost/schedule,
- preservation of profit margins within the contract price,
- avoidance of contract penalties,
- ensuring that out of scope work requires a contract change (to protect the profit margin with the contract), and
- delivery of a broadly satisfactory product with minimal application of resources.

These motivators are linked because cost and schedule are required to produce evidence, and evidence is required to show the solution meets objectives. Because of this dependency, these motivators may conflict, and may cause divergence in supplier and acquirer behaviours. The emergent (commercial) behaviours that arise depend on the relationship between supplier and acquirer, the seriousness of the safety concerns or cost impacts, and the supplier's and acquirer's worldviews (McDermid & Rae, 2012).

Given these issues, how might a framework be established to ensure that uncertainty at the time of contract signature can be bounded? What is the compromise that enables the appropriate design solution to be identified during tender processes, and this solution to be achieved during contract execution?

The remainder of this chapter examines how an approach may be established. Illustration of the benefits of the approach will be via the fictional example used throughout this thesis. Consider the upgrade of the DHC-4 Caribou's flight control system to a digital flight control system. The objective of the acquirer is to achieve this upgrade, including the safety regulatory functions on behalf of the acquirer's regulatory authority, through a contract. The tender process for this contract needs to identify the possible solutions that will achieve and demonstrate safety. The following sections examine how this can be effectively achieved.

7.5 Bounding Uncertainty Using the Tender Process

The tender phase provides a means for the acquirer to seek information prior to contract. This information, can reduce uncertainty and thus reduce potential contract risks. How much the uncertainty has to be reduced is an important question, and this suggests the concept of bounding uncertainty.

Firstly, it is important to elaborate what is meant by bounded uncertainty in this context. Put in contractual terms, it is establishing limits (upper bounds) on the cost of producing a product that achieves safety and an acceptable safety case that demonstrates safety. Bounds can be narrowed by the provision of information to the acquirer from the supplier during pre-contract phases (e.g. tender phase). The limiting factor on information provision will be the affordability, for a tenderer, of conceptual and preliminary phases of requirements and design lifecycle phases within the resources that are commercially viable during the tender.

In Section 7.4.2 a set of fundamental questions was introduced based on the identified roles for contracts with respect to safety regulation: enforcement of design requirements, obtaining assurance evidence, and resolving shortfalls in assurance evidence. These questions were refined with respect to: architecture, behavioural arguments and evidence provision/suitability; the topics of Chapters 4 through 6. How much should the regulator know about these topics during the tender phase to be satisfied of a likely positive outcome, should the project go to contract?

Returning to the flight control system example, and assuming that the contract authority is using an open tender. The original aircraft manufacturer has no off-the-shelf solution available, and other contractors have expressed interest in developing a solution.

The remaining sections of this chapter describe how this tender may be prepared and evaluated, a preferred tenderer identified, and a contract established and executed for this option. Section 7.6 will consider the architectural topic. Section 7.7 will consider the behavioural arguments and evidence topics. Section 7.8 will then examine how issues arising as a result of the remaining uncertainty are identified and resolved post contract signature. The example will assume a single phase tender process, albeit the concepts can conceivably be applied to multi-phase tender processes also.

7.6 Obtaining Solution Architectural Certainty

Obtaining architectural certainty from the tender phases prior to entering into a contract is important as it enables insight into potential architectural shortfalls. It also forces supplier consideration of architectural suitability including fault avoidance and tolerance. This is important as there is evidence in industrial practice that this is sometimes overlooked (refer Chapter 10). A four step process is proposed, as follows:

1. Set measurable benchmarks for architectural suitability,
2. Inform architectural suitability using the tender process,
3. Evaluate architectural suitability during the tender evaluation, and
4. Provide architectural assurance during contract execution.

7.6.1 Setting Benchmarks for Architectural Suitability

The first step to obtaining architectural certainty is to set benchmarks for solution architectural suitability. The benchmarks should not be specifying solutions so they do not stifle novelty or limit flexibility; they should set measurable criteria against which different solutions can be evaluated. Benchmarks provide the acquirer a way of specifying what attributes the design has, and a way of comparing solutions.

A review of the literature reveals that there is limited published guidance on benchmarks for architectural suitability, particularly for systematic faults and failures. Some standards permit assurance levels to be reduced based on architecture, but this is not a measure of the architectural adequacy. Therefore, new approaches are required if architectures are to be effectively evaluated during tender evaluations. One such approach (refer Chapter 4) uses the concept of an Architectural Safety Assurance Level and Layered Fault Tolerance Requirements.

To set the benchmark for the supplier, clauses are required for the tender and contract SOR. The following is an example of a generic SOR clause to achieve this:

The [System Name] architecture and mechanisms for achieving fault avoidance and fault tolerance, against each type of credible systematic fault, shall meet the requirements for layers of fault avoidance and fault tolerance, where the number of layers is commensurate with the worst credible failure condition, as specified at {reference a Table in the SOR detailing the benchmark numbers of layers for each failure condition severity}

A specific instantiation of this clause for the Architectural Safety Assurance Level approach is presented below. Note that the top level safety objective clauses have also been included to provide context to the ASAL framework clauses, and were adapted from clauses existing in the Australian Defence Force Contracting Templates (Defence Materiel Organisation , 2012) and (Directorate General Technical Airworthiness, 2010).

Top-level Safety Goal

The [System Name] shall not cause an intolerable hazard to safety when operating in the intended roles, configurations and operating environments of the [Acquirer].

Criteria for Risk Treatment and Retention

The [System Name] shall meet the requirements of 14CFR25.1309[37]*, and all associated Advisory Circulars, Orders, and Notices.*

The risk of the [System Name] causing a hazard to safety when operating in the intended roles, configurations and operating environments of the [Acquirer] shall be:

- *tolerable to the [Acquirer] per a risk management framework agreed by the [Acquirer]; and*
- *explicitly documented and communicated to the [Acquirer].*

[37] (United States of America, 2012) Subpart F – Equipment §25.1309 Equipment, systems, and installations sets the acceptable risk criteria for civil transport category airplanes. Similar clauses exist for other classes of aircraft in Part 23, 27 and 29 for civil aircraft and in MIL-HDBK-516B or DEF STAN 00-970 for military types.

Architectural Safety Requirements

The [System Name] design shall employ the fail safe design criteria of AC25.1309[38] *to provide protection against both random and systematic classes of faults and failures, regardless of their origin.*

The [System Name] architecture and mechanisms for achieving fault protection and fault tolerance against systematic faults shall meet the Architectural Safety Assurance Level (ASAL) requirements defined in [Table 15].

The [System Name] shall meet the ASAL Architecturally Layered Fault Tolerance Requirements as defined in [Table 16]; or be shown to provide an equivalent level of fault tolerance by alternative means.[39]

7.6.2 Informing Architectural Suitability

To reduce architectural uncertainty before contract signature, the tender phase requires information about architecture. Since the information will be used by the acquirer to evaluate the suitability of the architecture against the benchmarks, it is useful to ensure the information directly addresses the benchmarks set out in Section 7.6.1.

One approach is to require the tenderer, through the tender SOW, to provide a Conceptual System and Software Architecture Suitability Document, or similar document. It would describe how the system's architecture and mechanisms for achieving fault avoidance and fault tolerance against systematic faults would meet the benchmarks. The intent is to provide a description of the architecture that the acquirer can evaluate against the benchmark, without forcing the supplier to completely design and implement the system before contract signature. For a largely mature design, the document can focus on what already exists, and whether or not it requires supplementation; for a developmental design it provides a framework for the supplier to cost the architectural elements of their system with improved accuracy. The following is an example of the generic Tender SOW clauses to achieve this:

[38] (Federal Aviation Administration, 1988) describes the acceptable means of compliance with 14CFR25.1309. Similar guidance exists for other classes of aircraft.

[39] An alternative means may be appropriate where the system architecture does not conform to the software, LRU and system level model used for expressing protection mechanisms against systematic faults in Table 16.

Total Layers of Defence. *The [Tenderer] shall prepare a [Conceptual System and Software Architecture Suitability Document] per TDRL [XX] to describe how the [System Name] architecture and mechanisms for achieving fault prevention and fault tolerance, against each type of credible systematic fault, is proposed to meet the {reference to SOR's requirements for number of layers of fault prevention and fault tolerance to systematic faults}.*

Adequate Constraints. *The [Tenderer] shall prepare a [Conceptual System and Software Architecture Suitability Document] per TDRL [XX] to describe how each constraint (i.e. absence assertion or detection and handling mechanism) is proposed to achieve the architecturally layered fault prevention and fault tolerance requirements as defined by the SOR {reference the SOR requirement}.*

A specific instantiation of these clauses for the Architectural Safety Assurance Level approach is as follows:

Informing Architectural Suitability

The [Tenderer] shall prepare a [Conceptual System and Software Architecture Suitability Document] to describe how the [System Name] architecture and mechanisms for achieving fault prevention and fault tolerance against systematic faults meets the Architectural Safety Assurance Level (ASAL) requirements defined in [Table 15].

The [Tenderer] shall prepare a [Conceptual System and Software Architecture Suitability Document] to describe how each constraint (i.e. absence assertion or detection and handling mechanism) is proposed to achieve the ASAL Architecturally Layered Fault Prevention and Fault Tolerance Requirements as defined in [Table 16]; or be shown to provide an equivalent level of fault prevention and fault tolerance by alternative means.

An example of a TDRL and Conceptual System and Software Architecture Suitability Document DID is provided at Appendix C.

For the flight control system example, let's assume that each of the proposed options provides a Conceptual System and Software Architecture Suitability Document, for which the proposed architecture are described as follows:

- Option A
    - Redundant digital flight control system consisting of triple redundant primary flight control computers, and dual redundant secondary flight control computers.

    - Primary and secondary flight control computers are architecturally spread between pairs of control surfaces in each axis to provide protection against mechanical control system elements failures or jamming.
    - Dual sensors including air data and inertial reference systems, attitude/heading reference systems and triplex actuators and actuator sensors.
    - Command/monitor architecture flight control computers, with fault prevention on the command channel and fault tolerance from the monitor channel, and flight control computer interactions.
    - Fault tolerance on input sensor data, control law outputs, and system state.
- Option B
    - Quad redundant digital flight control system incorporating two flight control computers with two independent channels per computer.
    - Computers and channels are architecturally spread between pairs of control surfaces in each axis to provide protection against mechanical control system elements failures or jamming.
    - Dual sensors including air data and inertial reference systems, attitude/heading reference systems and triplex actuators and actuator sensors.
    - Incorporation of software fault prevention and tolerance within each computer.
    - Fault tolerance on input sensor data, control law outputs, and system state.
- Option C
    - Quadruplex digital flight control computers incorporating a single channel per computer.
    - Computers are architecturally spread between pairs of control surfaces in each axis to provide protection against mechanical control system elements failures or jamming.
    - Incorporation of fault prevention and fault tolerance within each computer.
    - Fault tolerance on input sensor data and control law outputs.
- Option D
    - Quad redundant digital flight control system incorporating two flight control computers with two independent channels per computer.
    - Sensors include a single air data system, dual attitude/heading reference systems and dual actuators and actuator sensors.

  - Design is based upon a flight control system from a fixed wing military aircraft, and adapted for this application.
  - Fault tolerance on input sensor data and control law outputs.
- Option E
  - Simplex digital control system, single control panel, and single sensors including air data system, attitude and heading references, and actuator position sensors.
  - Flight tolerance as range and rate checks on control law outputs only.

Note that these architectural descriptions are deliberately brief. They are intended to be illustrative for the purposes of making a point about how contracting processes can be used to inform their suitability. In practice, the level of detail would need to be superior to the level of detail provided in Chapter 4, as Chapter 4 was deliberately brief.

7.6.3 Evaluating Architectural Suitability

The purpose of requesting this information is to permit evaluation of how the safety and software architecture requirements are priced in the tender response. The retrospective incorporation of constraints to treat systematic failure modes is rarely straightforward, particularly when architectural change is required. Therefore, it is in the acquirer's interests to establish that the contractor has determined an architecture based on the types of constraints required to meet safety objectives. While sub-system architectures may not be fully defined, the absence of this information will permit the acquirer to adjust the contractor's proposed costing based on the suitability and uncertainty of the tenderer's proposed architecture. This provides normalisation of tenderers' responses.

As can be seen from the differing architectures proposed by Options A through E, the complexity of each solution differs notably. Using the benchmarks set for the architecture, each option is evaluated. The evaluation results are as follows:

- Options A and B – Treatments to all general classes (i.e. omission, commission, early, late and value) of systematic fault use layers of fault avoidance and fault tolerance mechanisms. Architectures are likely to be suitable.
- Option C – Treatments to all general classes of systematic fault use layers of fault avoidance and fault tolerance mechanisms, with the exception of several sub-classes of omission and commission failures relating to system state anomalies. Architecture is potentially suitable with some enhanced fault tolerance. These issues are flagged for further consideration once evidence provision is evaluated.

- Option D – Treatments relating to value failures of the air data system sensor rely on fault avoidance via absence arguments only. The fidelity of the output range and rate checks does not adequately detect credible value failures resulting from the undetected sensor failures. There is limited software fault tolerance proposed for these failures. Therefore the architecture is deemed to contain weaknesses against these systematic faults and thus would require changes to adequately treat. Architecture is potentially unsuitable, and is flagged for further consideration once evidence provision is evaluated.
- Option E – Treatments relating to omission and value failures of sensors and flight control computers rely on fault avoidance from absence arguments only. The fidelity of the output range and rate checks does not adequately detect the aforementioned classes of sensor failures either. This is assessed to provide grossly inadequate defences against these classes of systematic failures. Architecture is deemed unsuitable, and option is eliminated from the selection.

7.6.4 Providing Architectural Assurance

Once the preferred tenderer has been identified, and any uncertainties regarding the architectural assurances are tolerable (assuming in this case that it will end up being either Options A, B, or C because of their architectural suitability), then it is possible to develop a contract between the supplier and acquirer.

Under the contract, the acquirer will need to maintain the benchmarks for product suitability by inclusion of SOR clauses similar to those defined in Section 7.6.1. Further the acquirer will require a way to establish if the final 'as-delivered' architecture meets the prescribed benchmarks. This can be achieved by requiring the contractor to deliver (via appropriate SOW contract clause) a System and Software Architectural Assurance Document, or similar. The document should describe how the system's architecture and mechanisms for achieving fault prevention and fault tolerance against systematic faults actually achieves the benchmarks. The following is an example of the generic Contract SOW clauses to achieve this:

Total Layers of Defence. *The [Contractor] shall prepare a [System and Software Architectural Assurance Document] per CDRL [XX] to describe how the [System Name] architecture and mechanisms for achieving fault prevention and fault tolerance, against each type of credible systematic fault, meets the {reference to SOR's requirements for the number of layers of fault prevention and fault tolerance to systematic faults}.*

Adequate Constraints. *The [Contractor] shall prepare a [System and Software Architectural Assurance Document] per CDRL [XX] to describe how each proposed constraint (i.e. absence assertion or detection and handling mechanism) achieves the architecturally layered fault prevention and fault tolerance requirements as defined by the SOR {reference the SOR requirement}.*

A specific instantiation of these clauses for the Architectural Safety Assurance Level approach is as follows:

Total Layers of Defence. *The [Contractor] shall prepare a [System and Software Architectural Assurance Document] per CDRL [XX] to describe how the [System Name] architecture and mechanisms for achieving fault prevention and fault tolerance against systematic faults achieves the Architectural Safety Assurance Level (ASAL) requirements defined in [Table 15].*

Adequate Constraints. *The [Contractor] shall prepare a [System and Software Architectural Assurance Document] per CDRL [XX] to describe how each proposed constraint (i.e. absence assertion or detection and handling mechanism) is proposed to meet the ASAL Architecturally Layered Fault Prevention and Fault Tolerance Requirements as defined in [Table 16]; or be shown to provide an equivalent level of fault prevention or tolerance by alternative means.*

The Contract Data Requirements List (CDRL) should require that various iterations of the document be delivered at relevant systems engineering milestones to permit the acquirer to monitor the evolution of the architecture. This monitoring is important because it allows the acquirer to measure the progression of the architecture throughout the lifecycle, and to respond if there are divergences to acquirer understanding and assumptions from the tender evaluation. An example of a CDRL is included at Appendix C.

Data Item Descriptions (DIDs) are required for all the deliverables listed in the CDRL (or TDRL mentioned in the previous section). DIDs are provided at Appendix C. DIDs are generally structural, and provide a heading framework to support provision of the relevant information. However the SOR clauses setting benchmarks for the product, and the SOW clauses requiring provision of the information, are the means by which the adequacy of the architecture is enforced. DID compliance only ensures that topical information is provided.

7.7 Obtaining Argument and Evidence Certainty

Obtaining argument and evidence certainty from the tender phases is important because it enables early insight into potential argument and evidence shortfalls. It also provides context specific agreement between acquirer and supplier on the measures of argument and evidence sufficiency for which there is no agreed universal approach. A four step process is proposed as follows:

1. Set benchmarks for argument and evidence suitability,
2. Proposal of argument and evidence using the tender process,
3. Evaluate argument and evidence suitability during the tender evaluation, and
4. Provide argument and evidence assurance during contract execution.

7.7.1 Setting Benchmarks for Arguments and Evidence

The first step for obtaining argument and evidence certainty is to set benchmarks for argument and evidence sufficiency. In keeping with the notion of a compromise between goal-based and prescriptive paradigms, and the notion of pre-constraining parts of the argument, the benchmarks should not identify specific techniques or methods for evidence generation. They should instead provide a coherent framework for how evidence will be related to safety properties, and provide a set of criteria for establishing when evidence generation is completed.

A review of the literature reveals that there is limited literature in the public domain that sets explicit benchmarks for measuring argument and evidence sufficiency (refer Chapters 2, 5 and 6). Therefore, new approaches are required.

For argument and claims, one approach has been described in Chapter 5. The approach uses concept of a Claims Safety Assurance Level (CSAL), and a set of generic arguments centred around the ‘attributes’ of lifecycle products of specified ‘constraint’ level requirements and applicable abstract level requirements, low level requirements, source code and executable object code. For evidence, one approach has been developed in Chapter 6. It introduces the concept of an Evidence Safety Assurance Level (ESAL) and ‘Tolerability of Limitations’. The remaining sub-sections discuss how these approaches can be incorporated into tenders and contracts.

7.7.2 Proposal of Argument and Evidence

To reduce uncertainty about the intended safety argument at the time of contract signature, the tender phase requires a mechanism to be informed of the argument. This implies that it is useful to know which claims are going to be applied to each architectural ‘constraint’.

One approach is to require the tenderer, through the tender SOW, to provide a Safety Assurance Plan, or similar. The document would describe which set of claims will be demonstrated for each ‘constraint’. This may be tabular or using argument notations such as those described by Section 2.4.4 (e.g. GSN). To ensure consistency in tenderer responses it is advantageous to align where possible the claims to the hierarchy of lifecycle products and associated attributes. A DID for the Safety Assurance Plan is included at Appendix C. The following is an example of a generic Tender SOW to achieve this:

The [Tenderer] shall prepare a [Safety Assurance Plan] per TDRL [XX] to propose the attributes/properties that will be assured, for each lifecycle product, for each constraint described in the [Conceptual System and Software Architecture Suitability Document].

A specific instantiation of these clauses for the Claims Safety Assurance Level approach is as follows:

Assurance of Constraints using Claims Assurance (CSAL)

The [Tenderer] shall prepare a [Safety Assurance Plan] to describe the Claims Safety Assurance Level (CSAL) proposed for each constraint described in the [Conceptual System and Software Architecture Suitability Document] as per [Table 20].

The tender phase also requires a mechanism to provide information on the likely scope of the body of evidence and its potential limitations. One approach would be to require the tenderer, through the tender SOW, to provide two things:

- a Development Plan to describe which methods and techniques are going to be applied across the development, and
- a Safety Assurance Plan to describe how any limitations in the evidence produced from the methods and techniques described in the development plan are tolerable with respect to relevance, trustworthiness and results.

Development Plans are routinely used. However the key contribution is a partner document (the Safety Assurance Plan) that presents the analysis and justification for the adequacy of the Development Plan, with respect to the tolerability of limitations in evidence concept. By requiring each tenderer to explicitly justify the adequacy of their development against defined criteria (e.g. the CSAL and ESAL framework), then suppliers are provided a consistent set of expectations for costing their development programs. The Safety Assurance Plan may be similar to documents such as the:

- System Safety Program Plan (SSPP) from MIL-STD-882,
- Plan for Software Aspects of Certification (PSAC) from RTCA/DO-178B, or
- Software Safety Plan from DEF STAN 00-55.

However the plan described in this chapter is focussed at demonstrating a specific set of outcomes with respect to arguments and evidence.

The Safety Assurance Plan is quite different from a Verification Plan. A Verification Plan will usually provide the description of activities used to demonstrate requirements satisfaction. The Safety Assurance Plan presents the analysis and justification for the adequacy of the Development Plan, by describing the claims and justifying the evidence proposed for each type of ‘constraint’. Conventional plans such as verification plans, test plans, etc. are still envisaged as companion documents to the Safety Assurance Plan and will form part of the body of evidence for the Safety Case. DIDs for the Development Plan and Safety Assurance Plan are provided at Appendix C.

The following is an example of a generic Tender SOW clause to achieve production of the Development Plan and Safety Assurance Plan:

Development Plan. *The [Tenderer] shall prepare a [Development Plan] per TDRL [XX] to describe the methods and techniques proposed to be used throughout the development lifecycle, including description of techniques or methods used prior to this development but for which evidence is relevant.*

Safety Assurance Plan. *The [Tenderer] shall prepare a [Safety Assurance Plan] per TDRL [XX] to describe how the evidence produced from the application of the [Tenderer] proposed methods and techniques is proposed to assure tolerability of limitations in evidence with respect to relevance, trustworthiness and results, for each attribute of each lifecycle product, for each constraint described in the [Conceptual System and Software Architecture Suitability Document].*

A specific instantiation of these clauses for the Evidence Safety Assurance Level and Claims Safety Assurance Level approach is as follows:

Assurance of Evidence (ESAL and Tolerability of Limitations)

<u>*Defining the Evidence*</u>

The [Tenderer] shall prepare a [Development Plan] to describe the methods and techniques proposed to be used throughout the software development lifecycle, including description of techniques or methods used prior to this development but for which evidence is relevant.

The [Tenderer] shall prepare a [Development Plan] to describe how all evidence, both new and existing, or produced from the application of [Tenderer] proposed methods and techniques will be documented, stored, and retrievable.

The [Tenderer] shall prepare a [Development Plan] to describe how CDRLs [refer list at Appendix C] will be produced per the schedule [X].

<u>*Assessing the Evidence*</u>

The [Tenderer] shall prepare a [Safety Assurance Plan] to describe how the evidence produced from the application of the [Tenderer] proposed methods and techniques is proposed to achieve the Evidence Safety Assurance Level (ESAL) requirements for tolerability of limitations as defined in [Table 24]; for each attribute of each lifecycle product [per Appendix B to this paper], at the CSAL [defined per Table 19] and as described in the [Conceptual System and Software Architecture Suitability Document] for each proposed constraint.

The [Tenderer] shall prepare a [Safety Assurance Plan] to describe the means, either via provision of evidence or via access provisions to tenderer facilities and data, for the [Acquirer] to inspect or review all evidence, both new and existing, from the application of [Tenderer] proposed methods and techniques for the purposes of certification evaluation by the [Acquirer].

It is also beneficial to evaluate the tenderer's understanding of implementing the plans and how they will demonstrate safety. Therefore, exemplar elements of the safety case should be sought. The following SOW clause elicits such examples. The tenderer is free to propose how the information is presented (tabular or using an argument notation such as GSN). Chapters 5 and 6 provide an example of how this may be done.

Exemplar Elements of the Software System Safety Case

The [Tenderer] shall prepare an [Exemplar Software System Safety Case] to show the implementation of the ASAL, CSAL and ESAL framework for at least one constraint in *each generalised category, type or class of constraint proposed. The [Tenderer] shall describe the set of categories, types or classes by which they have categorised the proposed constraints.*

For the flight control system example, assume that each of the proposed options provides a Development Plan and Safety Assurance Plan. Note that for the purposes of clarity this is an illustrative summary without the corresponding justification. The full content of the plans is described by the DIDs at Appendix C, and the examples from Chapters 5 and 6 illustrate how such arguments and evidence may be presented. Because this chapter is demonstrating how the contracting process can be used to down-select tenders from a safety assurance perspective, the focus of this example is to highlight the differences between the proposals, rather than the detail of what the proposals present.

- Options A and B are holistically quite similar in the range of information – evaluation is required to determine the specific evidence differences.
  - ARP4754 system safety program with software development assurance to RTCA/DO-178B Level A.
  - Constraints identified for each fault prevention and fault tolerance objective.
  - Constraint assurance proposed, attributes identified and template arguments provided for each attribute of each lifecycle product.
  - Draft arguments for relevance, trustworthiness and results for each attribute. Attributes traceable to RTCA/DO-178B objectives.
  - Evidence listed corresponding to evidence listed in template argument patterns for attributes (refer to Chapter 6 from an example of how evidence may be presented).
- Option C
  - Defence Standard 00-56 Iss 4 system safety program with software assurance to Defence Standard 00-55 Iss 2 SIL4, including the application of formal methods.
  - Constraints identified for each fault prevention and fault tolerance objective.
  - Constraint assurance proposed, attributes identified and template arguments provided for each generalised class of attribute.
  - Draft arguments for relevance, trustworthiness and results for each generalised attribute classes. Some repetition in arguments and evidence

    - traceability between high level requirements, abstract refined level requirements and low level requirements that requires close evaluation.
    - Evidence listed corresponding to evidence listed in template argument patterns for attributes.
- Option D
    - MIL-STD-882D system safety program, with new software developed to RTCA/DO-178B Level A, and reused software developed to MIL-STD-498.
    - Attributes, attribute arguments for relevance, trustworthiness, and results of evidence provided for all newly developed software.
    - Substantial reuse of software is proposed, with arguments relating to relevance and trustworthiness of evidence proposed to be satisfied by service history.
- Option E
    - MIL-STD-882D safety program, with software developed to MIL-STD-498.
    - Limitations in arguments being provided against any attributes. Relevance of evidence is not argued. Trustworthiness of evidence is by peer review.
    - Limitations in evidence against notable attribute categories including traceability and verification coverage.

7.7.3 Evaluation of Argument and Evidence

The purpose of the tender requesting this information is for evaluation of how evidence requirements are priced in the tender response. The retrospective supplementation of evidence is rarely straightforward, particularly when it results in a change to requirements, design or code. Therefore, it is important to establish if the contractor has proposed sufficient evidence. Insight into the following is required:

- the techniques and methods proposed,
- what evidence will be produced?,
- how this evidence will combine?, and
- what limitations in the evidence might be intolerable?;

This will permit the acquirer to adjust the contractor's proposed costing based on the suitability and uncertainty of the tender's proposed evidence set. For example, if there is an intolerable limitation in evidence, the acquirer could estimate the cost to resolve the limitation, and increase the tenderers cost proposal accordingly. This provides normalisation of tenderers responses.

Considering the examples proposed in the previous section, it is evident that the evidence set proposed by Options A through E varies substantially for each proposal. Using the benchmarks set for argument and evidence, each option is evaluated. The evaluation results are summarised as follows:

- Option A – There is a limitation with the extensiveness of normal and robustness verification proposed against low level requirements relating to time-dependent properties, including synchronisation, of the flight control laws with respect to fault tolerance for jitter (early and late) related effects and failure modes on sensor inputs. The tenderer is requested to clarify their proposal.
- Option B – There is a limitation in the extensiveness of analytic and empirical verification of behaviours relating to fault tolerance of value failures of air data system and attitude/heading reference system sensors. This is due to fault tolerance mechanisms being used into device drivers which can only be verified in the Systems Integration Laboratory but for which there is no means with the current toolset to inject these fault conditions for the purposes of verification. This limitation is flagged for clarification with the tenderer.
- Option C – Limitations in evidence for requirements and verification traceability exist between abstract refined, low level requirements, and source code. The contractor states that they will not resolve such a limitation, as this evidence is not required by their established processes. The limitations are assessed to be intolerable due to the role of traceability in understanding behaviours of a product.
- Option D – Limitations in evidence for reused software are substantial with respect to low level requirements, low level requirements verification, and coverage of implementation from requirements based verification. These limitations are assessed to be intolerable.
- Option E – Already eliminated based on architectural evaluation.

Options A and B require further clarification with the Tenderers, and this will be sought. Options C and D are eliminated from the tender evaluation due to intolerable evidence limitations, and Option E was already eliminated based on architectural shortfalls. Clarification with Options A and B reveals the following additional information for the evaluation:

- Option B – the limitation remains as the tenderer claims that low level verification undertaken prior to integration verification will provide sufficient evidence in this regard. Therefore verification of these requirements on the target computer with credible fault conditions is via inference only. These limitations are assessed to be intolerable. Option B is eliminated from consideration.

- Option A – the extensiveness of normal and robustness verification has been adequately clarified and is acceptable.

Therefore, Option A is selected as the preferred Tenderer, and negotiations are commenced to progress to contract signature. By coincidence, Option A corresponds to the examples used in Chapter 4, 5 and 6.

Note that in reality there are many other selection criteria for a product, and so it is common for capability, force integration, and political factors amongst others to affect selection. These other factors may sometimes require compromise on the ideal safety solution. This doesn't invalidate the process proposed in this thesis, and the process in this thesis enables the acquirer to be informed about the safety assurance aspects such that it is possible to make informed trade-offs between safety assurance and other selection criteria. For example, it may be possible to choose one of the other options, and make decisions regarding risk treatment or retention, because other benefits outweigh the impact of its limitations. This may be the strategy chosen for rapid acquisitions with operational imperatives.

7.7.4 Providing Argument and Evidence Assurance

Once the preferred tenderer has been identified (Option A); and uncertainties regarding the claims and evidence assurances are tolerable, then a contract can be written.

Under the contract, the acquirer will require a means to establish if the final 'as-delivered' claims and evidence meets the prescribed benchmarks. This can be achieved by requiring the contractor to deliver (via appropriate SOW contract clause) a Safety Assurance Summary Document. The document would describe how the assurance of the 'attributes' of software lifecycle products actually achieves the benchmarks established during tender processes. The following is an example of the generic Contract SOW clauses to achieve this:

Achievement of Claims and Attributes of Software Lifecycle Products

The [Contractor] shall prepare a [Safety Assurance Summary] per CDRL [XX] to describe the attributes that have been assured, for each software lifecycle product, for each constraint described in the [System and Software Architecture Document].

<u>Assessing the Evidence</u>

The [Contractor] shall prepare a [Safety Assurance Summary] per CDRL [XX] to describe how the evidence produced from the application of the [Contractor] proposed methods and techniques has assured the tolerability of limitations in evidence with respect to relevance, trustworthiness and results, for each attribute of each software lifecycle product, for each constraint described in the [System and Software Architecture Document].

A specific instantiation of these clauses for the Architectural Safety Assurance Level approach is as follows:

Assurance of Constraints using Claims Assurance (CSAL)

<u>Proposal of CSAL</u>

The [Contractor] shall prepare a [Safety Assurance Plan] per CDRL [XX] to describe the Claims Safety Assurance Level (CSAL) proposed at commencement of development for each constraint described in the [(Preliminary) System and Software Architecture Document] as per [Table 20].

<u>Achievement of CSAL</u>

The [Contractor] shall prepare a [Safety Assurance Summary] per CDRL [XX] to describe the Claims Safety Assurance Level (CSAL) established for each constraint described in the [System and Software Architecture Document] as per [Table 20].

Provision of Evidence (ESAL and Tolerability of Limitations)

<u>Defining the Evidence</u>

The [Contractor] shall prepare a [Development Plan] per CDRL [XX] to describe the methods and techniques proposed to be used throughout the development lifecycle, including description of techniques or methods used prior to this development but for which evidence is relevant.

The [Contractor] shall prepare a [Development Plan] to describe how all evidence, both new and existing, or produced from the application of [Contractor] proposed methods and techniques will be documented, stored, and retrievable.

The [Contractor] shall prepare a [Development Plan] to describe how CDRLs [XXXX] will be produced per the schedule [refer Appendix C].

Assessing the Evidence

The [Contractor] shall prepare a [Safety Assurance Summary] to describe how the evidence produced from the application of the [Contractor] proposed methods and techniques achieves the Evidence Safety Assurance Level (ESAL) requirements for tolerability of limitations as defined in [Table 24]; for each attribute of each lifecycle product [per Appendix B], at the CSAL [Table 19] and as described in the [System and Software Architectural Assurance Document] for each constraint.

The [Contractor] shall prepare a [Configuration Index] to describe the configuration of the [System] and [Software] relevant to the evidence, claims and architecture described by the [Safety Assurance Summary].

Safety Case

The [Contractor] shall prepare a [Safety Case] per CDRL XX to describe how the safety objectives, and safety assurance requirements of the contract SOR have been achieved for [System] and [Software], and to provide the argument and evidence to show the satisfaction of ASAL/CSAL/ESAL criteria for each constraint.

Examples of the argument and evidence to show the satisfaction of ASAL/CSAL/ESAL criteria for each constraint have been presented in Chapters 4, 5 and 6. It is envisaged that this information forms the content of the safety assurance summary and that the safety case is a summary level argument about what has been achieved (or not achieved) and demonstrated (or not demonstrated). The safety case must also identify the risk of what hasn't been achieved and demonstrated, as described by Chapter 8.

Certification Evaluation

The [Contractor] shall prepare a [Safety Assurance Plan] to describe the means, either via provision of evidence or via access provisions to tenderer facilities and data, for the [Acquirer] to inspect or review all evidence, both new and existing, from the application of [Contractor] proposed methods and techniques for the purposes of certification evaluation by the [Acquirer].

The [Contractor] shall provide evidence or access to evidence as described in the [Acquirer] approved [Contractor]'s [Safety Assurance Plan] for the purposes of certification evaluation by the [Acquirer].

The [Contractor] shall prepare the following deliverables and deliver them in accordance with the document delivery schedule defined in the CDRL (refer to Appendix C):

- *[System and Software Architectural Assurance Document] [CDRL XX]*
- *[Development Plan] [CDRL XX]*
- *[Safety Assurance Plan] [CDRL XX]*
- *[Safety Assurance Summary] [CDRL XX]*
- *[Safety Case] [CDRL XX]*

The following DIDs (non-exhaustive list) may also be in the CDRL depending on intellectual property rights. These are optional with respect to this framework, and the acquirer's contracting policy and certification guidance documents should be sought for specific requirements regarding deliverables.

- System Development Specification
- Sub-System Design Document
- Software Requirements Specification
- Software Design Document
- Source Code Repository
- Executable Code Repository
- Toolset Repository
- Configuration Index / Version Description
- System and Software Lifecycle Data Repository
- Verification Plan / Software Verification Plan
- Verification Results / Software Verification Results
- System Verification Results

7.8 Resolving Issues after Contract Signature

Despite best intentions, whenever there is uncertainty there is potential for it to lead to an undesirable outcome as development progresses. The previous sections have largely been focussed on bounding the uncertainty in areas that affect safety. However, once a contract is commenced, if issues do arise with respect to architecture, claims or evidence, then it is important to agree the approach for resolution of these issues.

Considering the on-going example of Option A, and let's assume that during preliminary design review several issues are identified as follows:

- Issue 1 – Proposed treatments to value failures of air data system airspeed data are identified to be inadequate under simulated high alpha conditions. A revised treatment is proposed requiring an adaptation to flight control law transition criteria to provide an improved fault tolerance against this fault.
- Issue 2 – Verification and validation of the accuracy of the requirements relating to discrete implementation of the legacy analogue control laws is identified to contain shortfalls relating to the reuse of modelling. Additional modelling is viewed as required by the acquirer.

There are two main options for providing contract scope for the work to resolve unforeseen issues that arise: either within the original contract, or through a contract change. Both are discussed in the following sub-sections.

7.8.1 Resolution within Contract Scope

Resolution within the contract scope is dependent on the supplier acknowledging the requirement to resolve the issue. However, when profit margins and schedule are at risk, suppliers may argue work is out of scope. Consider the two issues:

- Issue 1: This treatment is deemed in-scope of contract because it was a contractor oversight during the conceptual design proposal. The contractor accepts this and evidence is provided commensurate with previously identified attributes, lifecycle products and constraints.
- Issue 2: Acquirer and supplier enter into contractual dispute regarding the provision of additional evidence modelling the discrete implementation, because the supplier claims their limitations in the modelling are tolerable.

One way to address Issue 2 is to make absolutely explicit this requirement for limitations to be resolved to the satisfaction of the acquirer through a statement of work line item. This line item can then be priced and suppliers will be empowered to resolve such issues. An example of how this might be achieved is as follows:

Intolerable Limitations in Evidence, Claims or Architecture

Where the [Acquirer]'s certification evaluation establishes that the [Contractor] has not achieved the requirements of the {reference applicable SOR and SOW clauses relevant to architecture, argument and evidence}, or there are shortfalls in the 'Tolerability of Limitations' of evidence versus the criteria specified by this contract, then the [Contractor] shall undertake one or more of the following remediation actions to resolve the shortfalls to the satisfaction of the certification authority:

- *engineering change to architectural constraints,*
- *engineering change to implementation of architectural constraints, or*
- *additional analysis, verification and validation by further or supplementary application of methods or techniques.*

The [Contractor] shall amend all relevant deliverables per the CDRL to incorporate the engineering changes and additional evidence.

Note to Contractors

The above clause provides the means for the certification authority to address shortfalls against architecture, argument and evidence expectations. It is intended to require the contractor to accurately cost the full value of achieving a certifiable product, and not simply to provide the product as is based on the specification or scope of work defined.

While this clause may be interpreted to result in unbounded programmatic risk for the contractor, the intent is to focus both acquirer and contractor efforts at establishing unambiguous consensus during the tender process and contract negotiations. The contractor should not sign the contract if they believe there remains substantial uncertainty regarding the provision of evidence against the framework, and instead request further clarification during contract negotiations.

The aim is to ensure that the tender phases and contract negotiation phases have systematically identified, disclosed and evaluated the intended body of evidence and that all intolerable shortfalls have been included within the contract. Thus the example clauses would only come into effect if an issue remains, and this would be less likely and less serious because the evidence planning was systematic in the first place.

The drawback is that suppliers may interpret this as a risky statement of work line item and cost it commensurately. Some may even push back and ask for it to be removed. However there are benefits to the behaviour this generates for tender evaluation. If the acquirer evaluates the cost attribution against this line item from each tenderer, and there are notable differences in the costing (or absences because it isn't priced), then the acquirer can use this to establish the tenderer's confidence in their own estimates. This is a very useful during tender evaluation. Even if the clause is removed during contract negotiations, its inclusion during the tender process is revealing about supplier confidence in their proposals and costing.

7.8.2 Resolution Outside of Contract Scope

Resolution of shortfalls outside the contract scope is easy from the perspective of defining the scope of work; as usually the analysis to determine that the architectural changes, design changes or evidence supplementation will be clear from the analysis done to demonstrate it is outside the original contract. If there is contingency funding to fund the contract change, then it will also be relatively straightforward for the acquirer.

However, if contingency funding is not available then additional funding must be sought from Government. Most Governments responsible for funding military aviation system acquisitions are not sympathetic to issues that emerge which were not forecast within original funding, allocated as contingencies, or articulated as program risks.

For the purposes of this thesis, the approach described at Section 7.8.1 is preferred at the tender phase, so that the likelihood of additional out of scope work is well understood during the tender phase, and minimised in the contract phase.

7.9 Contract Execution

The contract execution phase of the lifecycle is where the contractor develops an architecture and body of evidence in-line with the ASAL/CSAL/ESAL benchmarks.

Focus will be on progressively establishing achievement of the objectives of the ASAL/CSAL/ESAL framework over the lifecycle. The recommended approach is through on-going visibility through a series of systems engineering reviews, and through progressive delivery of evidence via drafts at these reviews. For example the Design Reviews (Conceptual, Preliminary, Critical at System and Sub-system levels) defined by (US DoD, 1995), and (United States Air Force Space Command, 2009) provides an example of how this might be achieved.

The key goals of the certification authority during the contract execution will be visibility of potential shortfalls against the ASAL/CSAL/ESAL framework such that these can be addressed in a cost effective manner for the contractor under the terms of the contract. The following types of shortfalls should be monitored for:

- evidence shortfalls that inform product suitability against ASAL benchmarks,
- product shortfalls against ASAL benchmarks,
- shortfalls against CSAL benchmarks, and
- evidence shortfalls against ESAL benchmarks.

A goal of the contractor during the contract execution will be to achieve contract milestones within costs and schedule constraints, while meeting the requirements of the contract SOR and SOW. Timely visibility of shortfalls will be essential to keeping the project within cost and schedule constraints. It is recommended that the contractor employ specialists that understand fault tolerance and the principles on which the ASAL/CSAL/ESAL framework is based, in order to minimise potential of ASAL/CSAL/ESAL framework ignorance based rework. It is also recommended that these specialists have a direct line to the project manager in order to ensure technical assurance aspects of the project inform project management and resourcing decisions.

7.10 Summary

This chapter has examined factors affecting the provision of safety assurance evidence and safety regulation for military aviation contracts, including integration of the standard with the contract lifecycle, enforcement of design requirements, obtaining of assurance evidence and resolution of shortfalls in product and evidence.

This chapter has examined how the proposed ASAL/CSAL/ESAL assurance framework might be contracted for in the acquisition or modification of military aviation systems. Examples of Tender SOR, Tender SOW, Contract SOR and Contract SOW clauses to implement the ASAL/CSAL/ESAL framework have been provided and the rationale behind these explained. DIDs have been provided at Appendix C.

The approach chosen to implement the ASAL/CSAL/ESAL framework is based upon establishing the minimum necessary understanding of architectural fault tolerance and assurance during the tender/proposal phase, to enable effective tender/proposal evaluation, and to support bound-able contract negotiations and contract signature. Furthermore, guidance is provided on contracting for architectural, claims and evidence assurance across the tender/contract lifecycle, including project definition and approval, tender preparation, tender responses, tender evaluation, contract preparation, contract negotiation and contract execution.

The impact of uncertainty at the time of contract signature has been examined with respect to the potential for a successful contractual outcome. Approaches have been proposed for obtaining assurances and bounding uncertainty by pre-contract and throughout the contract. An example was used to illustrate the benefit in the approach.

The proposed framework ensures that product design requirements pertaining to safety can be communicated, established and agreed through the contracting process. It achieves this by setting SOR and SOW requirements for both the tender and contract phases. The framework also make the requirements for access and delivery of assurance evidence explicit by setting benchmarks, but without constraining the design solution unnecessarily. The framework also provides a means to address both new and legacy developments through using the tender process (and contract process) as an uncertainty reduction activity, thus providing flexible accommodation across differing acquisition paradigms. Finally, the framework provides arrangements for resolving shortfalls in product and evidence are explicit in the contract and meet both acquirer and supplier expectations.

At the start of the chapter, the question was raised as to how to seek the right information and effectively evaluate it with respect to safety for military aviation software systems. This chapter has provided a way to improve this activity, and address the challenges it creates. Evaluation of feasibility and effectiveness is presented in Chapter 10.

8 Relating Assurance to Risk

Chapters 4, 5 and 6 have presented a framework for reasoning about the suitability of product behaviours and the evidence used to provide knowledge of them. The knowledge of product behaviours; layers of defences; individual defences; constraints; lifecycle products; attributes of lifecycle products; and the relevance, trustworthiness and results of evidence all provide confidence in the knowledge of risks. The ASAL, CSAL and ESAL frameworks (of Chapters 4, 5 and 6) have defined properties of these aspects to measure the knowledge of product behaviours and knowledge.

In practice, however, there will often be issues with development. The reality of project cost and schedule constraints means that authorities may be faced with shortfalls. There may be a product shortfall that is known to affect a defence, or there are evidence shortfalls which lead to uncertainty. Both have an impact on risk.

In Chapter 3, a model containing principles and usability guidelines was established. This chapter describes how the knowledge gained from the frameworks of Chapters 4, 5 and 6 can be used to inform knowledge of risks, and decisions regarding the treatment or retention of risks duly informed. Hence, this chapter focusses on addressing the principles and usability criteria shown in bold italics within Figure 72.

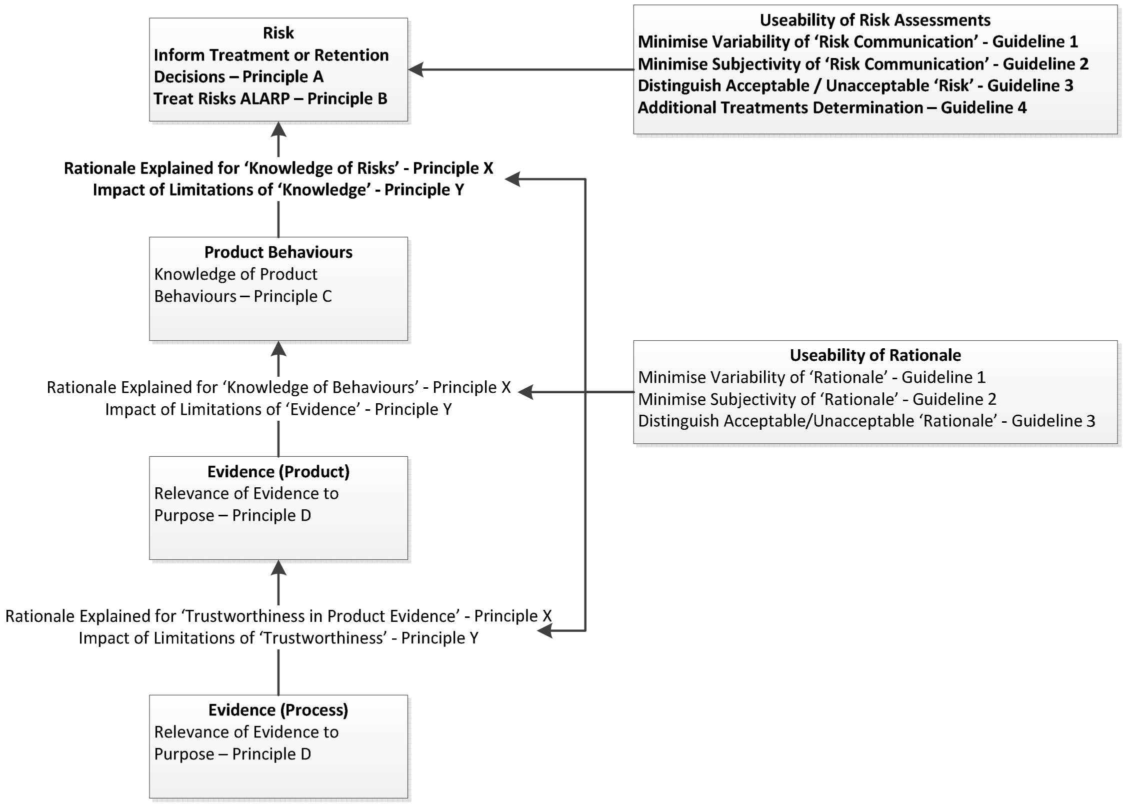

Risk
Inform Treatment or Retention Decisions – Principle A
Treat Risks ALARP – Principle B
Useability of Risk Assessments
Minimise Variability of 'Risk Communication' - Guideline 1
Minimise Subjectivity of 'Risk Communication' - Guideline 2
Distinguish Acceptable / Unacceptable 'Risk' - Guideline 3
Additional Treatments Determination – Guideline 4
Rationale Explained for 'Knowledge of Risks' - Principle X
Impact of Limitations of 'Knowledge' - Principle Y
Product Behaviours
Knowledge of Product Behaviours – Principle C
Useability of Rationale
Minimise Variability of 'Rationale' - Guideline 1
Minimise Subjectivity of 'Rationale' - Guideline 2
Distinguish Acceptable/Unacceptable 'Rationale' - Guideline 3
Rationale Explained for 'Knowledge of Behaviours' - Principle X
Impact of Limitations of 'Evidence' - Principle Y
Evidence (Product)
Relevance of Evidence to Purpose – Principle D
Rationale Explained for 'Trustworthiness in Product Evidence' - Principle X
Impact of Limitations of 'Trustworthiness' - Principle Y
Evidence (Process)
Relevance of Evidence to Purpose – Principle D

Figure 72: Implementing Key Principles of Safety Assurance with Respect to Risks

8.1 What is Risk?

Section 1.2 defined risk as the combination of the likelihood of harm and the severity of that harm (Ministry of Defence, 2007). In the context of aircraft systems in this thesis, harm would normally be associated with an aircraft accident or incident. Based on this definition common practice has been to define risk matrixes in terms of these two elements of risk, as shown in Table 30. Note that Table 30 shows a composite of risk matrixes (and risk language) from several standards. The displaced colour segments emphasise differences between the constituent risk matrixes, highlighting that the definitions may vary by standard or be contextual to the system being developed.

Severity **Probability / Frequency**	**Catastrophic**	**Critical**	**Marginal**	**Negligible**
Frequent	High / Unacceptable / Intolerable	High / Unacceptable / Intolerable	Serious/ Undesirable / Intolerable	Medium / Acceptable with Review / Undesirable
Probable	High / Unacceptable / Intolerable	High / Unacceptable / Intolerable	Serious / Undesirable / Undesirable	Medium / Acceptable with Review / Tolerable
Occasional	High / Unacceptable / Intolerable	Serious / Undesirable / Undesirable	Medium/ Undesirable / Tolerable with Review	Low / Acceptable / Tolerable with Review
Remote	Serious / Undesirable / Undesirable	Medium / Undesirable / Tolerable with Review	Medium / Acceptable with Review / Tolerable with Review	Low / Acceptable / Tolerable
Improbable	Medium / Acceptable with Review / Tolerable with Review	Medium / Acceptable with Review / Tolerable with Review	Medium / Acceptable with Review / Tolerable	Low / Acceptable / Tolerable
Eliminated / Incredible	Eliminated / Tolerable with Review	Eliminated / Tolerable	Eliminated / Tolerable	Eliminated / Tolerable

Table 30: Risk Matrix Composite from MIL-STD-882C/E and DEF STAN 00-56 Iss 2

While the aforementioned definition of risk is consistent with the widespread usage of risk definitions in safety and risk management standards, there are drawbacks to this definition.

8.1.1 Difficulties Estimating Likelihood

A notable difficulty with the definition of risk is in relation to how the likelihood of harm is estimated (Rae, et al., 2012). In traditional safety assessments, this is by assuming the likelihood is characterised by probability distribution functions or frequency measurements, and thus assuming that likelihood equals probability/frequency of failure. This had led to traditional safety methodologies using quantitative analysis of probabilities of failures to establish if probabilistic targets have been met (e.g. probabilities calculated during fault tree analysis). However, there is increasing recognition that while the quantitative approach is supported by reasoning based on logic, there is very limited empirical evidence that the quantitative estimates of probability are reliable when used for assessing risk for non-trivial systems (Rae, et al., 2012). If the probabilistic approach to risk evaluation were valid, then there should be evidence of the following:

- random processes (i.e. those characterised by probability distributions) contributing to accidents – in having reviewed numerous accident reports from (Australian Transport Safety Bureau, 2012) and (National Transportation Safety Board, 2013), the author hasn't found widespread evidence of random processes being major contributors to accidents;
- probabilities estimated during safety analysis being achieved in practice – analysis of predicted manufacturers data versus in-service data collection from aircraft operations, and evidence of development, review or approval of design and maintenance treatments to such shortfalls (Air Lift Systems Program Office, 2011), suggests that the estimated probabilities are often not achieved; and
- risk assessments based on probabilities being complementary to the risk treatment or retention decision process – in risk assessments for shortfalls of aircraft systems, improving the reliability of the failed item doesn't often equate to safety improvement, as it doesn't always improve the system's resilience against the source of the problem (Air Lift Systems Program Office, 2011).

In addition to (Rae, et al., 2012)'s evidence, this suggests evidence of the probabilistic approach being valid is limited.

8.1.2 Qualitative Risk Assessment

Some developers try to avoid the problems of quantitative probabilities by presenting qualitative assessments of likelihood instead. Unfortunately many of them fall into one of more of the following traps:

- presenting a qualitative indication of event frequency without justification;
- a quantitative combination of contributing qualitative factors; and

- presenting a qualitative assessment that summarises many factors that would reduce the likelihood, but with the following drawbacks:
    - it is difficult to distinguish likelihoods more remote than occasional,
    - it is difficult to determine what is overlooked in the assessment, and
    - arguments tend to focus on the remoteness of initiating conditions and operational mitigations, rather than on the effects on the behaviours of the system and the operator/s.

While the qualitative assessments of risk are often more compelling than the quantitative assessments based on probabilities, there isn't an established means of consistently qualifying the risk estimate assessment, and thus prompting appropriate decisions on additional risk treatment or retention.

8.1.3 Likelihood versus Probability

As stated above, likelihood is often equated to probability of failure in definitions of risk. While there are some similarities between what the words mean, there are differences between these terms that may be suggestive of the problems outlined above.

In the English language (refer (Oxford University Press, 2010) and (The Macquarie Library, 2002)) they are basically synonymous, often defined in terms of each other. Sometimes probability is associated with quantitative values, whereas likelihood is qualitative, but this distinction is not universally agreed.

In general mathematics they are identical - Likelihood(parameter) = P (event | parameter) (Azzalini, 1996). However, in the statistical branch of mathematics they mean different things. For probability, the probability distribution function (PDF) is characterised based on observed or estimated data (Devore, 2011). For example, if a large sample of historical data has been collected, then using this data it is possible to find the probability of an individual event within the PDF approximated by the data sample. The PDF models the frequency of events, not the reasons for them.

For likelihood, the specific outcome is anticipated but the PDF parameters are unknown (Azzalini, 1996). The type of PDF may be known in some specific cases where the underlying mechanisms are well understood, but usually the parameters may not be known. Likelihood is equal to the probability of the observed data given some unknown parameter value (Azzalini, 1996). Likelihood can be used to estimate the PDF and its parameters and to quantify uncertainty in those estimates. In essence likelihood implies that the underlying factors need to be understood before the probability of the PDF can be estimated, and based on the level of knowledge of those factors there will be a 'proportionate' amount of uncertainty. The definition of likelihood suggests that the knowledge required to inform a likelihood assessment differs from that required to establish a PDF based on estimated or observed data.

8.1.4 Alternatives to Probabilities in Risk Matrices

The most widespread recognition of difficulties with probabilities is evident from the way safety assurance standards have dealt with software by equating software levels or integrity levels with severity (refer to Section 2.4), albeit by adopting this approach the assessment of risk is no longer explicit. MIL-STD-882C/E proposes that risk can be assessed by replacing probability with the degree of control the software has in relation to the hazard or accident. The degree of control is categorised and a risk matrix established, as shown in Table 31. The assumption then is that the level of risk is equated to a level of rigour to treat the risk, and not for example that an autonomous piece of software is reduced to no safety impact.

Severity **Control Category**	**Catastrophic**	**Critical**	**Marginal**	**Negligible**
I – Autonomous	High	High	Medium	Low
II – Semi-Autonomous	High	Serious	Medium	Low
III – Redundant Fault Tolerant	Serious	Medium	Low	Low
IV – Influential	Medium	Low	Low	Low
V – No Safety Impact	Low	Low	Low	Low

Table 31: Software Risk Matrix Composite from MIL-STD-882C/E

The control category approach assumes that the degree of control the software has over critical functions and information is central amongst the underlying factors contributing to likelihood of the hazard. This is true in part; but the degree of control is not the only factor. So while the control category approach offers an improvement over probabilities, it is too narrow minded with respect to the complete set of factors that would characterise the likelihood of the hazard or accident. It also doesn't provide a means of treating risk via changing the parameters of the risk matrix. Treatment was by level of rigour, which doesn't really measure the likelihood of a fault or event and the associated consequences being realised (implicitly it assumes the level of rigour has been successfully achieved, which we've shown is rarely the case). This thesis proposes that both product and evidence knowledge factors are necessary to inform likelihood.

8.2 Redefining Risk – a Strength of Defences Paradigm

Chapters 4, 5 and 6 of this thesis have established a range of factors that inform knowledge of mechanisms which could contribute to risks. Based on these factors, risk of an accident is a function of:

- the accident consequence severity (i.e. how bad the accident could be in terms of fatalities, injuries, material loss, economic loss, etc.), and
- the likelihood of any event or fault propagating through the causal chain to an accident: which is a function of:
    - the strength of defences (refer Chapter 4) in the causal chain between events, initiation of faults and the accident; and
    - the knowledge established from evidence (refer Chapters 5 and 6) of the behaviour of the defences in the implementation.

The stronger the defences are, the better the defences are at blocking the causal chain to the accident, and the better our knowledge of the behaviours of the defences, then the lesser the risk of the accident. The goal is to seek knowledge of the defences in the causal chain, and to seek knowledge of the behaviours of these defences from evidence under applicable conditions to establish the strength of defences in a given causal chain or propagation path. Concurrently, it is important to characterise or measure the uncertainty that remains in order to characterise the likelihood that the underlying knowledge is valid.

The following sub-sections present the rationale using GSN for how these factors relate to evaluating confidence in the knowledge of risks, using the factors identified from Chapters 4, 5, and 6.

8.2.1 Confidence in the Knowledge of Risks

Figure 73 proposed that confidence in the knowledge of risks is characterised by the extent to which the knowledge of risks outweighs uncertainty. Knowledge and uncertainty of risks can only be compared if they can be characterised by measurement, and thus it is necessary to establish how knowledge and uncertainty can be measured.

The goals G_Knowledge_Risks and G_Uncertainty_Risks from Figure 73 are developed in Sections 8.2.2 and 8.2.3 respectively. C_Knowledge_V_Uncertainty is examined in Section 8.2.3.

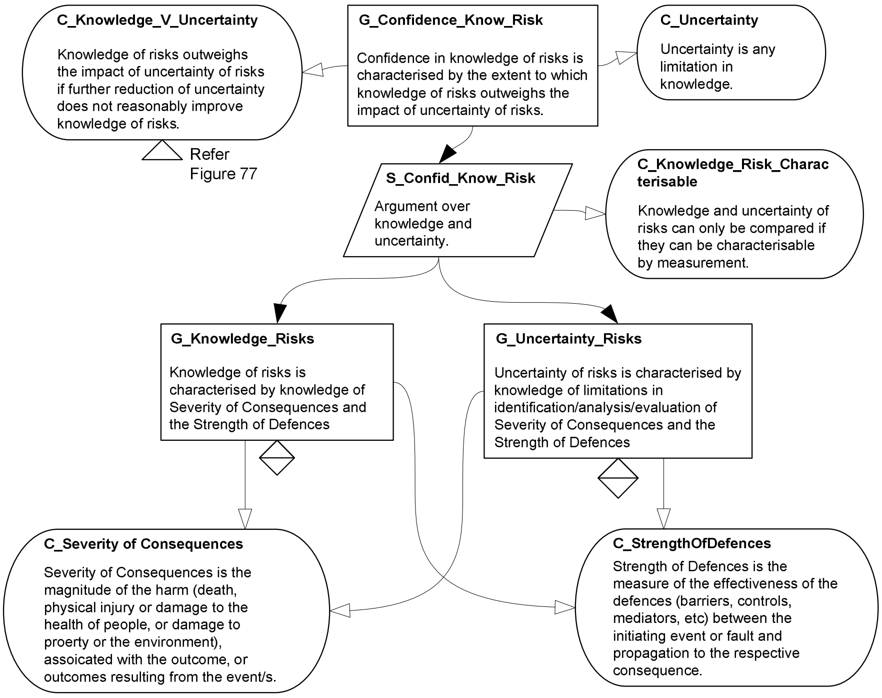

C_Knowledge_V_Uncertainty
Knowledge of risks outweighs the impact of uncertainty of risks if further reduction of uncertainty does not reasonably improve knowledge of risks.
Refer Figure 77
G_Confidence_Know_Risk
Confidence in knowledge of risks is characterised by the extent to which knowledge of risks outweighs the impact of uncertainty of risks.
C_Uncertainty
Uncertainty is any limitation in knowledge.
S_Confid_Know_Risk
Argument over knowledge and uncertainty.
C_Knowledge_Risk_Characterisable
Knowledge and uncertainty of risks can only be compared if they can be characterisable by measurement.
G_Knowledge_Risks
Knowledge of risks is characterised by knowledge of Severity of Consequences and the Strength of Defences
G_Uncertainty_Risks
Uncertainty of risks is characterised by knowledge of limitations in identification/analysis/evaluation of Severity of Consequences and the Strength of Defences
C_Severity of Consequences
Severity of Consequences is the magnitude of the harm (death, physical injury or damage to the health of people, or damage to proerty or the environment), assoicated with the outcome, or outcomes resulting from the event/s.
C_StrengthOfDefences
Strength of Defences is the measure of the effectiveness of the defences (barriers, controls, mediators, etc) between the initiating event or fault and propagation to the respective consequence.

Figure 73: Confidence in the Knowledge of Risks

8.2.2 Knowledge of Risks

Knowledge of risks is a function of the knowledge of the severity of credible consequences and of the strength of defences in the causal chain between the event or fault initiation and the realisation of the consequence. Figure 74 presents an argument over both of these factors.

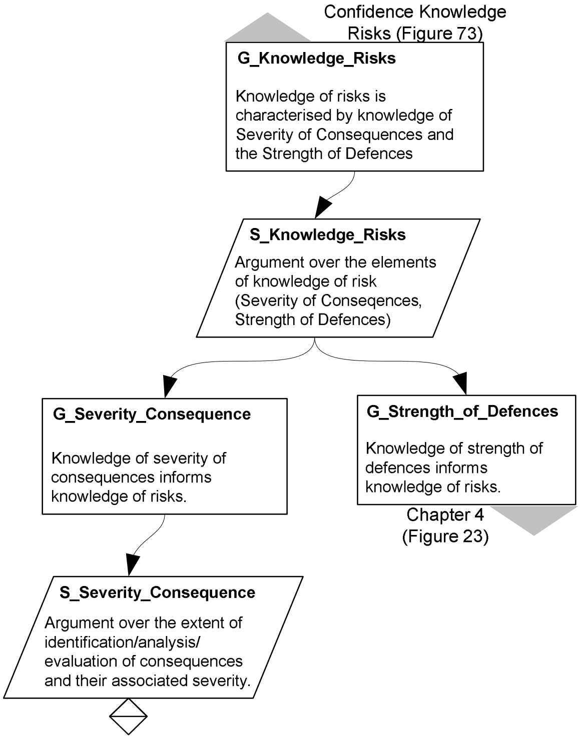

Confidence Knowledge Risks (Figure 73)
G_Knowledge_Risks
Knowledge of risks is characterised by knowledge of Severity of Consequences and the Strength of Defences
S_Knowledge_Risks
Argument over the elements of knowledge of risk (Severity of Conseqences, Strength of Defences)
G_Severity_Consequence
Knowledge of severity of consequences informs knowledge of risks.
G_Strength_of_Defences
Knowledge of strength of defences informs knowledge of risks.
Chapter 4 (Figure 23)
S_Severity_Consequence
Argument over the extent of identification/analysis/ evaluation of consequences and their associated severity.

Figure 74: Knowledge of Risks

The goal G_Severity_Consequence would be developed by arguing over the extent of the identification, analysis and evaluation of the consequences and their associated severity. In essence it is an argument about the extensiveness of the hazard and accident analysis, with focus on identifying the range of consequences that could occur under intended and unintended behaviours of the system. Chapter 2 has identified the types of techniques that may be relevant to such analysis and Chapter 4 (Figure 16 and Figure 17) provides guidance on how it may be modelled and analysed. As developing this goal further is dependent on the type of hazard and analysis undertaken, it isn't developed further within this thesis. The developer may use existing hazard analysis argument patterns, such as (Department of Computer Science, 2004), to reason about knowledge of severity of consequences using evidence from hazard analysis.

The goal G_Stength_of_Defences is used to represent the knowledge of defences obtained through the reasoning presented in Chapter 4. Refer to Chapter 4 for continuation of the development of this goal.

8.2.3 Uncertainty of Risks

Figure 75 is the complement of Figure 74 in that it details the measuring of limitations in identification, analysis and evaluation of the severity of consequences and the strength of defences. This branch of the argument is very important for characterising likelihood in the risk equation as its purpose is to measure uncertainty, a key property of a likelihood estimate.

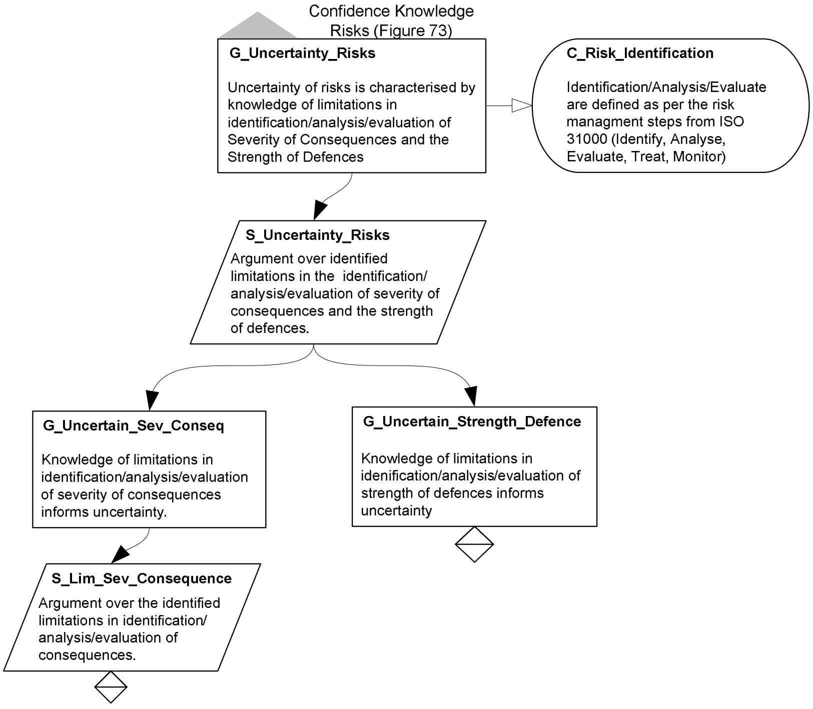

Confidence Knowledge Risks (Figure 73)
G_Uncertainty_Risks
Uncertainty of risks is characterised by knowledge of limitations in identification/analysis/evaluation of Severity of Consequences and the Strength of Defences
C_Risk_Identification
Identification/Analysis/Evaluate are defined as per the risk managment steps from ISO 31000 (Identify, Analyse, Evaluate, Treat, Monitor)
S_Uncertainty_Risks
Argument over identified limitations in the identification/ analysis/evaluation of severity of consequences and the strength of defences.
G_Uncertain_Sev_Conseq
Knowledge of limitations in identification/analysis/evaluation of severity of consequences informs uncertainty.
G_Uncertain_Strength_Defence
Knowledge of limitations in idenification/analysis/evaluation of strength of defences informs uncertainty
S_Lim_Sev_Consequence
Argument over the identified limitations in identification/ analysis/evaluation of consequences.

Figure 75: Uncertainty of Risks

The goal G_Uncertain_Sev_Conseq would be developed by arguing over the limitations of the extent of the identification, analysis and evaluation of the consequences and their associated severity. In essence it is an argument about the identification of shortfalls of the hazard and accident analysis. As developing this goal further is dependent on the type of hazard and analysis undertaken, as per Section 8.2.2, it isn't developed further within this thesis.

The goal G_Uncertain_Strength_Defence is developed in Figure 76. Figure 76 is a complement to Figure 23 in Chapter 4. While being based on the same underlying models as the Chapter 4 case, Figure 76 adjusts the sub-goals to provide positive measurement of shortfalls. For example G_Paths _Limitations examines for existence of additional propagation paths, G_LoD_Limitations examines for the necessity for additional defences in propagation paths, and G_Suitability_of_Defence_Limitations examines for unresolved weaknesses in individual defences.

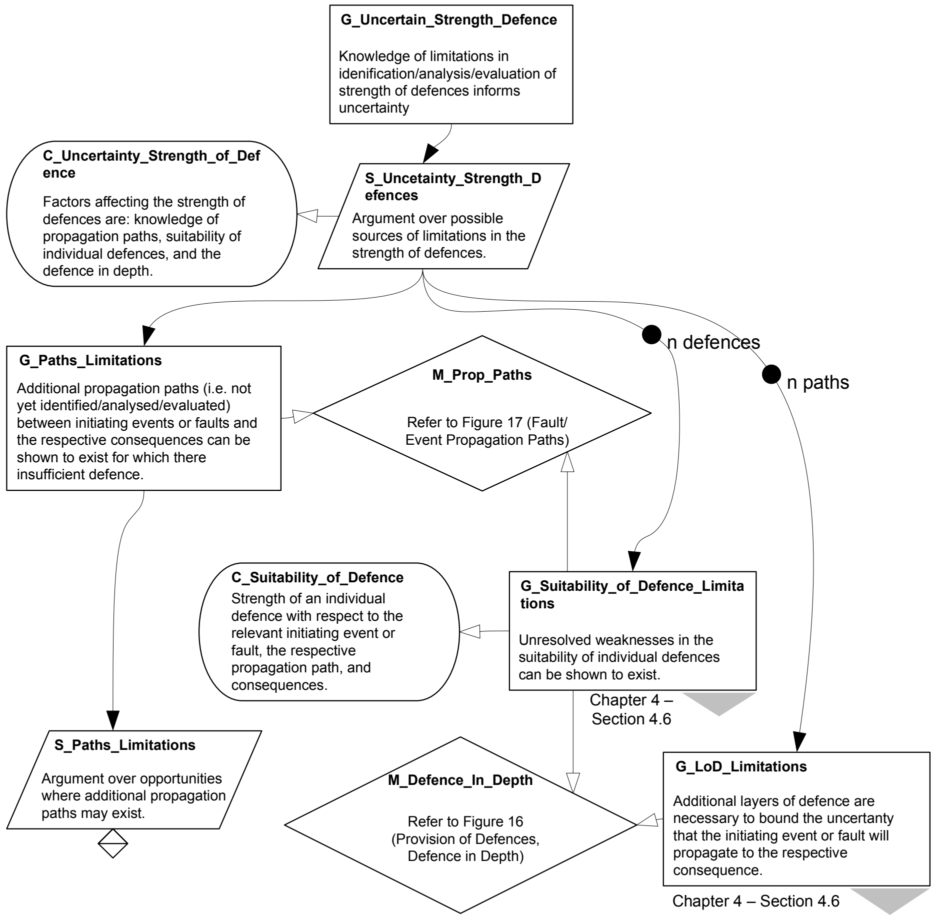

G_Uncertain_Strength_Defence
Knowledge of limitations in idenification/analysis/evaluation of strength of defences informs uncertainty
C_Uncertainty_Strength_of_Defence
Factors affecting the strength of defences are: knowledge of propagation paths, suitability of individual defences, and the defence in depth.
S_Uncetainty_Strength_Defences
Argument over possible sources of limitations in the strength of defences.
n defences
n paths
G_Paths_Limitations
Additional propagation paths (i.e. not yet identified/analysed/evaluated) between initiating events or faults and the respective consequences can be shown to exist for which there insufficient defence.
M_Prop_Paths
Refer to Figure 17 (Fault/ Event Propagation Paths)
C_Suitability_of_Defence
Strength of an individual defence with respect to the relevant initiating event or fault, the respective propagation path, and consequences.
G_Suitability_of_Defence_Limitations
Unresolved weaknesses in the suitability of individual defences can be shown to exist.
Chapter 4 – Section 4.6
S_Paths_Limitations
Argument over opportunities where additional propagation paths may exist.
M_Defence_In_Depth
Refer to Figure 16 (Provision of Defences, Defence in Depth)
G_LoD_Limitations
Additional layers of defence are necessary to bound the uncertanty that the initiating event or fault will propagate to the respective consequence.
Chapter 4 – Section 4.6

Figure 76: Uncertainty of Strength of Defences

G_LoD_Limitations and G_Suitability_of_Defence_Limitations refer to Chapter 4 for information on how they are developed further. While Chapter 4 doesn't develop these goals specifically, the information to develop these goals is obtained from examining the adequacy of analysis supporting the knowledge aspects from Chapter 4, and the associated relationships to Chapters 5 and 6.

8.2.4 Knowledge Versus Uncertainty

Sections 8.2.2 and 8.2.3 explain how knowledge and uncertainty of risks may be characterised, such that confidence in the knowledge of risks could be obtained from comparison per the discussion of Section 8.2.1 (Figure 73). Figure 77 describes that the knowledge outweighs the uncertainty when the knowledge no longer makes improvement to the knowledge reasonable. Figure 77 defines reasonable as being based the notion of a cost benefit analysis. There must be perceived benefit to improved knowledge of control of the risks, perhaps because the source of uncertainty reveals additional information about the behaviour of a defence which might undermine the utility of the defence, or because there are credible propagation paths not yet revealed by the existing knowledge. If the costs of turning uncertainty into knowledge are grossly disproportion to the potential benefit, then the knowledge improvement may not be reasonable.

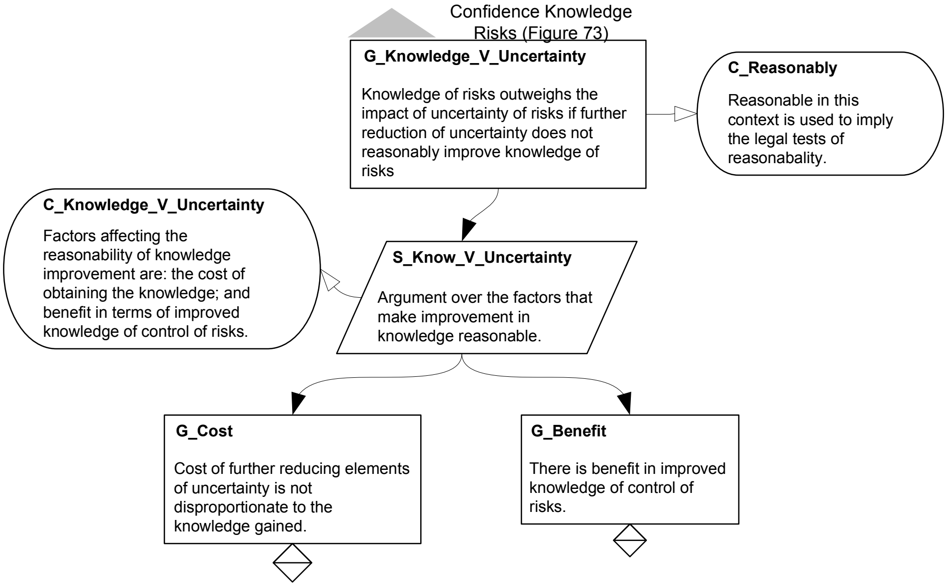

Confidence Knowledge Risks (Figure 73)
G_Knowledge_V_Uncertainty
Knowledge of risks outweighs the impact of uncertainty of risks if further reduction of uncertainty does not reasonably improve knowledge of risks
C_Reasonably
Reasonable in this context is used to imply the legal tests of reasonabality.
C_Knowledge_V_Uncertainty
Factors affecting the reasonability of knowledge improvement are: the cost of obtaining the knowledge; and benefit in terms of improved knowledge of control of risks.
S_Know_V_Uncertainty
Argument over the factors that make improvement in knowledge reasonable.
G_Cost
Cost of further reducing elements of uncertainty is not disproportionate to the knowledge gained.
G_Benefit
There is benefit in improved knowledge of control of risks.

Figure 77: Knowledge versus Uncertainty

Goals G_Cost and G_Benefit are developed further in Figure 78 and Figure 79.

Cost

Figure 78 shows that the total cost of reducing elements of uncertainty is made of the sum of the cost of reducing the uncertainty for each element of uncertainty. The cost of reducing elements of uncertainty has to be considered both individually for each element of uncertainty, and in total. S_Cost is argued both individually and in total. This is because the proportionality of the relationship between cost and benefit may differ depending on which pieces of uncertainty are combined.

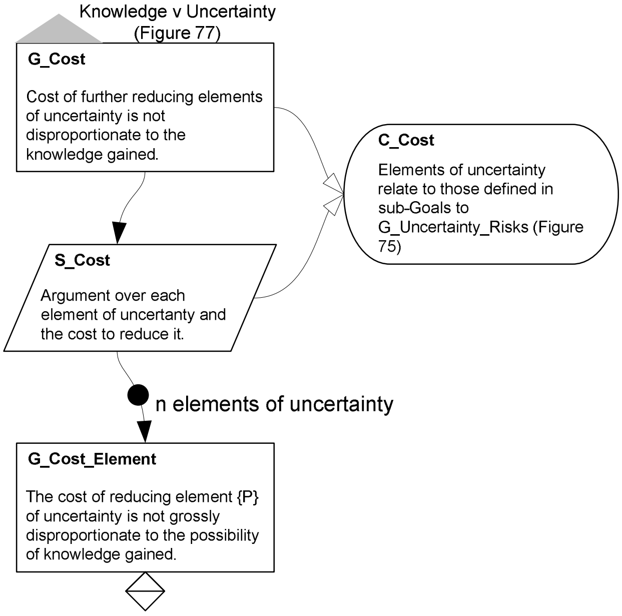

Knowledge v Uncertainty
(Figure 77)
G_Cost
Cost of further reducing elements of uncertainty is not disproportionate to the knowledge gained.
C_Cost
Elements of uncertainty relate to those defined in sub-Goals to G_Uncertainty_Risks (Figure 75)
S_Cost
Argument over each element of uncertanty and the cost to reduce it.
n elements of uncertainty
G_Cost_Element
The cost of reducing element {P} of uncertainty is not grossly disproportionate to the possibility of knowledge gained.

Figure 78: Cost of Additional Knowledge

Benefit

There is a benefit to improved knowledge, if the knowledge changes any of the factors informing risk (Figure 79).

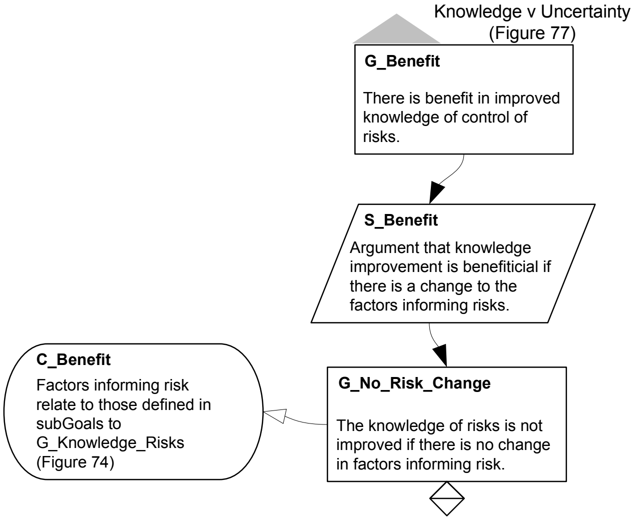

Knowledge v Uncertainty
(Figure 77)
G_Benefit
There is benefit in improved knowledge of control of risks.
S_Benefit
Argument that knowledge improvement is benefiticial if there is a change to the factors informing risks.
C_Benefit
Factors informing risk relate to those defined in subGoals to G_Knowledge_Risks (Figure 74)
G_No_Risk_Change
The knowledge of risks is not improved if there is no change in factors informing risk.

Figure 79: Benefit of Additional Knowledge

Figure 80 identifies the factors informing risk and how additional knowledge of these is likely to change the evaluation of risk. Specifically, the severity, additional paths, need for additional defences at differing layers and the adequacy of individual defences are all factors that might change the risk evaluation.

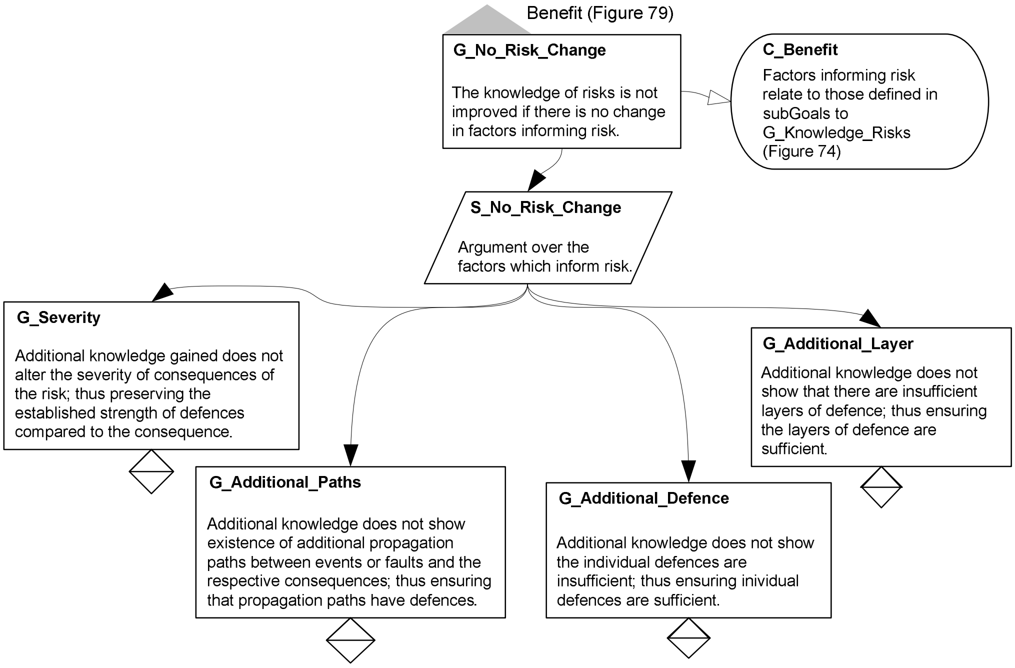

Benefit (Figure 79)
G_No_Risk_Change
The knowledge of risks is not improved if there is no change in factors informing risk.
C_Benefit
Factors informing risk relate to those defined in subGoals to G_Knowledge_Risks (Figure 74)
S_No_Risk_Change
Argument over the factors which inform risk.
G_Severity
Additional knowledge gained does not alter the severity of consequences of the risk; thus preserving the established strength of defences compared to the consequence.
G_Additional_Paths
Additional knowledge does not show existence of additional propagation paths between events or faults and the respective consequences; thus ensuring that propagation paths have defences.
G_Additional_Defence
Additional knowledge does not show the individual defences are insufficient; thus ensuring inividual defences are sufficient.
G_Additional_Layer
Additional knowledge does not show that there are insufficient layers of defence; thus ensuring the layers of defence are sufficient.

Figure 80: No Change in Risk from Additional Knowledge

The goals G_Severity, G_Additional_Paths, G_Additional_Defence, and G_Additional_Layer depend on the specific element of uncertainty, in the context of the chain of evidence and defences of a specific solution. For this reason, they are context dependent and are not developed further in this thesis.

A critic might ask how is the value of the knowledge known when the knowledge is not yet known. At one extreme, there is the value of knowing there is a problem, perhaps because some issue is known about from the evidence that already exists. At the other extreme, the value may not be fully appreciated until the evidence exists. The developer must estimate the value of the as yet unattained evidence based on the evidence they already have. If the uncertainty is significant, then it is not possible to show the cost is grossly disproportionate, unless the severity is very low. However, if the uncertainty in small, then the extent of knowledge will be suggestive of the value from the more specific sense in which the uncertainty can be characterised. Basing the comparison on the concept of a gross disproportionality aids in resolving this conundrum, by favouring seeking the knowledge unless it is very certain that it won't provide a benefit.

8.3 Conducting the Risk Evaluation

One means of establishing when knowledge outweighs uncertainty, is to use the ASAL, CSAL, and ESAL frameworks as the threshold for when knowledge outweighs uncertainty. In these cases, risk is increased when there is a limitation against the applicable ASAL, CSAL and ESAL thresholds specified by Chapters 4, 5 and 6 respectively. The following sub-sections explain how the risk maybe evaluated based on the architectural, claims and evidence aspects.

This approach of using the ASAL, CSAL and ESAL benchmarks as the threshold for knowledge outweighing uncertainty is valid because inherent within each SAL framework is an explicit evaluation of the impact of uncertainty.

8.3.1 Architectural Assurance Impact on Risk

Chapter 4 has established that the risk is related to the adequacy of defences for each propagation path between the initiating fault or event and the realisation of consequences. Therefore, the risk is increased whenever insufficient defences are provided within each given propagation path. Table 32 proposes a way in which this relationship may be expressed, with risk increasing for every reduction in defences on a propagation path.

Assured Layers of Defences[1] at Perspective Layers[2]	**Severity of Consequences**			
	Catastrophic	**Hazardous / Major**	**Minor**	**No Safety Effect**
No Defences	Very High	High	Medium	Low
One Layer Defence	High	Medium	Low	Low
Two Layer Defences	Medium	Low	Low	Low
Three Layer Defences	Low	Low	Low	Low

Notes:

1. A defence is assured when the CSAL is satisfied for each requisite lifecycle product needed to avoid a discontinuity in the chain of evidence, and each attribute of each lifecycle meets the requisite ESAL criteria.
2. The perspective layers are defined as per Chapter 4.
3. Risks are characterised as Low, Medium, High and Very High. Low risks are acceptable risks in this framework. High and Very High risks are generally unacceptable. Medium risks are only tolerable if the certification authority decides to retain the risk due to operational imperative rather than to treat it. There may be time bounds on how long the certification authority may wish to retain a medium risk, when the exposure to risk is evaluated against the cost of treating the risk.

Table 32: Architectural Impact on Risk

The risk quantities used in Table 32 are intended to be interpreted as relative rather than absolute. The quantities are intended to communicate the relative increase in risk over the specified benchmarks, and should be used to compare which risks are afforded the primary attention for treatment.

<u>Example – A-DHC-4</u>

Recall the example from Chapter 4 (Section 4.8). The initial design iteration only included defences at the intra-system and extra-system layers, and thus there was no defence at the direct layer. Using Table 32, the risk associated with using this design would be Medium, based on there being only two layers of defence and the severity being Catastrophic. Assume now that the system has already been designed, and that rather than simply revising the design at the conceptual or preliminary phases the design progresses to final certification with this limitation. The certification authority is faced with a Medium risk and must decide if the risk should be treated.

For this specific circumstance, the cost of doing this would be the cost of implementing the software change to perform the reasonability check of attitude data sources, a software change to the sensor data conditioning software component. This would require changes to requirements and design documentation, in addition to revised safety analysis, and verification and validation, including regression work. The developer estimates the cost of this to be $1M and three months impact to the schedule, although in practice the costs used within the cost benefit analysis will be dependent on much greater range of factors than just the developer price proposal. For example, the operational impact of delaying the capability, project office resources needed to deal with the extension, contract change proposal costs, etc. Full modelling of these costs is outside the scope of this example, but would be required in practice.

The benefit in risk terms is the reduction of risk from Medium to Low, based on the provision of the additional defence. The defence provides a means to ensure that the divergent attitude sources are now being trapped by the primary control computer, and therefore the primary computer continues operation during these events. By having the primary computer continue to operate, most importantly the defences for numerous other propagation paths that are implemented by the primary control computer are also preserved.

The project office would then evaluate the cost and benefit with respect to the grossly disproportionate measure, based on knowledge of all known risks they need to manage in order to establish if this treatment is to be implemented.

8.3.2 Claims Assurance Impact on Risk

Chapter 5 has established that the assurance of a defence is related to both the continuity in the chain of evidence of lifecycle products between constraint and implementation, and the degree to which knowledge of certain attributes of each lifecycle products is provided by evidence. Therefore, if there is an intolerable limitation in either the chain of lifecycle products or attributes of each lifecycle product, then the defence to which the constraint relates can no longer be claimed to be adequately assured. In these cases the achieved CSAL must be reduced for that defence to a CSAL for which the lifecycle hierarchy and evidence does achieve with tolerable limitations in evidence.

The assertion that a defence is not adequately assured has a direct bearing on the achieved ASAL and risk should be evaluated per Table 32 based on the revised numbers of assured defences at perspective layers. An example is provided at Section 8.3.3.

8.3.3 Evidence Assurance Impact on Risk

Chapter 6 has established that the assurance of an attribute of a lifecycle product is related to the relevance, trustworthiness and results of evidence. The tolerability of limitations in evidence with respect to relevance, trustworthiness and results of evidence has a direct bearing on the degree to which the CSAL can be claimed to be achieved.

Therefore, if there is an intolerable limitation in evidence of an attribute of a lifecycle product, then the CSAL of the lifecycle product hierarchy can no longer be substantiated and must be reduced.

Because of the direct relationship between ESAL, CSAL and defences, the reduction in achieved CSAL has a direct bearing on the achieved ASAL and risk should be evaluated per Table 32 based on the revised numbers of assured defences at perspective layers.

Example – A-DHC-4

Recall the example from Chapter 5 (Section 5.5) and Chapter 6 (Section 6.9), which considered that the direct layer defence, when implemented in the original design activity, has an identified evidence shortfall against the Traceable to High Abstraction attribute for the Low Level / Detailed Design Requirements Lifecycle Product.

An evidence shortfall against this attribute is intolerable for CSAL 3 because it breaks the chain of evidence between the implementation and the constraint level requirements. Using the approach describe above, this means the direct defence is un-assured and the risk is identified at Medium using Table 32. Closer examination reveals that the issue is due to incomplete traceability tables, due to a lack of independent review and software quality assurance review of the traceability tables. This impacts the traceability attribute but also the related verification attributes, due to the relevance and trustworthiness of the traceability information being degraded.

The cost to resolve the traceability issue and conduct the additional verification activities is $30K, based on approximately 300 hours work. The benefit is as per the examples discussion from Section 8.3.1, and so the comparison should find in favour of the treatment as $30K is much smaller than $1M from the earlier case.

Further, assume that during the verification based on the revised traceability table, additional faults are found with the direct defence, and problem reports raised to investigate the additional problems. In these cases the costs for correcting these would need to be estimated and the cost benefit activity repeated.

While this thesis doesn't spell it out further, it does reveal an interesting property of the way the cost benefit activity is structured. Note that in the first activity, introducing the direct defence in total was much more expensive than resolving the traceability issue for the evidence. In practice, the cost to resolve an uncertainty issue based on an evidence shortfall is usually proportionally much less than the cost to introduce additional defences. This suggests that a cost benefit analysis should often find in favour of doing analysis of the uncertainty, rather than not doing the analysis. In practice doing the analysis is usually much cheaper than changing the design, particularly in retrospect. While doing the analysis may suggest a design change, the investment to know either way, is usually proportionally smaller. This suggests that the factor identified in Section 8.2.4 which might undermine the cost benefit approach, will rarely undermine the viability of the approach in practice. These concepts are similar to Buying Information To Avoid Risk (BITAR) (Boehm, 2008).

8.3.4 Treating Risks

In the event that Table 32 identified that the risk is increased for a particular propagation path, then the developer has the following treatment options:

- reduce the severity of the consequences, thus reducing the need for additional layers of defences;
- provide an additional defence, in lieu of this inadequately assured defence; or
- resolve issues or provide additional evidence to satisfy the target CSAL/ESALs for this defence to provide adequate assurance of the defence.

Which option the developer chooses depends on the specific system and circumstances of the project. In civil aviation, the developer is normally bound to treat the issue within their system, whereas in military aviation, the airworthiness authority has greater flexibility to make choices about treatment and retention of risks. In cases where there are operational imperatives, the military authority may also choose to utilise external defences to manage risks, or at worst retain risks, until such a stage as one of the above treatments can be implemented (refer Chapter 4).

8.4 Summary

This chapter has examined how the concepts of assurance described in Chapters 4, 5 and 6 may be related to risk. Specifically, the traditional definition of risk has been examined, and difficulties with estimating likelihood in practice have been summarised.

A revised approach to evaluating risk has been described based on the strength of defences in the propagation paths between initiating faults or events, and the applicable consequences. The strength of defences is further characterised based on the proportionality of knowledge and uncurtaining pertaining to the defences and their evidence (refer Principle X and Y from Figure 72). The revised approach has been defined based on the work of earlier chapters within this thesis. A method has also been introduced for establishing when knowledge outweighs uncertainty based on a cost benefit analysis of improving knowledge through additional analysis, revised/additional defences, or revised/additional evidence. This information is then used to inform risk treatment and retention decisions and to inform ALARP evaluations (Principle A and B from Figure 72). An example was provided to illustrate the utility of using the described approach to risk evaluation.

The approach has been developed to adhere to the usability guidelines identified in Figure 72. The method minimises variability of risk communication by using a set of defined terms for qualitative risk evaluation (Guideline 1). Although subjectivity will never be absent, the method removes subjectivity from being immediately contributory to the risk evaluation by replacing a direct likelihood assessment with a more measureable indication of the strength of defences (Guideline 2). Acceptable and unacceptable risks are implied by the qualified terms used to describe risk level (Guideline 3), and the approach uses a cost benefit analysis to guide the extent of additional treatments determination (Guideline 4).

9 Issues and Assumptions

This chapter examines a series of issues relating to the assumptions made in this thesis, and reasons about their potential impact.

9.1 Imperfect Hazard or Failure Analysis

One problem is when the hazard or failure analysis is imperfect. This is because knowledge of hazards and failures is fundamental to making decisions about treating them. Depending on the framework, this dependency and impact may differ. Thus, it is important to question the impact of imperfect failure analysis, and how does this impact compare to other contemporary approaches?

9.1.1 Dependency No Worse Than Contemporary Approaches

One can argue that the dependency on hazard and failure analysis of the principles and framework in this thesis is no worse than standards such as ARP4754/DO-178, MIL-STD-882C/D/E, or DEF STAN 00-56 Iss 2. This is because the hazard and failure condition information used for decisions about establishing layers of defences (fault prevention and fault tolerance), is fundamentally the same information that these other paradigms use to make decisions about defining safety requirements. Thus any limitation in this information will have an impact on the completeness and appropriateness of what is allocated in any of these frameworks. The same can also be said for the safety argument paradigm (e.g. DEF STAN 00-56 Iss 4), where product related safety arguments tend to only argue the hazards and failures they know about.

However, there are several perspectives that the layered defence concept helps to reduce the impact of imperfect hazard and failure information.

Limitations Made Explicit

This framework prompts the developer to explicitly reason about limitations in the hazard and failure analysis, and to seek evidence of limitations in knowledge of severity of consequences, propagation paths, and defences. This provides a benefit in that the limitations in hazard and failure analysis are made explicit.

More Systematic

The framework incorporates the taxonomy of failure categorisation that can be applied from extra-system layer defences right down to direct defences. The application of these categories and the associated analysis to identify propagation paths, improves the degree to which the analysis and results will be systematic, and thus reduces the opportunity for omissions in hazard and failure analysis.

Defence in Depth

Thirdly, a major element of this framework is the concept of layered defences. This means that more than one defence is provided to prevent more severe failure consequences. If a hazard or failure has been missed in the analysis, then there is a greater potential that the higher level defences provide coverage for the unknown fault. The opportunity for defences to take a differing perspective (refer Section 4.4.2) depending on the layer in which they are defined (i.e. fault perspective, system perspective, functional perspective), greatly aids with coverage of faults that may not be known about at the fault perspective. While the unknown fault or event may not be treated in a way that it would have been had knowledge of it been forecast and treated directly, the perspective taken by the intra and extra-system layers should help to ensure the overall behaviour is within tolerable consequences. While absolute confidence can't be established from this, layered defences against the taxonomy of fault categories is a better circumstance that relying on single defences against unknown faults or events.

9.1.2 Facilitates Understanding of the Impact

If, once a system is fielded, an omission in the hazard or failure analysis is identified, this can have significant ramifications for the continued operation of the system. In some cases the operation of the system may have to be ceased until such a stage as the impact can be properly established. Therefore, it is important that the output of safety assurance frameworks facilitate impact analysis of such issues. To do this it will need to allow any substantive treatments to an issue to be identified, so that an accurate risk assessment can be prepared based on estimated consequences and the extant behaviours of the system that might treat it.

The framework provided by this thesis assists the impact analysis because it effectively provides a catalogue of defences (i.e. the fault prevention and fault tolerance mechanisms) the system has, how many of them there are, and information on how those defences work. This information can be used to assess if any existing defence treats the issue arising from the imperfect hazard and failure analysis, and permits an accurate risk to be established with minimal additional engineering investigation. This is an improvement over many existing frameworks because these other frameworks may 'obscure' possible defences that are implemented at component level, but that don't have obvious traceability and consideration at the system architectural level. Risk assessments in this framework are also built upon this knowledge of layered defences, and thus revising the risk assessment (refer Chapter 8) should be straightforward based on any shortfalls in defences.

9.2 Suitability of Defences and 'Constraints'

The suitability of a defence (and the associated 'constraint' which communicates the requirement for implementation by the system) is a function of the following:

- the coverage of the initiating fault or event in the applicable propagation path,
- the viability of the prevention or tolerance strategy based on the properties of the prospective fault or event,
- for fault tolerance strategies:
    - the detectability of the fault or the event,
    - the appropriateness of the handling mechanism,
    - the accuracy to which the fault transformation of the handling mechanisms can be estimated, and
- the degree to which the 'constraint' implements the defence.

While Chapter 4 and 5 provides stimuli for reasoning about these factors, it is evident that the suitability (or unsuitability) of these factors is a major contributor to the usefulness of a defence in depth based framework. In Chapter 4, the Fault Propagation and Transformation Notation (FPTN) was used to support reasoning about some of these factors, however it was evident that this notation and associated analysis is not conclusive about these factors alone. Therefore, the suitability of defences is dependent on validating the above factors. As validation requires sufficient target system context, it may be necessary to conduct laboratory and controlled test and evaluation (ground or flight) in order to generate the necessary evidence.

While a number of contemporary safety assurance standards refer to safety verification and validation of safety requirements, the benefit this framework offers is that it provides a structured framework for determining what properties of the system need to be validated (by evidence). However, it may be a more difficult property of the framework to implement because it may require a more empirical approach to validating defences than is currently evident in real world practice.

9.3 Independence and Diversity of Defences and 'Constraints'

Independence and diversity of defences (and the associated 'constraint' which communicates the requirement for implementation by the system) is fundamental to assuring the defence in depth of the layered approach. While Chapters 4 and 5 clearly identify the requirements of this independence and diversity, the practicalities of real systems means that there are many dependences and inter-relationships within a system that must be considered in establishing the independence and diversity. This can quickly become very complex and potentially unmanageable.

Figure 17 in Chapter 4 illustrates some of this complexity by superimposing fault propagation paths on each defence. Further to this, it is necessary to consider all enabler (e.g. common power supplies, data-buses), and interrelationships (both intended and covert), in establishing the strength of independence and diversity. Such considerations may suffer from similar limitations to those identified in Section 9.1.

There are notable examples where these relationships were not well understood in design (e.g. Anomalies in Digital Flight Control System F-16, X-28, X-3, C-17, YC-14, A320/330/340 (Rushby, 1993), B777 (Australian Transport Safety Bureau, 2007)). It continues to be a difficult and complex aspect of critical system design.

The benefits of the framework in this thesis are that it makes reasoning about defences, including their independence and diversity explicit, and the requirement to provide evidence of the analysis also explicit.

9.4 Managing Change

Change is inevitable, and thus an important usage for safety assurance information established during development is for that information to inform change impact analysis. Change may be imposed on a system externally by changes in operating environment, usage context and other external influences. Alternatively it may be imposed internally as a result of modifications, re-design and changed operational practices. Often these internal changes are prompted by one or more external impacts.

To inform change impact analysis, it is necessary to know such things as:

- the physical and logical make up of a system,
- the interfaces between its physical and logical components,
- the system behaviours and the components that make it up, and
- the system behaviours in the presence of faults (internally and externally induced).

While much of this information is outside the safety assurance context, safety assurance information can contribute to knowledge about each of these things. In Section 9.1.2 the usage of the information provided by the framework from this thesis was discussed with respect to managing the impact of imperfect hazard analysis. In many respects the catalogue of defences (i.e. the fault prevention and fault tolerance mechanisms) the system has, how many of them there are, and information on well those defences might work also assists change impact analysis. At a lower level, the knowledge of limitations in evidence also informs change impact analysis, as it provides a starting context on where evidence generation might best preserve or improve the tolerability of limitations during any change.

9.5 Summary

This chapter has provided explanation of issues and assumptions that impact definition and application of the concepts and framework described by this thesis. The layers of defences and the tolerability of limitations concepts offer a number of benefits when dealing with the impact of imperfect hazard and failure analysis, suitability of defences and ‘constraints’, and managing change. The next chapter provides evaluation of the contributions of this thesis.

10 Evaluation

In Chapter 1 the thesis proposition was stated as follows:

It is feasible to establish principles for defining effective safety assurance frameworks. These principles enable frameworks to be developed to satisfy safety objectives for military aviation systems in typical acquisition contexts.

Based upon the analysis of literature surveyed in Chapter 2, the thesis proposition was extended as follows:

This thesis demonstrates that it is feasible to establish principles and usability criteria for defining effective safety assurance frameworks for aviation systems in typical acquisition contexts. This thesis provides meta-arguments that can be used as the basis for defining a novel integrated framework for the assurance of aviation systems. The thesis demonstrates how this approach can be used to address the identified limitations and challenges of the certification of aviation systems. Further, by reducing uncertainty for supplier delivery of safety evidence across contracting processes, the framework is intended to help limit emergence of safety evidence issues, the resultant cost and schedule implications, and reduce the likelihood of retaining intolerable safety risks.

The proposition is supported by:

- establishing principles based on concepts that preserve the benefits and reduce the limitations of existing assurance paradigms for aviation certification;
- by examining the principles in the practical context of real aviation systems; and
- through the development and application of a novel integrated framework for assurance of aviation software systems based on these principles, addressing identified deficiencies in existing assurance frameworks.

The evaluation of this thesis proposition is addressed from two perspectives:

- demonstrating the feasibility and utility of the principles and framework defined by this thesis, and
- demonstrating the framework provides practical benefits in addressing the deficiencies of existing assurance frameworks.

10.1 Forms of Evaluation Applied

To establish the forms of evaluation to be applied, literature regarding the design of studies for research for social sciences has been consulted (refer (Van de Ven, 2007), (Oppenheim, 2001), (Robinson, 2000)). It is difficult to apply novel approaches to real projects at their initial proposal, and thus evaluation by practical experiment is not straightforward. The following forms of evaluation have been applied to evaluate the principles and framework presented in this thesis:

- Peer Review
- Constructed Example Case Study
- Anti-hypothesis Evaluation
- Audit Tool for Real System Development
- Contract Evaluation for Projects

These methods have been selected based on an assessment of the most effective way of evaluating the approach given the constraints of real-world evaluation. It is also not feasible to run real projects multiple times to provide both a control group and experimental groups, as is traditionally required in experimentation; to put it another way, the scientific method cannot be applied.

10.2 Overview of Research Evaluation

The following sub-sections provide an overview of how each form of evaluation was conducted and the level of evaluation the method provides.

10.2.1 Peer Review

Peer review provides evaluation with respect to the experience of participants. Therefore, it is important that suitable participants are selected and suitable methods are used to capture the experiences and judgments of practitioners. To address this, peer review has been undertaken by the following methods:

- survey questionnaire of stakeholders to safety assurance for aviation systems,
- one-on-one interviews with stakeholders,
- workshop sessions involving a group of stakeholders, and
- seminars providing presentation and opportunities for questions on the concepts.

The survey methods included both in-person interviews and independent completion by respondents in order to reduce the sensitivities of results to biases in just interviewing.

A detailed survey questionnaire was prepared using the principles for questionnaire design from (Oppenheim, 2001) and (Berdie, et al., 1986). The questionnaire asked a mix of open and closed questions regarding the concepts and application thereof presented in this thesis and the supporting literature (refer Appendix D). The following provides an overview of the structure of the survey questionnaire.

- Part A – Demographic
- Part B – Architectural Assurance
  - *B1 Motivating Issues*
  - *B2 State of Practice*
    - Treatment of Systematic Faults
    - Role of Architecture
    - Fault Avoidance and Tolerance
    - Fail Safe Design Criteria
    - Examination of Real Systems
  - *B3 General Principles*
    - Layers of Defences and Bounding Uncertainty
  - *B4 Our approach*
    - ASAL definition
    - ASAL Framework Application
    - Certification Assessments / Audits
    - Development by Design Agency
- Part C – Claims & Evidence Assurance
  - *C1 Motivating Issues*
  - *C2 State of Practice*
  - *C3 General Principles*
    - Key Principles of Assurance Levels
    - Relationship to Architecture
  - *C4 Our Approach*
    - CSAL Definition
    - Accounting for Behaviours
    - ASAL to CSAL Relationship
    - Attributes of Lifecycle Products
    - ESAL Definition
    - Trustworthiness of Evidence
    - Framework Application
- Part D – Contracting for Assurance of Military Aviation Software Systems
  - *D1 Motivating Issues*
    - Paradigm: Goal-based / Prescriptive
    - Integrating the Standard's Lifecycle with the Tender/Contract Lifecycle
    - Differences with Military System Acquisition Contracts
    - Impact of Uncertainty
  - *D2 State of Practice*
  - *D3 General Principles*
  - *D4 Our Approach*
    - Obtaining Architectural Certainty
    - Setting Benchmarks for Architectural Suitability
    - Informing Architectural Suitability
    - Evaluating Architectural Suitability
    - Providing Architectural Assurance
    - Obtaining Argument & Evidence Certainty
    - Setting Benchmarks for Argument & Evidence
    - Proposal of Argument & Evidence
    - Evaluation of Argument & Evidence
    - Argument and Evidence Assurance
    - Contracting Framework Application
    - Cost and Schedule Implications
    - Lifecycle Implications
    - Management Implications
    - Resolution within Contract Scope
    - Usability

Each part of the survey was structured to relate motivating issues to state of practice, to general principles to our approach. This structure provides a means of identifying deviations between supposed relationships between the different sections. It permits identification of where respondents diverge from the hypothesis, and provides correlation as to whether respondents are divergent at a general principle perspective, or at an implementation perspective.

Part A of the survey used rating scales based on demographic ranges specific to each question. To avoid scale bias, the rating scale for Parts B to D was designed to provide a symmetrical full scale for respondents per the guidance of (Berdie, et al., 1986), as follows:

- Strongly Disagree
- Inclined to Disagree
- Undecided
- Inclined to Agree
- Strongly Agree

The survey covered more than 20 participants representing stakeholders from:

- military and civil aviation domains;
- government and industry;
- certification authorities, project authorities, specialist consultants, industrial practitioners, and scientific laboratories;
- nations including Australia, Canada, United Kingdom, New Zealand and the United States of America; and
- Military Regulatory/Certification Authorities.

The criteria for participant selection included the follow factors:

- providing coverage of the aforementioned demographic categories;
- ensuring that the participants had appropriate qualifications and training in the domain, and have a professional attitude to improving safety assurance;
- establishing that the participants experience with real industrial programs; was first hand, and relevant to the topic and domain;
- confirming that participants represented a variety of ‘world views’; and
- ensuring that sample size of participant responses was sufficient to ensure trends could be clearly distinguished across the majority of the survey.

Respondent acquiescence to survey fatigue was avoided by giving respondents plenty of time (i.e. three months) to complete the survey, and by structuring the survey into separable sections that allowed completion in smaller sessions. Respondent acquiescence to the author's 'world view' was avoided to ensuring diversity of responses in the motivating issues, state of practice and general principles section, such that these diversity of viewpoints would be applied to the section on our approach. A public forum survey was not pursued because it would be too difficult to control the other participant selection criteria in such a context.

This coverage of stakeholders is somewhat unique and not typically achieved by other theses. Six of the surveys were elicited during a series of three workshops held at Australian defence facilities, in which the content of the survey topics was briefed by the author, and then the surveys completed through facilitation of discussion on each topic. Material from this thesis was also evaluated through the following conference presentations and seminars:

Event	Topic	Academic Peer Review
Improving Systems and Software Engineering Conference (ISSEC) Aug 2010	Architectural Assurance	Yes
IET System Safety Conference Oct 2010	Claims and Evidence Assurance	Yes
Australian System Safety Conference May 2012	Contracting Framework	Yes
Seminar to RAAF, DMO and Contractors at RAAF Richmond Mar 2011	Architectural, Claims and Evidence Assurance	No
Seminar to RAAF, DMO and Contractors in Canberra Jun 2012	Full Framework	No

Table 33: Conferences and Seminars

The limitation of the peer review form of evaluation is that potential deficiencies in the approach may not be revealed, as it is an intellectually abstract approach rather than a practical end-to-end approach. Participant selection may also affect the results. Therefore, peer review needs to be complemented by other methods that provide practical end-to-end evaluation, and avoid participant bias.

10.2.2 Constructed Example Case Study

Evaluation through case study has involved application of the approach using constructed examples derived from real-world systems. A constructed example has been used because application to a real project was not feasible within a doctoral program. Throughout Chapters 3 through 9 of this thesis the A-DHC-4 Advanced Caribou example has been used both for the purposes of explanation, but also to provide confidence in the utility of the principles and framework.

This form of evaluation must address the practicalities of real world systems. For this thesis, realism of the example has been achieved by deriving the A-DHC-4 example from elements of real world systems.

The limitation of the constructed example case study is that potential deficiencies in the approach may not be revealed as, while the example provided end-to-end evaluation, the example is limited to specific aspects. Therefore the constructed example case study needs to be complemented with a method of evaluation that identifies issues from coverage of entire projects.

10.2.3 Anti-hypothesis

The purpose of anti-hypothesis evaluation is to show whether or not known project problems can be correlated to aspects the principles and framework of this thesis. In essence, it is a search for evidence of problems when the approach suggested by this thesis is not used. This form of evaluation provides the benefits of revealing issues encountered across the full scope of real historical projects.

Anti-hypothesis evaluation has been conducted by examining problems in historical projects and correlating these problems as evidence of the anti-hypothesis. A review of 22 historical Australian Defence Force aviation projects between 1998 and 2012 has been conducted (refer Appendix E).

The limitation with anti-hypothesis evaluation is that it doesn't address the application to newly encountered problems. Therefore the anti-hypothesis evaluation has to be complemented by a method that evaluates the application to a problem in real time.

10.2.4 Audit Tool for Real System Development

In order to provide an evaluation method that examines a problem in a real world system, the principles and framework were applied as an audit method for a real world system development. During the development and verification audit phases of the C-130J Hercules Block 7.0 upgrade mission computer and flight management system software audit program, members of the international audit team trialled parts of the architectural, claims and evidence assurance framework as an audit tool to identify potential product and process shortfalls within the program.

This evaluation approach used lifecycle evidence from a real project to identify the effectiveness of problem identification when compared to RTCA/DO-178B audit methodologies (refer (Federal Aviation Administration, 2003), (Head of Certification Experts Department, 2011)). The limitation with this approach is that the principles and framework of this thesis were used to identify shortfalls, rather than to also treat them. However the constructed example case study evaluation described by Section 10.2.2 provides end to end application including treatment.

10.2.5 Contract Evaluation for Project

In order to provide an evaluation method that examines the principles and methods of this thesis for contract decision-making, elements of this thesis have been applied to the establishment and evaluation of a two phase contract arrangement (risk mitigation activity, and prime contract) for the Australian Lead In Fighter Capability Assurance Program, which is currently in progress. The development of the prime contract SOW was informed by assessing safety and assurance evidence produced under the risk reduction activity contract in order to establish the adequacy of the architecture and assurance evidence. While the ASAL, CSAL and ESAL method was not applied directly, the reviews of safety evidence directly considered the principles of architectural, claims and evidence assurance. Prime contract SOW clauses and CDRL requirements were refined based on the outcomes of these assessments.

The limitation with this approach was that the contractual framework was not applied in its entirety to this project. However, the evaluation of historical project contracts as part of the anti-hypothesis evaluation does evaluate circumstances where the framework of this thesis wasn't applied, and contracts were deficient in these respects.

10.3 Evaluation Results and Discussion

This section presents the results and discussion associated with each form of evaluation.

10.3.1 Peer Review Results and Discussion

Twenty surveys were completed by either individual response, interview or through several organised workshops over the period of February to May 2012. This sub-section summarises the results from the survey and discusses their implications for the framework described by this thesis. The results are presented per the structure of the survey questionnaire. Responses are characterised in the narrative of this chapter using the following definitions:

- *weak* – the two-thirds majority of responses either strongly disagree or are inclined to disagree.
- *mixed* – the responses are spread between disagree, undecided and agree.
- *positive* – the two-thirds majority of responses were inclined to agree.

- *positive to strong* – the two-thirds majority of responses are spread between inclined to agree and strongly agree, with inclined to agree having the majority of these responses.
- *strong to positive* – the two-thirds majority of responses are spread between inclined to agree and strongly agree, with strongly agree having the majority of these responses.
- *strong* – the two-thirds majority of responses strongly agree.

Normalisation of the results is achieved through constructing the survey with a consistent scale and approach to structuring questions across the different parts of the survey. Each section of the survey is also qualitatively[40] analysed in isolation, therefore avoiding numerical correlation between specific survey sections, and minimising the need to normalise sections of the survey against others.

Demographic

- Respondent experience was evenly spread in terms of years.
- Respondents were primarily from the aviation domain, although one third of responses took in the maritime, land and information systems domains also.
- Respondents were evenly spread between development and compliance assurance roles.

Architectural Assurance

Motivating Issues. There was strong indication in contemporary industrial practice of**:**

- hazardous sources of systematic faults not being adequately treated, or systematic faults preventing or disrupting the design certification and service release,
- architecture being used to provide mitigations to systematic faults in systems, and to provide layers of defences against sources of systematic faults, and
- the fail safe design criteria being used to treat of sources of systematic faults.

[40] Quantitative statistical analysis of survey results has not been undertaken, as it would provide limited benefit for the topics covered by the questionnaire. Should further quantitative statistical analysis be undertaken, results would require normalisation by assigning weights based on the relative strength of each question, importance of the question in the topics of evaluation, and the variance of response data.

State of Practice

Treatment of Systematic Faults. There was positive indication of inadequate treatment of systematic faults due to limitations in:

- design practices and assessment of requirements validity across the coupling of current software assurance standards with system safety methodologies,
- evidence showing that the behaviours of the system and software are appropriate with respect to safety, and
- current software assurance standards due to emphasis on process adherence rather than critical evaluation of product properties and behaviours.

Role of Architecture. There was strong indication that existing software assurance standards don't deal with system and software architecture in terms of its role to provide fault avoidance or fault tolerance of systematic faults.

Qualifying the Extent of Fault Avoidance and Tolerance. There was a mixed indication that current software assurance standards assist certification authorities establish the:

- effectiveness of the system's tolerance against systematic faults,
- the classes of systematic faults the system is tolerant against, and
- the extent to which any redundancy or other documented fault avoidance or fault tolerance mechanisms may be violated by the occurrence of systematic faults.

Fail Safe Design Criteria. While there was strong indication that the fail safe design criteria are intended to apply to both random and systematic sources of faults, and that architecture plays a critical role, there was mixed indication as to whether existing standards adequately encompassed the fail safe design criteria.

Examination of Real Systems. There was positive indication that the observations made in Chapter 4 regarding fault avoidance and fault tolerance with respect to systematic faults is valid. This indication was strong in regards to both the categorisation of layers, and also the numbers of perspective of layers when compared to real systems in practice.

General Principles

Layers of Defences. There was a positive indication that the layers of defences principle helps to bound the uncertainty of overall system behaviour in the presence of item/component/system failures.

Bounding Uncertainty. There was strong to positive indication that dealing with uncertainty is important in risk evaluations, and that architectural assurance is a suitable means of qualifying how uncertainty impacts risks.

Our Approach

ASAL definition. There was positive to strong indication that the ASAL framework is feasible, both in terms of the requirements of defences and also the role and perspective of the layers. There was also positive indication of the benefits of the ASAL framework. Respondents did note though that the ASAL framework needs to be coupled to an evidence framework in order to provide adequate evidence for certification.

ASAL Framework Application. There was positive indication that the ASAL framework would be both useable and beneficial to certification authorities for certification assessments, and that there would be benefits beyond current standards' approaches. Indication from design agencies was more mixed, although on the balance favoured positive indication rather than disagreement. Some respondents were positive about the concept of defences and 'constraints' although they had reservations that supporting methods as yet wouldn't enable them to model the relationships effectively. Extension to existing methods might be required.

The evaluation responses resulted in the labels for defence layer categories being refined from software, LRU and system levels to direct, intra-system and extra-system (refer Chapter 4), to provide a more universal description for the layers role in defence in depth.

Claims & Evidence Assurance

Motivating Issues. There was strong to positive indication that safety assurance standards should set product safety outcomes and evidence provision requirements (including benchmarks). There was mixed indication that safety assurance standards should set process requirements, although responses in the positive outnumbered those is disagreement. While flexibility was favoured, there was positive support for removing flexibility that leads to unsafe design (i.e. unsafe designs should not comply with the standard). There was strong indication of confusion over the role of current assurance levels in achieving safety.

State of Practice. There was positive to strong indication that assurance levels and integrity levels don't have any inherent product meaning, but that they should either have a direct product meaning, or be explicitly relatable to other qualitative measures that do have a product meaning. There was mixed indication that stakeholders know what claims and evidence are most appropriate for arguing safety.

General Principles

Key Principles of Assurance Levels. There was positive indication that assurance levels should have a product meaning, and that they should set objective benchmarks for the properties of the product that should be established. There was also positive indication that the assurance framework should make explicit the claims supporting the definition of the assurance framework. There was positive indication that the impact of limitations in compliance must be comprehensible.

Relationship to Architecture. There was positive indication that product and evidence assurance should be explicitly related, and that linking architectural and claims/evidence assurance was a plausible focal point. Respondents did not identify any other ways of linking product and evidence assurance.

Our Approach

CSAL Definition. There was positive to strong indication that the CSAL framework is feasible, both in terms of the hierarchy of lifecycle products and the attribute of them. There was also positive indication of the benefits of the CSAL framework for structuring claims.

The evaluation responses resulted in the expression of CSAL being refined when compared to earlier work to better express the intended relationship to limitations and discontinuities in the lifecycle product hierarchy.

CSAL Framework Application. There was positive indication that the CSAL framework would be both useable and beneficial to certification authorities for certification assessments, and that there would be benefits beyond current standards approaches. Indication from design agencies was consistent with the certification authority views.

Systematically Accounting for Behaviours. There was positive to strong indication that the lifecycle product hierarchy was appropriate, and that it captured the different abstractions at which evidence should be presented for systems.

ASAL to CSAL Relationship. There was positive indication that the proportional relationship between ASAL and CSAL levels was feasible, although it was acknowledged that there may be other approaches that would work in some contexts.

There was positive to strong indication that the CSAL to ASAL relationship provided appropriate failure condition severity and consequence context to the claims assurance.

Attributes of Lifecycle Products. There was positive indication that the attributes of the lifecycle product hierarchy was a feasible basis for an assurance framework, as this provides a logical structure for claims about evidence. There was positive indication that the set of attributes for lifecycle products is adequate. Respondents were positive about the benefits of the lifecycle product hierarchy and attribute models.

The evaluation responses resulted in the expression of Appendix B tables being refined when compared to earlier work to better align with the lifecycle product hierarchy and attribute relationships.

ESAL Definition. There was strong to positive indication that basing evidence assurance on the tolerability of limitations in evidence provision is feasible, and that it prompts useful questioning of the sufficiency of evidence. The perspectives of relevance, trustworthiness and results seemed logically sound. There was positive to strong indication that the ESAL framework is feasible. There were numerous concerns though that many developers are process oriented and may not be capable of structuring tolerability of limitations arguments objectively, although there was acknowledgment that this may be a better circumstance that an entirely unconstrained goal-based paradigm. There was a very mixed response regarding the characterisation of trustworthiness based on competency frameworks, indicating limitations in the support for current competency frameworks. There was also concern that arguments about trustworthiness would be nothing more that rhetoric.

ESAL Framework Application. There was positive indication that the ESAL framework would be both useable and beneficial to certification authorities for certification assessments, and that there would be benefits beyond current standards' approaches. Indication from design agencies was consistent with the certification authority views.

Many respondents indicated that the 'tolerability of limitations' concept appeared useful in that it provides some inherent rules for providing and measuring supplier justifications. There was some support for developing the concept further, and providing further examples.

Overall ASAL, CSAL, ESAL Framework

In regards to the framework overall, the survey indicated the following:

- Acquirers and Certification Authorities indicated that the approach may avoid several historical (and current) project issues where architectural safety shortfalls were responsible for project cancellation or significant project delays and cost increases. However, they noted that correlation in retrospect is easier than while projects are running.
- Some respondents were deterred by the notion of yet another assurance framework. Others noted that current approaches had limitations, and this approach seems compatible and extends current approaches.
- Some respondents were deterred by the complexity of the inter-related assurance concepts and contracting mechanisms, although several indicated that the concepts were less complex than many of the systems to which they would apply. This would perhaps provide natural selection of suppliers that cope better with complexity.
- The majority of respondents indicated that one or more fully worked examples of implementing the underlying ASAL/CSAL/ESAL frameworks would be beneficial.

Contracting for Assurance of Military Aviation Software Systems

Motivating Issues.

Standards Paradigm: Goal-based or Prescriptive. There was positive to strong indication that:

- the standard paradigm (i.e. goal-based or prescriptive) is a crucial factor in contracts for achieving adequate provision of evidence and thus effective regulation;
- goal-based standards permit flexibility for designers which give benefits in defining effective products, but goal-based standards may lead to limitations with establishing contractually enforceable benchmarks;
- prescriptive standards set benchmarks for evidence and activity completion, but they may lead to limitations in relevance of the evidence to product safety; and
- the safety assurance paradigm should be compatible with contracts and that contracts which provide cost and schedule certainty are preferred by both suppliers and acquirers.

Integrating the Standard’s Lifecycle with the Tender/Contract Lifecycle. There was positive to strong indication that:

- integration of the safety assurance standard with the contract is crucial and that it should assist in reducing uncertainty about the product, argument and evidence prior to the establishment of a contract (i.e. through pre-contract processes);
- the contract and standard should support the resolution of safety issues, and not hinder it by contributing uncertainty to the dispute; and
- there is evidence in industrial practice of project slippages, overruns or cancellations due to issues concerning safety assurance and certification using current approaches.

Differences with Military System Acquisition Contracts. There was positive to strong indication that regulatory enforcement is enabled by the contract rather than via laws for the military circumstances, although respondents acknowledged that laws still applied.

Impact of Uncertainty. There was positive to strong indication that:

- uncertainty in the specification of design requirements and provision of assurance evidence through the contract increases the risk of the contract being unsuccessful; and
- information regarding design solution, safety argument and evidence, if sought and used effectively during tender processes, can reduce uncertainty, and thus reduce potential contract success risks.

State of Practice. There was positive indication that existing standards and contracting approaches offer limited guidance on how:

- safety assurance standard and contract integration can be achieved effectively; and
- safety regulation should be achieved through contractual mechanisms.

General Principles. There was positive indication that:

- obtaining architectural certainty from the tender phases and prior to contract enables early insight into potential architectural shortfalls that may impact safety;
- setting of benchmarks for architectural suitability assist with architectural certainty, and they should set measurable criteria against which different solutions can be evaluated;
- obtaining argument and evidence certainty from the tender phases and prior to contract enables insight into potential argument and evidence shortfalls;

- setting of benchmarks for argument and evidence suitability assists with claims and evidence certainty, and that they should set measurable criteria against which argument and evidence can be evaluated; and
- such information can be used to evaluate tenders.

Our Approach. There was positive indication that the contracting process defined by this thesis is feasible. There was positive response to knowledge of architecture being provided during tender processes, although some suppliers were concerned about how they might progress their design processes to that point for some tenders, particularly those involving sub-vendors. Acquirers and suppliers indicated that the proposed approach does provide product and evidence focus during the tender phase that appears beneficial, although until they actually apply it, this is only speculation. There was consensus that knowledge of architecture and knowledge of evidence at tender would reduce the difficulty of contract execution. The majority of respondents indicated that one or more worked examples of a tender costing based on the proposed tender clauses would be beneficial.

Cost and Schedule Implications

- Suppliers expressed reservations about being able to resolve issues they haven't costed within contract scope, although praised that the underlying frameworks would potentially provide improved knowledge of product and evidence requirements during tender phases and thus reduce the risk of a need for issue resolution within contract.
- Some suppliers and acquirers expressed concern that this would increase the cost of tender processes, and potentially deter some tenderers.
- Some suppliers had reservations about the perceived paradigm shift, and how they would cost effectively educate their staff to work within such a framework.

Risk Evaluation

- Regulators and operational representatives indicated that knowledge of product behaviours and remaining defences would help with planning operational treatments, and with developing emergency procedures.
- Regulators indicated that they were still unclear how evidence/assurance shortfalls correlated to risks, and suggested developing the framework further to address risk measurement.

Analysis of Peer Review Results

Analysis of the peer review results indicates the following:

- There is correlation between the respondent comments and the motivating issues. This indicates that the motivating issues are probably valid.
- A cross section of prescriptive versus goal-based 'world views' were evident in responses to motivating issues and general principles revealing that there is diversity in 'world views', although the results don't directly suggest a resolution.
- There was not a direct correlation between 'world views' of supplier and regulators and position/negative comments indicating that there are broader issues of 'world view', education, paradigm shaping, and framework limitations involved.
- There is correlation between respondent comments on feasibility and usefulness and the general principles on which the framework is based. This indicates that the general principles may be widely agreeable, even if their opinions on the framework differ.
- Suppliers focussed strongly on cost and schedule implications, and competitiveness with respect to other suppliers. The level of knowledge on the topic of safety assurance varied substantially between suppliers, and also between suppliers, acquirers and regulators.
- While there was supplier sentiment that regulations are already too constraining for their businesses to be innovating, there was acknowledgement of the problems with the current approaches to assurance.
- Acquirers focussed on successful tender processes leading to successful contract execution. The level of knowledge on the topic of safety assurance varied substantially between acquirers and regulators.
- Views of safety and risk varied between respondents, as did appetite for risk.

Overall the peer review has provided evidence of feasibility of both the underlying principles, concepts and framework of this thesis. Benefits have been confirmed, and in some cases additional benefits highlighted. A number of issues pertaining to clarification of the framework were identified, and these have been addressed throughout the write-up of this thesis. A number of limitations with the effectiveness of the framework have also been identified, and these will require further evaluation in the context of a real world case study, albeit the evidence suggests that the benefits may outweigh the limitations.

10.3.2 Constructed Example Results and Discussion

The use of the A-DHC-4 Advance Caribou example (which is abstracted from several real world systems) through Chapters 4 – 8, has demonstrated both the feasibility of the concepts and framework, as well as the potential applicability to real systems. Because it is abstracted from elements of several real world systems and architectures, the example has validity to other similar aviation systems. The properties of architectures used in these systems also have commonality with the automotive and rail domains, and thus the results have potential read across to these domains also. Domains that use protection systems, such a nuclear power, use architectures that differ from those addressed by this example, and thus further analysis outside the scope of this thesis would be required to establish the amount of read across to these other such domains.

The examples has also shown that it is feasible to use the processes established by Chapter 4 to 8 for architectural assurance, claims assurance, evidence assurance, contracting for assurance and relating assurance to risk. As these processes are based on the concepts and principles used by the framework, then the example also provides evidence of the feasibility of the concepts and principles. Chapter 9 also provides analysis of the soundness of concepts.

Each example fragment shows the benefits of the approach, in terms of how the product design is improved, or how limitations in assurance evidence are resolved. Give that the A-DHC-4 example was based on elements of several real world systems, then it is likely that these benefits would be realised in practice also.

10.3.3 Anti-hypothesis

The review of 22 historical Australian Defence Force aviation projects between 1998 and 2012 (refer Appendix E) has identified the following correlations between the concepts of this thesis not being applied and risks based on either product shortfalls or risk/uncertainty being retained. The study examined historical project evidence using the following criteria:

- Project Paradigm – new acquisition, modification, or acquisition including modification,
- Purchase mechanism – commercial contract or foreign military sales,
- Airworthiness Authority – civil or military authority,
- Safety and Software Assurance – safety standard, safety argument notation, software assurance standard,

- Risk Retention via formal instrument such as an issue paper – product and uncertainty based risks,
- Safety Architecture – evidence of the architectural concepts of this thesis,
- Assurance Evidence – evidence of the assurance concepts of this thesis, and
- Contractual Impact – the cost / schedule impact due to limitations in any of the aforementioned topics.

Appendix E shows the following:

- limitations in assurance evidence or safety architecture, correlates with elevated risks based on product safety limitations;
- limitations in assurance evidence or safety architecture, correlates with elevated risks based on uncertainty;
- limitations in assurance evidence or safety architecture, correlates with cost and schedule impacts;
- the non-application of software or safety standards, correlates with elevated risks based on uncertainty;
- while the purchasing mechanism could not be correlated to retention of risk directly, there was evidence that foreign military sales purchases sometimes tolerated greater levels of uncertainty;
- there was variation in the risk appetite between civil and military airworthiness authorities, as was there variation from military authorities of different countries;
- modification seemed to lead to a greater opportunity for uncertainty based risks than did initial acquisitions – data access seemed to be a major reason for this;

Interestingly there was not a specific correlation between limitations in knowledge of architecture and in claims/evidence leading only to uncertainty based risks. There was also indication that uncertainty in architecture and claims/evidence can relate to product risks. This observation reveals that it is necessary to relate architecture and evidence in informing knowledge and risks, as has been recognised by the principles established by this thesis, and the ASAL/CSAL/ESAL framework that implements the principles.

While not conclusive proof of the effectiveness of the concepts and framework of this thesis, the anti-hypothesis evaluation has shown strong evidence of problems developing in real projects when the concepts and principles asserted by this thesis are not applied in their entirety. As a major case study has not yet been conducted, it is not yet possible to assess if there are cases where these concepts and framework also lead to similar problems as those in historical projects, however the other evaluations in this thesis do suggest the benefits should be realisable.

10.3.4 Audit Tool for Real System Development

The concepts and principle of this thesis were used as an audit tool during audits the C-130J Hercules Block 7.0 upgrade mission computer and flight management system software development. The audit team members made the following conclusions:

- **Positives** – several benefits were identified:
  - Defences could be identified that correlated to the direct, intra-system and inter-system layers. For several fault propagation paths examined, the numbers of defences was proportional to the severity of the hazard.
  - Identified shortfalls in defences in several areas lead to the identification of shortfalls in software safety analysis evidence, thus revealing the benefits of examining for defences.
  - Examining for defences to specific faults/failures of interest was beneficial to the audit process.
  - Constraints in the form of software safety requirements were identifiable and could be associated with defences. Using selected constraints as the focal point for evidence examination was both feasible and beneficial, and was also consistent with the underlying methods of the civil software approval guidelines.
  - It was feasible to establish lifecycle product hierarchies using real evidence.
  - Identifying evidence limitations using the lifecycle product attributes and the properties of evidence: relevance, trustworthiness and results was both feasible and beneficial.
  - Ranking limitations in evidence using the tolerability of limitations concept was both feasible and beneficial to making audit recommendations.
- **Issues** – several issues were raised:
  - Identifying the fault propagation paths was not always possible due to deficiencies in contractor evidence. However, the positive to this finding was that the deficiencies in contractor evidence were now made explicit, and could be ranked along with the other audit findings.
  - The language used to define to layers of defences in the earlier iterations of the architectural assurance work were associated with where and how the defence was implemented, but didn’t properly indicate the perspective with

regard to the fault being treated. This issue has been addressed, through the clarification of layers of defences and perspectives in Chapter 4.

- Auditors sometimes disagreed on the limitations of methods to relevance and trustworthiness, although the disagreement was beneficial in regards to the debate that followed. This issue has been addressed, through the categorisation of limitations in Chapter 6.
- The auditors commented on the possibly large number of propagation paths and therefore tolerability of limitations assessments required for most practical systems. While each assessment would be a valid assessment, the auditors expressed interest as to whether the assessments could be grouped or prioritised. While the number of propagation paths is a function of the size and complexity of the system in question, opportunities have been identified in Chapters 5 and 6 on how to group assessments. Chapter 8 also suggests a means of ranking the extensiveness of assessments based on failure mode severity and risk.

The trial application of the framework showed that it complemented existing audit approaches, and the framework will have further usage by Australian Defence Force audit programs.

10.3.5 Contract Evaluation for Project

Under the risk mitigation activity contract phase of the Lead-In-Fighter Capability Assurance Program, the following relevant safety assurance planning and evidence generation has been undertaken:

- Type Certification Planning
- System Safety Program Planning
- Software Safety Program Planning
- Software Development Assurance Planning
- Systems Requirements Specification
- Design Specifications
- Design Documents
- Preliminary Hazard Identification
- Mitigation Strategies for Hazards
- System-level Fault Trees

Reviews by the project office of safety assurance plans and evidence were undertaken against the concepts of this thesis in order to inform SOW development for the Prime contract phase. Safety evidence was reviewed by the project office in order to identify possible architectural defences, and the associated design requirements as constraints. System Safety and Software Plans were reviewed to identify lifecycle product hierarchies applicable to constraints, as were they reviewed to identify the scope of evidence proposed. The evidence was gauged in critical areas by carrying out relevance and trustworthiness assessments. The results of these assessments identified a number of shortfalls against the CSAL and ESAL benchmarks. These findings were used to inform project office CDRL feedback comments.

A holistic assessment of the limitation in confidence in the knowledge of risks based on knowledge and uncertainty from the aforementioned assessments was undertaken in order to inform SOW clause development for the prime. The assessments resulted in additional SOW clauses being developed in the topics of type certification, system safety and software assurance, and DID content of several safety and software CDRLs being revised. In some key areas, these revisions were undertaken through instantiating clauses based on similar intent to those clauses defined in Chapter 7 of this thesis. Due to the current nature of this project, additional information can't be disclosed in this thesis due to commercial sensitivities.

Approval to enter into the contract for the Prime Contract phase by the Australian Government and Defence Materiel Organisation was provided and contract signature was achieved in June 2013. The contract authority was able to obtain approval from the Australian Government to proceed to contract signature based on sufficient confidence in the knowledge of risks, and tolerable understanding of the uncertainty involved in those risks. This risk assessment was supported by independent attestations from technical and operational authorities who were informed by the project office's evaluation activities and evidence provided. This outcome provides some practical evidence that contract evaluations and refinement based on the concepts of this thesis was able to clearly inform contract approval decisions based on project risks.

The outcomes from the trialled application to this contract evaluation were positive, particularly when coupled with a two phase contracting approach that conforms to the intent of the tender / contract distinction used in Chapter 7.

10.4 Further Evaluation

Due to the type of problem this thesis covers, and due to the timescales of a Doctoral Program, it has not been possible to completely exhaust the chosen means of evaluation. For example, the peer review and constructed example were both bounded to fit to timescales. Peer review would benefit from an expanded set of respondents, and the construction example could be expanded in scope, both of which might provide additional information useful to evaluation. Furthermore there are opportunities to expand the application as a contract evaluation and audit tool to more projects. In terms of providing more conclusive evaluation of the feasibility and effectiveness of concepts and integrated framework of this thesis, such expansion of the evaluation would be required. Further work in relation to evaluating under a major real world case study is also suggested in Section 11.2.1

10.5 Summary

The thesis proposition given in Chapters 1 and 2 stated that it was feasible to establish principles for defining effective system and software safety assurance frameworks, and that it is feasible to develop and contract to a framework that is based on those principles. The thesis and the evaluation in this chapter have demonstrated the feasibility of established principles for safety assurance and defining an assurance framework based on these principles. Benefits of the assurance framework based on these principles have been identified. A number of limitations with the effectiveness of the framework have also been identified, and these will require further evaluation in the context of a real world case study, albeit the evidence suggests that the benefits may outweigh the limitations. Opportunities for further evaluation have also been identified.

However, it will only be through the extended practical application of the concepts described in this thesis on real projects, that the feasibility and practicality of the approach can be fully evaluated.

11 Conclusion

11.1 Contribution

This thesis has defined and presented the results of evaluation of an approach to the safety assurance of aviation systems to combat limitations in contemporary safety and assurance standards which cause programmatic and certification difficulties. These limitations have been identified from a study of Australian Defence Force aviation projects. However as many Australian projects are multi-national, the limitations are more widely experienced. The contribution of this thesis is in the following areas:

- **clarification of the purpose of safety assurance;**
- **definition and evaluation of principles of safety assurance and usability criteria based on the improved purpose statements;**
- **definition and evaluation of meta-arguments which provide the rationale for definition of an assurance framework based on the stated principles and usability criteria;**
- **definition and evaluation of a novel integrated framework for the safety assurance of aviation systems based on the defined principles and usability criteria;**
- **definition and evaluation of a process for application of the novel integrated framework; and**
- **definition and evaluation of an approach for contracting for safety assurance.**

The following sections draw conclusions from each of the elements on the research.

11.1.1 Conclusions on the Principles and Usability Criteria

In Chapter 2 the current state of safety assurance and assurance standards is surveyed. The survey identified variation in the 'world-view' or 'Weltanschauung' of what's important in safety assurance, leading to a need to clarify the purpose of safety assurance and assurance standards. The improved statements (Chapter 3) of the purpose of safety assurance and assurance standards provide clarification and explanation for variation in 'world-view', and are in that sense a contribution.

While Chapter 2 revealed a wide range of research into specific issues associated with safety assurance, there has been limited integration of contemporary concepts into a holistic model for safety assurance. This thesis has established principles of safety assurance and usability criteria (Chapter 3) based on the improved purpose statements.

They are intended to capture important relationships between topics of process and product evidence, argument/rationale, product behaviours, and risk evaluations; and to contextualise these principles by establishing usability criteria for practical implementations of safety assurance. The model emphasises that safety assurance is a substantially greater topic than the safety argument, due to the broader role of standards. The contribution here is twofold: the holistic model of principles and usability criteria; and capability to deal with systematic behavioural properties which have previously been problematic to deal with in risk assessments.

The evaluation has provided positive indication for general principles, and confirmation that the principles preserve benefits and reduce some of the limitations of existing assurance paradigms for aviation systems.

11.1.2 Conclusions on Architectural Assurance

Chapters 4 and 8 identify that architecture is an important factor in characterising the knowledge and uncertainty of product behaviours, which is crucial to evaluating systematic behavioural risks. This is a topic that has been largely absent from contemporary assurance standards. A model is established for classifying architectural defences, where no model existed previously. The key contribution is the definition of meta-arguments which provide the rationale for architectural properties as a basis for an assurance framework. An architectural assurance framework is established based on these principles and meta-arguments.

The evaluation has provided positive indication of the benefits of using architectural properties to inform risk evaluations. Evaluation of the approach also indicates that subjectivity is limited to reasoning about specific architectures adhering to general principles of the role of architecture, rather than diversion from general principles through inferior argument. While it is acknowledged that the aviation domain specific context to assumptions underpinning architectural solutions may ultimately limit the universality of framework, the underlying concepts are sufficiently generic that they could be evaluated for other domains.

11.1.3 Conclusions on Assurance of Product Behavioural Knowledge and Evidence

Chapters 5 and 6 identify that the role of evidence in informing product behaviours depends on where and how the evidence is used in the hierarchy of lifecycle products. A consolidated view of the role of evidence with respect to assurance has been largely absent from contemporary assurance standards. A model is established for classifying the hierarchy of lifecycle product evidence, and the properties of evidence. The key contribution is the definition of meta-arguments which provide the rationale for properties of lifecycle product evidence as a basis for part of an assurance framework, and the concept of tolerability of limitations in evidence. A claims and evidence assurance framework is established which builds on the meta-arguments and the properties of evidence in establishing knowledge of product behaviours.

The evaluation has provided positive indication that the evidence categorisations are consistent with the types of evidence used, and that the tolerability of limitations concept is beneficial in terms of addressing the practical aspects of assurance. Evaluation of the approach also indicates that subjectivity is limited to reasoning about specific evidence adhering to general principles. The meta-arguments also offer explanation for assurance level definition that have been largely absent from contemporary assurance standards.

11.1.4 Conclusions on Contracting for Assurance

Chapter 7 identifies that safety assurance approaches have often ignored the topic of contracting for safety assurance, and have presented difficulties across contractual boundaries. The key contribution is the definition and evaluation of an approach for contracting for safety assurance based on the rationale of Chapters 4 through 6.

The evaluation has indicated that the approach should help identify ‘strong’ versus ‘weak’ solutions, and thus better inform tender selection, and contract execution. In the long term, it is hoped that this will reduce the frequency and magnitude of programmatic and certification difficulties due to limitations in safety assurance.

11.1.5 Overall Conclusions

The topic of safety assurance continues to generate widespread debate across both academia and industry. This thesis has stated that safety assurance frameworks are only effective if it is not only possible to produce a compelling safety case, but it is probable. The use of case-by-case arguments in goal-based approaches, or the use of prescriptive methods all present practical issues which limit effectiveness. This thesis has examined an approach which applies a compromise between case-by-case arguments and pre-constraining arguments to reduce the impact of issues which limit the practical effectiveness of safety assurance approaches. The contribution of this thesis is to show that there are practical benefits of minimising subjectivity and reducing variability by compromising between case-by-case arguments and pre-constrained arguments. Where subjectivity and variability is limited to the specific solution and evidence set, rather than with regards to adherence to general principles of architecture, product behaviours and the hierarchy of evidence, then subjectivity and variability is better managed and fewer 'bad' solutions should result. The thesis also shows that it is feasible for assurance levels to be traceable to a product meaning, and to be informative to risk evaluations in ways that existing approaches are not. While it hasn't been possible to provide conclusive proof that the models and rationale of this thesis are scientifically complete, it is not necessary to do so to show that frameworks developed based on these principles are useful. The likelihood of wasting resources on matters which are not material to safety should, on balance, be lower than using contemporary approaches. The practical limitations of the evaluation lead to the conclusion being indicative rather than definitive. Section 11.2 suggests further work to satisfy the demand for evidence to improve the robustness of conclusions.

11.2 Further Work

During the conduct of the research presented in this thesis, opportunities for future work have been identified. These opportunities provide indication that the contribution of this thesis offers more widespread benefits than those concluded within. A brief introduction to these areas of research is presented in the following sections.

11.2.1 Major Real World Case Study

While the soundness of concepts and the novel integrated framework has been evaluated, it is not been possible to carry out a major real world case study within the timescales of a Doctoral Program. Further validation by one or more real world case studies is necessary.

Such a real world case study would involve the application of this novel integrated framework to the entire design lifecycle. The design problem should be one that avoids novelty, as it will be necessary to make a comparison of the architectures, claims and evidence generated in such a case study against those developed using traditional approaches to assurance. Comparisons could then examine differences in the architectures, claims, and evidence between approaches, reason about how these differences inform risk evaluation, and draw conclusions on the effectiveness of the framework from such results.

11.2.2 Investigation into Other Safety Standards from Other Domains

The meta-arguments expressed by Chapters 4 through 6 of this thesis rely on the properties of aviation system architectures, implicit arguments of rationale, and contemporary types of evidence used within the aviation domain. To evaluate the universality of these concepts, it would be beneficial to investigate whether these concepts require adaptation in order to be applied to safety assurance in other domains, and whether these concepts could be expressed at an even more general level.

11.2.3 Use of Meta-arguments for Expressing Rationale of Standards

This thesis has used meta-arguments for expressing the rationale of the underlying concepts around which an assurance level approach has been developed. As the problem of implicit rationale affects standards across many domains, it would be beneficial to investigate whether the usage of meta-arguments for expressing rationale in standards provides benefits to the soundness and utility of assurance standards.

11.2.4 Application of Re-defined Risk Paradigm to Other Domains

The re-defined risk paradigm within Chapter 8 of this thesis provides a method of characterising and evaluating systematic contributions to risks. As many other domains also are bound by probabilistic risk methodologies, many of which suffer similar limitations to those historically applied in the aviation domain, there may be benefits to investigating if these concepts read across or require adaptation for other domains.

11.2.5 Applicability to ALARP

ALARP implicitly assumes that the cost of information to inform risk assessments is negligible when compared with the costs of treatments to risks. This thesis has shown that there is a minimum cost of information needed to inform risk assessments, and has provided a way for managing the costs and benefits of this information with regards to establishing confidence in the knowledge and uncertainty of risks. This thesis has also provided a way for evaluating contributions from systematic behaviours (e.g. software) in relation to risk. There would be benefits to further investigating whether clarification of the ALARP principle based on the cost of information needed to inform risk treatment or retention decisions provides a means of resolving the long standing difficulties of ALARP with technologies like software.

11.2.6 System of Systems

This thesis was not intended to address system of systems; however there are concepts developed in this thesis that may be beneficial to systems of systems and warrant further investigation.

Systems Analysis Informs Systems of System Analysis

Consider how the impact of a system on other systems in any potential systems of systems might be assessed. Any system of systems analysis will require, as a starting point, knowledge of the potential risks, accidents, hazards and behaviours of the individual systems within the system of systems. The framework within this thesis provides a means of providing knowledge of product behaviours, and relating these to hazards, accidents and risks which are analysed in system safety analyses. Thus, this framework may be a starting point for system of systems analysis.

The framework described in thesis has a focus on layers of defences (refer to Sections 9.1 and 9.2, and Chapter 4). Knowledge of these defences is useful when taking a system of systems perspective for the following reasons:

- when system of systems analysis identifies new interaction related hazards, the knowledge of the individual system defences permits straightforward evaluation of substantive treatments to these hazards;
- existing system defences against faults may already provide treatments to systems of systems hazard, and thus knowing that they exist reduces need for re-engineering systems;
- existing system defences against faults may provide the architectural framework necessary to enhance a system’s defences with lesser impact on the system’s established design if the system already incorporates a fault prevention and fault tolerance defence design features; and
- external defences (i.e. external defence, platform severity reduction defence and external severity reduction defences – refer Chapter 4) may be identified by an individual system which are suggestive of dependencies in a system of systems context.

System of Systems Impact on Individual Systems

Consider how the impact of the other systems in the system of systems on the individual system might be assessed. The impact assessment on individual systems in systems of systems analysis will require knowledge of how the individual systems behaviours may be impacted by system of systems behaviours. The framework within this thesis provides a means of providing knowledge of an individual system's defences against classes of faults. This knowledge is beneficial because it permits an assessment of how any system interaction hazards, and the faults they might induce, are handled by the individual system. Further, when they are not, it provides sufficient behavioural knowledge to determine if an additional defence may or may not be required.

Application of Concepts to System of Systems

This thesis describes the 'layers of defence' concept at a system architectural level to provide confidence that faults/events don't prevent the system achieving safety. This concept offers benefits to system of systems assurance paradigms, in that it provides a way to measure the resilience of a system of systems against both individual and collective threats. Although beyond of the scope of this thesis, a valuable future research question is: can a system of systems assurance framework be developed that is based on the layers of defences against threats concept?

This thesis also describes the 'tolerability of limitations' concept to provide a way of articulating the impact of unavoidable limitations in evidence. In the system of systems case, having knowledge of the behaviours of all the systems in the network is much less likely than having knowledge about systems that are within your control, and less knowledge about systems that are out of it. Thus the tolerability of limitations concept might also offer some benefits to system of systems assurance paradigms, in that when coupled with the layers of defence concept, it provides a methodology for determining the importance of evidence shortfalls, and correlating them to known defences. This may provide a means of establishing priorities in a system of systems where it is most valuable to seek further knowledge.

Thus it is evident that the framework provided within this thesis could help to complement system of systems analysis, and warrants investigation.

11.3 Concluding Remarks

A major factor identified in the effectiveness of safety assurance standards is how stakeholders are incentivised (or discouraged) in decision making. This thesis has shown the novel integrated framework, through implementation of the principles and guidelines, helps avoid historical project issues by helping developers to focus on reasoning about the risks related to behavioural properties of their products, and the production of evidence informing product behaviours.

Only with practical application on real world projects can the feasibility and effectiveness of these concepts be fully evaluated. The approaches described in this thesis help to improve the demonstration and achievement of safety, and thus through their practical use, it is hoped, will provide greater confidence in the knowledge and uncertainty associated with the treatment and retention of risks. It has not been possible to completely demonstrate this within the timescale of a Doctoral program, but the aspiration is that the definition of concepts within this thesis will lead to industrial application, and improvements in the safety assurance of aviation systems.

Appendix A – Fault Prevention and Tolerance in Aviation Systems

This appendix summarises the fault prevention and fault tolerance mechanisms in several aircraft that were examined in establishing architectural trends in safety assurance (refer to Chapter 4).

Fault Prevention and Fault Tolerance in Flight Control Systems

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-17A
Primary Computers	Three (3) Primary Flight Computers (PFC) – digital • process Normal, Secondary and Direct laws • actuation commands transmitted to ACE • execution of automated functions such as yaw damper • system monitoring, crew annunciation, and on-board maintenance capabilities Four (4) Actuator Control Electronics (ACE) – analogue • interface with the pilot control transducers and to control the Primary Flight Control System actuation with analogue servo loops • ACEs convert the transducer position into a digital value and then transmit that value to the PFCs • ACEs then convert PFC commands into analogue commands for each individual actuator • flight control surface servo loops are distributed among the four ACEs.	Three (3) Flight Control Primary Computers (FCPC) - digital • process Normal, Alternate and Direct Laws • one FCPC is selected as Master: it processes the orders and outputs them to the other computers which will execute them on their servo loops • Master checks that its orders are fulfilled by comparing them with feedback received; self-monitoring of the master can detect a malfunction and cascade control to the next computer • each FCPC can control up to eight (8) servo loops and provide complete aircraft control under normal laws.	Quad redundant digital flight control system incorporating two (2) flight control computers with two independent channels per computer process: • control stick, rudder pedal and trim commands • pitot static, rate gyro, accelerometer, AOA probe and fight control surface position feedback signals, and • send commands to each flight control surface actuator.	Quadruplex set of digital flight control computers Four channel synchronous operation All output hardware, signals, and feedback are monitored and compared to ensure failure detection and channel output voting. Dedicated cross-channel data link used between channels Input signal voting Actuator loop voting
Secondary Computers		Two (2) Flight Control Secondary Computers (FCSC) - digital • Are able to process direct laws • Either secondary can be the master in the case of loss of all FCPC • Each FCSC can control up to 10 servo loops and can provide complete aircraft control		

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-17A
Additional Control Computers	Other systems: • Flap Slat Electronics Unit (FSEU) • Proximity Switch Electronics Unit (PSEU) • Engine Data Interface Unit (EDIU) Airplane Information Management System (AIMS) Data Conversion Gateway (DCG) maintains separation between the critical flight controls busses and the essential systems busses.	High life devices are commanded by two Slat/Flap Control Computers Two (2) Flight Control Data Concentrators (FCDC) acquire the outputs from the various computers to be sent to the ECAM and Flight Data Interface Unit to provide isolation of the flight control computers from other systems.	No additional flight computers	Two dual digital Spoiler Control/Electronic Flap Computers
Computer Architecture	Each PFC has three identical computing ‘‘lanes” • a voting plane scheme is used by the PFCs on themselves, • single computing lane within a PFC channel is declared as the ‘‘master” lane, • all three lanes simultaneously computing the same control laws, • the outputs of all three lanes are compared against each other, • any failure of a lane that will cause an erroneous output from that lane will cause that lane to be ‘‘failed” by the other two lanes, and • Command Lane, Standby Lane, Monitor Lane.	Command/ Monitor computer architecture for both the FCPC and FCSC. • Monitor channel monitors for health of the command channel and control surface runaway • Specific variables are permanently compared in the two channels. • sensor inaccuracy, rigging tolerances, computer asynchronisation are taken into account • errors which are not detectable (within the signal and timing thresholds) are assessed in respect to their handling quality and structural loads effect • in the event of a divergence between command and monitor solutions, the affected computer is disengaged and the next highest priority computer takes over	Two independent channels per computer processor	FCC and SCEFC each use 3 MIL-STD-1750A CPUs. • In the FCC one processor serves as an I/O processor, and the other two perform control law computations. • In the SFEFC one processor serves as an I/O processor and the other two are configured as a self-checking pair. The AFCS control panel is implemented with four MIL-STD-1760A CPU configured as two self-checking pairs.
Dissimilarity	Dissimilarity between the PFC and ACE. • PFCs are identical digital computers • ACE are identical analogue devices Dissimilarity between the Air Data and Inertial Reference Unit (ADIRU) and Standby Attitude and Air Data Reference Unit (SAARU)	Dissimilarity between FCPC and FCSC digital computer designs • different processor architectures and manufacturers • different software between FCPC and FCSC and between command and monitor channels in each FCPC and FCSC No dissimilarity between Air Data and Inertial Reference Units (ADIRUs)	No dissimilarity between flight control computers.	No dissimilarity between flight control computers.

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-17A
Latent Failure Detection	Built in Test	Self-test and peripheral tests	Built in Test	Built in Test
Reconfiguration	The outputs from all three PFC channels are compared. • Each PFC compares its output for each particular actuator, and with the same command that was calculated by the other two PFC channels. • Each PFC channel does a mid-value select on the three commands, and that value is output to the ACEs.	When the active computer interrupts its operation, one of the standby computers almost instantly changes to active mode with no or limited jerk on the control surfaces.	Three (3) modes of operation • Control Augmentation System (CAS) – full digital capability including adaptive flight controls and stability augmentation. • Direct Electrical Link (DEL) – provided in the event of primary CAS failure, no longer process input from failed rate gyros and/or accelerometers. • Mechanical (MECH) – three or more channel failures, pitch roll sensor failures, failure of both servo-valves in one actuator, hydraulic starvation	All FCS critical inputs, processing and outputs are quad redundant (fail op, fail op, fail passive). The FCCs and SCEFCs operate as a frame synchronous set. In the event of loss of synchronisation, the computers will attempt to re-synchronise. A sensor selection algorithm derives a selected value for each signal as a function of the sensor failure states. • Average of middle two values (four valid signals) • Mid-value of three signals • Average of two signals
Servos	Actuators arrangements are as follows: • Elevators, ailerons, and flaperons are controlled by two actuators per surface; the rudder is controlled by three. • Each spoiler panel is powered by a single actuator. • The horizontal stabilizer is positioned by two parallel hydraulic motors driving the stabilizer jackscrew. The actuation powering the elevators, ailerons, flaperons, and rudder have several operational modes: Active, Bypassed, Damped, or Blocked.	Actuators arrangements are as follows: • Elevators, ailerons are controlled by two actuators per surface; the rudder is controlled by three. • Each spoiler panel is powered by a single actuator. • The horizontal stabiliser is positioned by two actuators. Servo-jacks can operate in one of three control modes depending upon computer status and type of control surface: active, damping, centring. Normally one servo is active and one is damped on each control surface	Dual servo values in each actuator fed by both flight control computers and two independent hydraulic sources. Aileron and twin rudders are differentially scheduled. Trailing Edge Flaps, Leading Edge Flaps and Stabs are scheduled both differentially and collectively.	All four FCCs are connected to each actuator. Outputs from each FCC are summed at the Electro hydraulic Servo Values providing a voting node. Output signal management software function in each FCC compares local channel actuator data with cross channel data to detect, identify and remove local faults

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-17A
Envelope protection / limiting	Computers provide the following protections: • Bank angle protection • Turn compensation • Stall and over-speed protection • Pitch control and stability augmentation • Thrust asymmetry compensation	Computers will prevent excessive manoeuvres and exceedance of the safe flight envelope. • Excessive load factors • Over-speed • Stall • Extreme pitch angle • Extreme bank angle	Conventional envelope protections not provided in a fighter jet	The following protections are provided: • Angel of attack limiting system • Deep stall avoidance • All engine out control • Safe go-around
Sensors	Dual redundant air data and inertial systems: • Air Data and Inertial Reference Unit (ADIRU) • Standby Attitude and Air Data Reference Unit (SAARU) • Autopilot Flight Director Computers (AFDC) All critical interfaces into the Primary Flight Control System use multiple inputs which are compared by a voting plane.	Triple redundant air data and inertial: • Three air data and inertial reference units (ADIRUs) • Accelerometers and rate gyros	The following sensors are used by the Flight Control Computers: • Pitot Static • Rate Gyro • Accelerometer, • AOA probe • And flight control surface position transducers	Quadruplex sensors including: • Stick and Pedal Force Sensors • Stick Position • Surface Position • Air data and stabiliser sensors • Air Data Computers • Inertial Reference Unit Six (6) AOA sensors Remaining sensors are dual. Inputs are voted, monitored, selected and sent to each processing channel before use in output signal processing. Digital inputs have validity bits
Mechanical Backup	Two spoiler panels and alternate stabiliser pitch trim are mechanically controlled	Mechanical backup: rudder and trimmable horizontal stabiliser – no artificial stabilisation required	Mechanical backup/linkage to the horizontal stabilators.	Backup mechanical system provides control of the ailerons, elevators, rudders and stabilizer surfaces.

Table 34: Summary of Fault Prevention and Fault Tolerance in Flight Control Systems

Fault Prevention and Fault Tolerance in Flight Management Systems

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-130J
Flight Management Computers	Dual integrated cabinets which provide the processing and the I/O hardware and software required to perform the following functions: • Flight Management • Display • Central Maintenance • Airplane Condition Monitoring • Communication Management (including flight deck communication) • Data Conversion Gateway (ARINC 429/629 Conversion) The applications hosted on AIMS are listed below, along with the number of redundant copies of each application per ship-set in parentheses: • Displays (4) • Flight Management/Thrust Management (2) • Central Maintenance (2) • Data Communication Management (2) • Flight Deck Communication (2) • Airplane Condition Monitoring (1) • Digital Flight Data Acquisition (2) • Data Conversion Gateway (4)	Two computers Flight Management Guidance Computer (FMGC) • Flight management for navigation, performance prediction and optimisation, navigation radio tuning and information display management • Flight guidance for autopilot commands, flight director and thrust commands – two types of guidance o Managed – lateral and vertical flight plan data o Selected – guidance targets selected on the glare-shield Flight Control Unit • Flight envelope and speed computation	Two AYK-14 Mission Computers (MC) One MC will be the active Bus Controller on AVMUX 1 – 6 and the other MC will be the an RT.	Communication / Navigation / Identification – Management System (CNI-MS) consists of • 2 Mission Computers (MC) – control the information exchanged between airplane systems via MIL-STD-1553 data-buses. One MC will be the active Bus Controller on nominated data-buses and the other will be the Backup Bus Controller for those same data-buses • 2 Bus Interface Units (BIU) – if both MCs fail the BIU assume the bus controller functions for the applicable data-buses • 2 CNI System Processors (CNI-SP) – contain the operational logic that permit crew control and functioning of the communication, navigation and identification equipment.
Control	The other flight deck hardware elements that make up the AIMS system are • Six flat panel display units • Three control and display units (left, centre and right) • Two EFIS display control panels • Display select panel • Cursor control devices • Display remote light sensors	Three Multipurpose Control and Display Units (MCDU) (only two at a time) provide: • flight plan definition and display • data insertion (speeds, weights, cruise level, etc.) • selection of specific functions One Flight Control Unit on the glare-shield provides manual entry of: • speed • heading • altitude • vertical speed Two thrust levers linked to the FMGCs and FADECs provide auto-thrust or manual thrust control	Left and Right Digital Display Indicators (DDIs) Up Front Controller (UFC) Digital Map Computer (DMC)	• 3 CNI Management Units (CNI-MU) – primary crew interface to the CNI-MS. • 1 Communications / Navigation / Breaker Panel (CNBP) • 2 Avionics Management Units (AMU) • 2 Heads Up Display (HUD)

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-130J
Display		Two Primary Flight Displays (PFD) and two Navigation Displays (NDs) provide visual interface with flight management and guidance related data. PFD: • FMGC guidance targets • Armed and active modes • System engagement targets ND: • Flight plan presentation • Aircraft position and flight path • Navigation items including radio aids and wind)	Left and Right DDIs UFC DMC Heads Up Display (HUD)	
Computer Architecture	Dual cabinets each contain four core processor modules (CPMs) and four input / output modules (IOMs), with space reserved in the cabinet to add one CPM and two IOMs to accommodate future growth. The shared platform resources provided by AIMS are • Common processor and mechanical housing, • Common input/output ports, power supply, and mechanical housing, • Common backplane bus (SAFEbus™) to move data between CPMs and between CPMs and IOMs, • Common operating system and built-in test (BIT) and utility software. Applications are integrated on common CPMs. The IOMs transmit data from the CPMs to other systems on the airplane, and receive data from these other systems for use by the CPM applications. A high-speed backplane bus, called SAFEbus™, provides a 60-Mbit/s data pipe between any of the CPMs and IOMs in a cabinet. Communication between AIMS cabinets is through four ARINC 629 serial buses.	Two computers Flight Management Guidance Computer (FMGC) FMGC are identical single channel computers MCDU are identical single channel computers	Two AYK-14 Mission Computers AYK-14 MCs are identical single channel computers. Other AYK-14 modules include a core memory and MIL-STD - 1553A/B, Tactical Data System, RS-232, and discrete Input / Output (I/O).	MC are identical single channel computers CNI-SP are identical single channel computers BIU are identical single channel computers MC, CNI-SP and BIU are all different computer architectures
Dissimilarity	No dissimilarity between AIMS cabinets	No dissimilarity between FGMCs No dissimilarity between MCDUs	No dissimilarity between MCs	No dissimilarity between MCs No dissimilarity between CNI-SPs No dissimilarity between BIUs
Latent Failure Detection	Built in Test	Built in Test	Built in Test	MC BIT CNI-SP PBIT and IBIT

Feature	Boeing 777	A330 / KC-30A	F/A-18A/B	C-130J
Reconfiguration	Hardware fault detection and isolation is achieved via a lock-step design of the CPMs, IOMs, and the SAFEbus™. Each machine cycle on the CPMs and IOMs is performed in lock-step by two separate processing channels, and comparison hardware ensures that each channel is performing identically. If a miss-compare occurs, the system will attempt retries where possible before invoking the fault handling and logging software in the operation system. The SAFEbus™ has four redundant data channels that are compared in real time to detect and isolate bus faults.	Selected guidance has priority over managed guidance mode. Normal mode, dual mode, single mode	One MC will be the active Bus Controller on AVMUX 1 – 6 and the other MC will be the RT. If the MC BC fails, the other MC assumed control of all buses.	One MC is capable of performing the functions of both MCs with no reduction in capability If one MC fails, the other MC assumes control of all seven buses with no loss of system integration performance. BIU provides backup in the event of dual MC failures. Each CNI-SP calculates its own solutions independently, and compares the results with the other CNI-SP. Either CNI-SP can perform all the functions alone, should the other CNI-SP fail. The CNI-SP operates in one of three modes: dual, single active/inactive and independent.
Sensors	Redundant Inertial Navigation Systems / Global Positioning Systems Radio Navigation (VOR, ILS, ADF, DME)	Each FMGC tunes its own side except when in single operation • One VOR • One ILS • One ADF • 5 DMEs 3 Inertial Reference Systems FMGC position is a blend of IRS and radio position Uses GPIRS position in priority mode	Comm #1 and Comm #2 (UHF, VHF, HF) EGI – INS / GPS VOR, ILS, TACAN , DME, ADF Combined Interrogator Transponder (CIT)	The CNI-MS controls the following equipment: • 2 UHF radios • 2 VHF radios • 2 HF radios • 2 Embedded GPS/INS (EGI) • 2 VOR/ILS/MB radios • 2 TACAN radios • 2 ADF radios • 2 IFF transponders
Backup	Stand-by navigation instruments Communications can be independently tuned	Stand-by navigation instruments Communications can be independently tuned	Stand-by navigation instruments Communications can be independently tuned	Stand-by navigation instruments CNBP can independently tune radios Bus Interface Unit (BIU) provides backup in the event of MC failures

Table 35: Summary of Fault Prevention and Fault Tolerance in Flight Management Systems

Appendix B – Attributes of Lifecycle Products

This appendix lists in full the attributes of each respective lifecycle products defined by the CSAL framework (refer Chapter 5). Greyed-out attributes are not applicable the specific lifecycle product.

Attributes of Specified Constraint Level Requirements

(specified at the level of the architectural constraint, and at the level at which requirements are allocated to hardware and software)

Attributes	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Existence Group					
Defined / Developed / Produced / Integrated	Specified Constraint Level Requirements for constraint {constraint} do not exist – therefore there is no basis for the relevant behaviour existing.	Intolerable	Intolerable	Intolerable	Tolerable
Specification Group					
Accurate / Consistent / Complete	The Specified Constraint Level Requirements isn't accurate / consistent / complete – therefore, there is potential for other lifecycle products or translations to refine or implement the behaviour erroneously.	Intolerable	Constrained	Constrained	Tolerable
Unambiguous / Precise	The Specified Constraint Level Requirements is ambiguous and/or imprecise – therefore, there is potential for other lifecycle products or translations to misinterpret the constraint.	Intolerable	Constrained	Constrained	Tolerable
Verifiable	The Specified Constraint Level Requirements cannot be verified (analytically or empirically) – therefore verification evidence for the constraint may not exist or may be irrelevant.	Intolerable	Constrained	Constrained	Tolerable
Validatable	The Specified Constraint Level Requirements cannot be validated (analytically or empirically) – therefore validation evidence for the constraint may not exist or may be irrelevant.	Intolerable	Constrained	Constrained	Tolerable
Traceable to Higher					
Traceable to Lower	The Specified Constraint Level Requirements has no traceability to a lower level refinement of the behaviour – therefore, there is no traceable basis for the refinement of the relevant Specified Constraint Level Requirements existing in the design	Intolerable	Intolerable	Tolerable	Tolerable

Attributes	**Impact of NOT Satisfying**	**Tolerability of Limitations in Satisfying**			
		CSAL3	**CSAL2**	**CSAL1**	**CSAL0**
Compatible with Target	The Specified Constraint Level Requirements isn't compatible with the target implementation – therefore, the specification of the constraint is unverifiable and additional behaviours that violate the constraint may be exhibited by the target.	Intolerable	Constrained	Constrained	Tolerable
Verification Group					
Coverage of Self	The Specified Constraint Level Requirement hasn't been covered by verification – therefore the specification of the constraint hasn't been verified.	Intolerable	Intolerable	Intolerable	Tolerable
Compliant with Higher					
Robust with Higher					
Coverage of Higher					
Validation Group					
Consistent with Safety Objective	The Specified Constraint Level Requirement isn't consistent with safety objectives – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Consistent with Other	The Specified Constraint Level Requirement isn't consistent with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Non-Interference with Other	The Specified Constraint Level Requirement infers with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Resolution of Inadequacies					
Inadequacies in Specified Constraint Level Requirements are identified and resolved	Compliance, robustness, traceability, verification and validation may identify inadequacies in Specified Constraint Level Requirements – therefore the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Constrained	Tolerable

Table 36: Attributes of Specified Constraint Level Requirements

Attributes of Refined Abstract Level Requirements

(optional, and refined from Specified Constraint Level Requirements, while still being abstract from Low Level Requirements, and used to provide a means making claims from evidence that cannot be produced directly against Specified Constraint Level Requirements or Low Level Requirements)

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Existence Group					
Defined / Developed / Produced / Integrated	Refined Abstract Level Requirements for constraint {constraint} do not exist – therefore there is no basis for the relevant behaviour existing.	Intolerable	Intolerable	Tolerable	Tolerable
Specification Group					
Accurate / Consistent / Complete	Refined Abstract Level Requirement for constraint {constraint} isn't accurate / consistent / complete – therefore, there is potential for other lifecycle products to refine or implement the behaviour erroneously.	Intolerable	Constrained	Tolerable	Tolerable
Unambiguous / Precise	Refined Abstract Level Requirements for constraint {constraint} are ambiguous and/or imprecise – therefore, there is potential for other lifecycle products or translations to misinterpret the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Verifiable	Refined Abstract Level Requirement for constraint {constraint} cannot be verified (analytically or empirically) – therefore verification evidence for the constraint will not exist or be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Validatable	Refined Abstract Level Requirements for constraint {constraint} cannot be validated (analytically or empirically) – therefore validation evidence for the constraint may not exist for me irrelevant.	Intolerable	Constrained	Tolerable	Tolerable
Traceable to Higher	Refined Abstract Level Requirements for constraint {constraint} aren't traceability to the higher level Requirements associated with the constraint {constraint} – therefore, the behaviours specified by this Refined Abstract Level Requirements may not be consistent with the constraint	Intolerable	Intolerable	Tolerable	Tolerable
Traceable to Lower	Refined Abstract Level Requirements for the constraint {constraint} aren't traceability to a lower level refinement of the behaviour – therefore, there is no basis for the refinement of the relevant Abstract Level Requirement existing in the design	Intolerable	Intolerable	Tolerable	Tolerable

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Compatible with Target	Refined Abstract Level Requirements for the constraint {constraint} aren't compatible with the target implementation – therefore, the specification of the constraint is unverifiable and additional behaviours that violate the constraint may be exhibited by the target.	Intolerable	Constrained	Tolerable	Tolerable
Verification Group					
Coverage of Self	Refined Abstract Requirements for constraint {constraint} haven't been covered by verification – therefore the requirement hasn't been verified.	Intolerable	Intolerable	Tolerable	Tolerable
Compliant with Higher	Refined Abstract Requirement for constraint {constraint} aren't compliant with the Higher Level Requirements – therefore, the behaviours specified by the Refined Abstract Level Requirements are not consistent with the constraint	Intolerable	Constrained	Tolerable	Tolerable
Robust with Higher	Refined Abstract Requirement for constraint {constraint} aren't robust with the Higher Level Requirements – therefore, the behaviours specified by the Refined Abstract Level Requirements may not be resilient to faults that might violate the constraint	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Higher	Refined Abstract Requirements for constraint {constraint} aren't verified against all related Specified Constraint Level Requirements – therefore, the behaviours implemented may not be consistent with the constraint	Intolerable	Intolerable	Tolerable	Tolerable
Validation Group					
Consistent with Safety Objective	Refined Abstract Requirements for constraint {constraint} aren't consistent with safety objectives – therefore the constraint may be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Consistent with Other	Refined Abstract Requirements for constraint {constraint} aren't consistent with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Non-Interference with Other	Refined Abstract Requirements for constraint {constraint} interferes with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Inadequacies Resolved					
Inadequacies in Refined Abstract Level Requirements are identified and resolved	Compliance, robustness, traceability, verification, and validation may identify inadequacies in Refined Abstract Level Requirements – therefore the behaviours implemented may not be consistent with the constraint.	Intolerable	Constrained	Tolerable	Tolerable

Table 37: Attributes of Refined Abstract Level Requirements

Attributes of Low Level / Detailed Design Requirements

(specified such that no additional refinement is required for logical implementation and all behaviours of the implementation are described by requirements)

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Existence Group					
Defined / Developed / Produced / Integrated	Low Level Requirements for constraint {constraint} do not exist – therefore there is no basis for relevant behaviour existing in software	Intolerable	Intolerable	Tolerable	Tolerable
Specification Group					
Accurate / Consistent / Complete	Low Level Requirements for constraint {constraint} aren't accurate / consistent / complete – therefore, there is potential for other lifecycle products to refine or implement the behaviour erroneously	Intolerable	Constrained	Tolerable	Tolerable
Unambiguous / Precise	Low Level Requirements for constraint {constraint} are ambiguous or imprecise – therefore, there is potential for other lifecycle products to misinterpret the constraint	Intolerable	Constrained	Tolerable	Tolerable
Verifiable	Low Level Requirements for constraint {constraint} cannot be verified (analytically or empirically) – therefore verification evidence for the refinement of the constraint will not exist or be invalid	Intolerable	Constrained	Tolerable	Tolerable
Validatable	Low Level Requirements for constraint {constraint} cannot be validated (analytically or empirically) – therefore validation evidence for the refinement of the constraint will not exist or be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Traceable to Higher	Low Level Requirements for constraint {constraint} aren't traceability to higher level requirements – therefore, behaviours specified by Low Level Requirements may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable
Traceable to Lower	Low Level Requirements for constraint {constraint} aren't traceability to an implementation level refinement of the behaviour – therefore, there is no basis for the refinement of the Low Level Requirements existing in the implementation.	Intolerable	Intolerable	Tolerable	Tolerable
Compatible with Target	Low Level Requirements for constraint {constraint} aren't compatible with the target computer – therefore specification of the constraint is unverifiable and behaviours that violate the constraint may be exhibited by the target.	Intolerable	Constrained	Tolerable	Tolerable

Attribute	**Impact of NOT Satisfying**	**Tolerability of Limitations in Satisfying**			
		CSAL3	**CSAL2**	**CSAL1**	**CSAL0**
Verification Group					
Coverage of Self	Low Level Requirements for constraint {constraint} haven't been covered by verification – therefore the requirements haven't been verified.	Intolerable	Intolerable	Tolerable	Tolerable
Compliant with Higher	Low Level Requirements for constraint {constraint} aren't compliant with the higher level requirements – therefore, the behaviours specified by the Low Level Requirements are not consistent with the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Robust with Higher	Low Level Requirements for constraint {constraint} aren't robust with the higher level requirements – therefore, the behaviours specified by the Low Level Requirements may not be resilient to sources of faults that might violate the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Higher	Low Level Requirements for constraint {constraint} aren't verified against all related higher level requirements – therefore, the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable
Validation Group					
Consistent with Safety Objective	Low Level Requirements for constraint {constraint} isn't consistent with safety objectives – therefore the constraint may be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Consistent with Other	Low Level Requirements for constraint {constraint} isn't consistent with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Tolerable	Tolerable
Non-Interference with Other	Low Level Requirements for constraint {constraint} interferes with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Inadequacies Resolved					
Inadequacies in Low Level Requirements are identified and resolved	Compliance, robustness, traceability, verification and validation may identify inadequacies in Low Level Requirements – therefore the behaviours implemented may not be consistent with the constraint	Intolerable	Constrained	Tolerable	Tolerable

Table 38: Attributes of Low Level Requirements

Attributes of Logical Implementation (‘Human Readable’)

(parser, compiler, assembler or translation tool readable code)

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Existence Group					
Defined / Developed / Produced / Integrated	Logical Implementation for constraint {constraint} does not exist – therefore no basis for the relevant behaviour existing.	Intolerable	Intolerable	Intolerable	Tolerable
Specification Group					
Accurate / Consistent / Complete	Logical Implementation for the constraint {constraint} is incorrect – therefore, the implementation will contain an erroneous behaviour.	Intolerable	Intolerable	Constrained	Tolerable
Unambiguous / Precise	Logical Implementation for constraint {constraint} is ambiguous or imprecise – therefore, there is potential for implementation of other components or translations to misinterpret the constraint and introduce vulnerabilities that violate the constraint.	Intolerable	Intolerable	Constrained	Tolerable
Verifiable	Logical Implementation for constraint {constraint} cannot be verified (analytically or empirically) – therefore verification evidence for the implementation of the constraint may not exist or may be invalid.	Intolerable	Intolerable	Constrained	Tolerable
Validatable	Logical Implementation for constraint {constraint} cannot be validated (analytically or empirically) – therefore validation evidence for the implementation of the constraint may not exist or be invalid.	Intolerable	Intolerable	Constrained	Tolerable
Traceable to Higher *{Low Level Requirements}*	Logical Implementation isn’t traceability to the Low Level Requirements associated with the constraint {constraint} – therefore, the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable
Traceable to Lower *{Logical Implementation – Machine Readable / Physical}*	Logical Implementation for the constraint {constraint} isn’t traceability to the Machine Readable Logical Implementation or Physical Implementation – therefore, there is no basis for the complete refinement existing in the implementation.	Intolerable	Intolerable	Tolerable	Tolerable
Compatible with Target	Logical Implementation for the constraint {constraint} isn’t compatible with the target – therefore, the implementation of the constraint is invalid and additional behaviours that violate the constraint may be exhibited by the target	Intolerable	Constrained	Constrained	Tolerable

Attribute	**Impact of NOT Satisfying**	**Tolerability of Limitations in Satisfying**			
		CSAL3	**CSAL2**	**CSAL1**	**CSAL0**
Verification Group					
Coverage of Self	Logical Implementation for constraint {constraint} hasn't been covered by verification – therefore the implementation hasn't been verified.	Intolerable	Intolerable	Tolerable	Tolerable
Compliant with Low Level Requirements	Logical Implementation for constraint {constraint} isn't compliant with the Low Level Requirements – therefore, the behaviours implemented are not consistent with the constraint	Intolerable	Constrained	Tolerable	Tolerable
Robust with Low Level Requirements	Logical Implementation for constraint {constraint} isn't robust with the Low Level Requirements – therefore, the behaviours implemented may not be resilient to sources of faults that might violate the constraint	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Low Level Requirements	Logical Implementation for constraint {constraint} isn't verified against related requirements – therefore, the behaviour implemented may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable
Validation Group					
Consistent with Safety Objective	Logical Implementation for constraint {constraint} isn't consistent with safety objectives – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Consistent with Other	Logical Implementation for constraint {constraint} isn't consistent with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Non-Interference with Other	Logical Implementation for constraint {constraint} interferes with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Inadequacies Resolved					
Inadequacies in Logical Implement-ation are identified and resolved	Compliance, robustness, traceability verification, and validation may identify inadequacies in implementation – therefore the behaviours implemented may not be consistent with the constraint	Intolerable	Intolerable	Constrained	Tolerable

Table 39: Attributes of Logical Implementation ('Human Readable')

Attributes of Logical Implementation (‘Machine Readable’)

(executable object code, binary implementation, physical devices)

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Existence Group					
Defined / Developed / Produced / Integrated	Logical Implementation Machine Readable for the constraint {constraint} has not been produced or integrated – therefore no basis for the refinement of the relevant behaviours of the constraint existing in the implementation	Intolerable	Intolerable	Intolerable	Tolerable
Specification Group					
Accurate / Consistent / Complete	Logical Implementation Machine Readable for the constraint {constraint} is incorrect – therefore, the implementation will contain an erroneous behaviour.	Intolerable	Intolerable	Constrained	Tolerable
Unambiguous / Precise	Logical Implementation Machine Readable for constraint {constraint} is ambiguous or imprecise – therefore, there is potential for implementation of other components or translations to misinterpret the constraint and introduce vulnerabilities that violate the constraint.	Intolerable	Intolerable	Constrained	Tolerable
Verifiable	Logical Implementation Machine Readable for constraint {constraint} cannot be verified (analytically or empirically) – therefore verification evidence for the implementation of the constraint may not exist or may be invalid.	Intolerable	Intolerable	Constrained	Tolerable
Validatable	Logical Implementation Machine Readable for constraint {constraint} cannot be validated (analytically or empirically) – therefore validation evidence for the implementation of the constraint may not exist or be invalid.	Intolerable	Intolerable	Constrained	Tolerable
Traceable to Higher *{Logical Implementation – Human Readable}*	Logical Implementation Machine Readable for constraint {constraint} isn’t traceability to the Logical Implementation Human Readable associated with the constraint {constraint} – therefore, the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable
Traceable to Lower					

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Compatible with Target	Logical Implementation Machine Readable for the constraint {constraint} isn't compatible with the target – therefore, the implementation of the constraint is invalid and additional behaviours that violate the constraint may be exhibited by the target	Intolerable	Constrained	Constrained	Tolerable
Verification Group					
Coverage of Self	Logical Implementation Machine Readable for constraint {constraint} hasn't been covered by verification – therefore the implementation hasn't been verified.	Intolerable	Intolerable	Tolerable	Tolerable
Compliant with Specified Constraint Level Requirements	Logical Implementation Machine Readable for constraint {constraint} compliant with the Specified Constraint Level Requirements – therefore, the behaviours implemented are not consistent with the constraint.	Intolerable	Intolerable	Constrained	Tolerable
Robust with Specified Constraint Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't robust with the Specified Constraint Level Requirements – therefore, the behaviours implemented may not be resilient to sources of faults that might violate the constraint.	Intolerable	Intolerable	Constrained	Tolerable
Coverage of Specified Constraint Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't verified against all applicable Specified Constraint Level Requirements – therefore, the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Intolerable	Tolerable
Compliant with Refined Abstract Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't compliant with the Refined Abstract Level Requirements – therefore, the behaviours implemented are not consistent with the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Robust with Refined Abstract Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't robust with the Refined Abstract Level Requirements – therefore, the behaviours implemented may not be resilient to sources of faults that may violate the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Refined Abstract Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't verified against all applicable Refined Abstract Level Requirements – therefore, the behaviours implemented may not be consistent with the constraint.	Intolerable	Intolerable	Tolerable	Tolerable

Attribute	**Impact of NOT Satisfying**	**Tolerability of Limitations in Satisfying**			
		CSAL3	**CSAL2**	**CSAL1**	**CSAL0**
Compliant with Low Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't compliant with the Low Level Requirements – therefore, the behaviours implemented are not consistent with the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Robust with Low Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't robust with the Low Level Requirements – therefore, the behaviours implemented may not be resilient to sources of faults that may violate the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Low Level Requirements	Logical Implementation Machine Readable for constraint {constraint} isn't verified against all applicable Low Level Requirements – therefore, the behaviours implemented by the software may not be consistent with the constraint	Intolerable	Intolerable	Tolerable	Tolerable
Compliant with Logical Implement-ation Human Readable	Logical Implementation Machine Readable for constraint {constraint} isn't compliant with the Logical Implementation Human Readable – therefore, the behaviours implemented are not consistent with the constraint.	Intolerable	Constrained	Tolerable	Tolerable
Robust with Logical Implement-ation Human Readable	Logical Implementation Machine Readable for constraint {constraint} isn't robust with the Logical Implementation Human Readable – therefore, the behaviours implemented by the software may not be resilient to sources of faults that may violate the constraint	Intolerable	Constrained	Tolerable	Tolerable
Coverage of Logical Implement-ation Human Readable	Logical Implementation Machine Readable for constraint {constraint} hasn't exercised all behaviours of the Logical Implementation Human Readable relevant to the constraint – therefore, there may be additional behaviours of the source code which violate the constraint	Intolerable	Intolerable	Tolerable	Tolerable
Validation Group					
Consistent with Safety Objective	Logical Implementation Machine Readable for constraint {constraint} isn't consistent with safety objectives – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Consistent with Other	Logical Implementation Machine Readable for constraint {constraint} isn't consistent with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable
Non-Interference with Other	Logical Implementation Machine Readable for constraint {constraint} interferes with other constraints – therefore the constraint may be invalid.	Intolerable	Constrained	Constrained	Tolerable

Attribute	Impact of NOT Satisfying	Tolerability of Limitations in Satisfying			
		CSAL3	CSAL2	CSAL1	CSAL0
Inadequacies Resolved					
Inadequacies in Executable Object Code are identified and resolved	Compliance, robustness, traceability and verification may identify inadequacies in Logical Implementation Machine Readable – therefore the behaviours implemented by the software may not be consistent or complete with the constraint	Intolerable	Constrained	Tolerable	Tolerable

Table 40: Attributes of Logical Implementation (‘Machine Readable’)

Appendix C – Tender/Contract DRL and DIDs

This appendix provides the Data Requirements List (DRL) and Data Items Descriptions (DID)s for Tender and Contract documentation (refer to Chapter 7).

Tender DRL

TDRL#	Title
ENG-120	Safety Assurance Plan
ENG-121	Development Plan
ENG-511	Conceptual Architectural Suitability Document
ENG-521	Exemplar Safety Assurance Case
Key #: TDRL numbers integrated to (Defence Materiel Organisation , 2012) numbering sequence.	

Table 41: Tender Data Requirements List (TDRL)

Contract DRL

CDRL#	Title	Delivery Schedule	
		Drafts	**Final**
ENG-120	Safety Assurance Plan	Refer Tender	ED+20
ENG-121	Development Plan	Refer Tender	ED+20
ENG-511	Architectural Assurance Document	SRR-20 SDR-20 PDR-20 CDR-20 TRR-20	FCR-40
ENG-521	Safety Assurance Case	SRR-20 SDR-20 PDR-20 CDR-20 TRR-20	FCR-40
ENG-522	Safety Assurance Summary	SRR-20 SDR-20 PDR-20 CDR-20 TRR-20	FCR-40
CM-120	Configuration Index	TRR-20	FCR-20
Key #: TDRL numbers integrated to (Defence Materiel Organisation , 2012) numbering sequence. ED = Effective Date, SRR = Systems Requirements Review, SDR = Systems Design Review, PDR = Preliminary Design Review, CDR = Critical Design Review, TRR = Test Readiness Review, FCR = Final Certification Review Note that the lead and lag timeframes specified above are notional for illustration purposes and may be adjusted pre-contract signature by the contract preparer to suit specific project lifecycle requirements.			

Table 42: Contract Data Requirements List (CDRL)

DIDs

DID NUMBER: ENG-120

TITLE: SAFETY ASSURANCE PLAN (TENDER / CONTRACT)

DESCRIPTION AND INTENDED USE

The purpose of the Safety Assurance Plan is to describe the [Tenderer/Contractor] proposed approach to assurance of constraints, claims assurance and evidence assurance.

The [Tenderer/Contractor] uses the Safety Assurance Plan to describe how they will demonstrate tolerability of limitations in evidence for constraints.

The [Acquirer] uses the Safety Assurance Plan to evaluate if the limitations in evidence are tolerable for constraints.

INTERRELATIONSHIPS

The Safety Assurance Plan is subordinate to the following data items, where these data items are required under the tender/contract:

- Tenderer/Contractor Engineering Management Plan
- System Safety Program Plan

The Safety Assurance Plan inter-relates with the following data items, where these data items are required under the tender/contract:

- Development Plan

APPLICABLE DOCUMENTS

The following documents form a part of this DID to the extent specified herein:

- [List any [Acquirer] or [Tenderer/Contractor] specific documents that are required to form part of this DID – tender/contract specific]

PREPARATION INSTRUCTIONS

Generic Format and Content

The data item shall comply with the general format, content and preparation instructions contained in the section entitled “General Requirements for Data Items”.

Specific Content

System Overview. The [Tenderer/Contractor] shall provide a descriptive overview of the system and architecture to which this plan relates.

Assurance of Constraints

Proposal of CSAL. The [Tenderer/Contractor] shall describe the Claims Safety Assurance Level (CSAL) proposed for each constraint described by the [Conceptual Architectural Suitability Document / Architectural Assurance Document] as per [Table 19].

Assurance of Evidence

Proposal of ESAL. The [Tenderer/Contractor] shall describe the Evidence Safety Assurance Level (ESAL) proposed for each attribute for each constraint described by the [Conceptual Architectural Suitability Document / Architectural Assurance Document] as per [Table 19].

Assessing the Evidence. The [Tenderer/Contractor] shall describe how the evidence produced from the application of the [Tenderer/Contractor] proposed methods and techniques (as described by the [Tenderer/Contractor] proposed [Development Plan] are proposed to achieve the Evidence Safety Assurance Level (ESAL) requirements for tolerability of limitations as defined in [Table 24]; for each attribute of each lifecycle product [per Appendix B], at the CSAL [defined per Table 19] and as described in the [Conceptual Architectural Suitability Document / Architectural Assurance Document] for each proposed constraint type. Where the strategy for attributes is common across groups of constraint types, then the information need not be duplicated, provided there is traceability for each proposed constraint type.

Support for Certification Evaluation. The [Tenderer/Contractor] shall describe the means, either via provision of evidence or via access provisions to tenderer facilities and data, for the [Acquirer] to inspect or review all evidence, both new and existing, from the application of [Tenderer/Contractor] proposed methods and techniques for the purposes of certification evaluation by the [Acquirer].

DID NUMBER: ENG-121

TITLE: DEVELOPMENT PLAN

DESCRIPTION AND INTENDED USE

The purpose of the Development Plan is to describe the [Tenderer/Contractor] proposed development strategy and execution.

The [Tenderer/Contractor] uses the Development Plan to describe the methods and techniques proposed and their relationship to the development lifecycle.

The [Acquirer] uses the Development Plan to evaluate the suitability of development lifecycle, methods and techniques, and the suitability of sources of evidence.

INTERRELATIONSHIPS

The Development Plan is subordinate to the following data items, where these data items are required under the tender/contract:

- Tenderer/Contractor Engineering Management Plan

The Development Plan inter-relates with the following data items, where these data items are required under the tender/contract:

- Safety Assurance Plan

APPLICABLE DOCUMENTS

The following documents form a part of this DID to the extent specified herein:

- [List any [Acquirer] or [Tenderer/Contractor] specific documents that are required to form part of this DID – tender/contract specific]

PREPARATION INSTRUCTIONS

Generic Format and Content

The data item shall comply with the general format, content and preparation instructions contained in the section entitled “General Requirements for Data Items”.

Specific Content

System Overview. The [Tenderer/Contractor] shall provide a descriptive overview of the system and architecture to which this plan relates.

Overview of Required Work. The [Tenderer/Contractor] shall provide an overview of required work, including product, processes and data.

General Plans

Development Process. The [Tenderer/Contractor] shall describe the lifecycle, and the applicable lifecycle products and their hierarchy.

Standards. The [Tenderer/Contractor] shall describe the standards to be applied for certification liaison, planning, reviews and audits, configuration management, quality assurance, requirements analysis, safety and security, design development, verification and validation, corrective actions, release to service, and documentation.

Detailed Plans

Processes and Procedures. The [Tenderer/Contractor] shall describe process and procedures to be used for certification liaison, planning, reviews and audits, configuration management, quality assurance, requirements analysis, safety and security, design development, verification and validation, corrective actions, release to service, and documentation.

Methods. The [Tenderer/Contractor] shall describe the methods and techniques proposed to be used throughout the development lifecycle, including description of techniques or methods used prior to this development but for which evidence is relevant. The [Tenderer] shall describe how all evidence, both new and existing, or produced from the application of [Tenderer] proposed methods and techniques will be documented, stored, and retrievable.

CDRL Delivery. The [Tenderer/Contractor] shall describe how [CDRLs] will be produced throughout the development lifecycle, per the delivery timeframes specified at the [CDRL].

Schedule of Activities. The [Tenderer/Contractor] shall describe schedule including lifecycle processes, activities, milestones and deliverables; any dependencies between schedule elements are also to be described.

Project Organisation and Resources. The [Tenderer/Contractor] shall describe the organisational structure, including the organisations involved, their relationships, and authority and responsibilities of each organisation. The [Tenderer/Contractor] shall describe the personnel resources and facilities; as well as any acquirer-furnished equipment, services, data and facilities.

DID NUMBER: ENG-511

TITLE: CONCEPTUAL ARCHITECTURAL SUITABILITY DOCUMENT (TENDER) / ARCHITECTURAL ASSURANCE DOCUMENT (CONTRACT)

DESCRIPTION AND INTENDED USE

The purpose of the [Conceptual Architectural Suitability Document/Architectural Assurance Document] is to describe and justify the suitability of the architecture, defences, and constraints with respect the architectural assurance criteria.

The [Tenderer/Contractor] uses the document to describe and justify the suitability of the architecture, defences, and constraints with respect the architectural assurance criteria.

The [Acquirer] uses the document to evaluate the suitability of architecture, defences, and constraints with respect to architectural assurance criteria.

INTERRELATIONSHIPS

The [Conceptual Architectural Suitability Document/Architectural Assurance Document] is subordinate to the following data items, where these data items are required under the tender/contract:

- Tenderer/Contractor Engineering Management Plan
- Development Plan
- System Safety Program Plan

The [Conceptual Architectural Suitability Document/Architectural Assurance Document] inter-relates with the following data items, where these data items are required under the tender/contract:

- Requirements Data
- Design Data
- Safety Lifecycle Data
- Hazard Log

APPLICABLE DOCUMENTS

The following documents form a part of this DID to the extent specified herein:

- [List any [Acquirer] or [Tenderer/Contractor] specific documents that are required to form part of this DID – tender/contract specific]

PREPARATION INSTRUCTIONS

Generic Format and Content

The data item shall comply with the general format, content and preparation instructions contained in the section entitled “General Requirements for Data Items”.

Specific Content

System Overview. The [Tenderer/Contractor] shall provide a descriptive overview of the system and architecture to which this plan relates.

Architectural Description. The [Tenderer/Contractor] shall describe architecture of the system including:

- architectural structure including sub-systems, equipment, hardware and software;
- interfaces and communications between sub-systems, equipment, hardware and software;
- references to hazards, consequences and severities;
- sources of faults/events;
- error, fault/event, and failure mode propagation paths and transformations; and
- defences including fault/error prevention and fault tolerance including direct defences, intra-system defences, extra-system defences, external defences, severity reduction defences and external severity reduction defences.

Layers of Defence. The [Tenderer/Contractor] shall describe how the architecture and mechanisms for achieving fault prevention and fault tolerance meets the Architectural Safety Assurance Level (ASAL) requirements defined in [Table 15]. The [Tenderer/Contractor] shall provide safety risk based justification of any deviations from ASAL criteria.

Adequate Constraints. The [Tenderer/Contractor] shall describe how each constraint (i.e. absence assertion or detection and handling mechanism) is proposed to achieve the ASAL Architecturally Layered Fault Prevention and Fault Tolerance Requirements as defined in [Table 16]; or be shown to provide an equivalent level of fault prevention and fault tolerance by alternative means. The [Tenderer/Contractor] shall provide safety risk based justification of any deviations from ASAL criteria.

DID NUMBER: ENG-521

TITLE: EXEMPLAR SAFETY ASSURANCE CASE (TENDER) / SAFETY ASSURANCE CASE (CONTRACT)

DESCRIPTION AND INTENDED USE

The purpose of the [Exemplar Safety Assurance Case / Safety Assurance Case] is to describe the [Tenderer/Contractor] approach to justification of safety assurance.

The [Tenderer/Contractor] uses the [Exemplar Safety Assurance Case / Safety Assurance Case] to describe how they will justify confidence in knowledge and treatment of risks taking into account architectural, claims and evidence assurance.

The [Acquirer] uses the [Exemplar Safety Assurance Case / Safety Assurance Case] to evaluate [if the strategy for safety assurance is likely to meet / safety assurance has met] acquirer requirements for confidence in knowledge and treatment of risks.

INTERRELATIONSHIPS

The [Exemplar Safety Assurance Case / Safety Assurance Case] is subordinate to the following data items, where these data items are required under the tender/contract:

- Tenderer/Contractor Engineering Management Plan
- System Safety Program Plan

The Safety Assurance Plan [Exemplar Safety Assurance Case / Safety Assurance Case] inter-relates with the following data items, where these data items are required under the tender/contract:

- Development Plan
- Safety Assurance Plan
- [Conceptual Architectural Suitability Document / Architectural Assurance Document]
- [Safety Assurance Summary]
- Safety Lifecycle Data, Hazard Log, Safety Case

APPLICABLE DOCUMENTS

The following documents form a part of this DID to the extent specified herein:

- [List any [Acquirer] or [Tenderer/Contractor] specific documents that are required to form part of this DID – tender/contract specific]

PREPARATION INSTRUCTIONS

Generic Format and Content

The data item shall comply with the general format, content and preparation instructions contained in the section entitled “General Requirements for Data Items”.

Specific Content

System Overview. The [Tenderer/Contractor] shall provide a descriptive overview of the system and architecture to which this plan relates.

System Operation. The [Tenderer/Contractor] shall describe:

- operational, testing and maintenance activities which may be hazardous to system or personnel;
- essential safety features for operations, test and maintenance;
- anticipated operational environment from conception to disposal;
- dependencies on support facilities.

System Safety Engineering. The [Tenderer/Contractor] shall describe:

- criteria/methodologies used to classify and evaluate hazards;
- how hazards were analysed from architectural, hardware, software and human factors perspectives, including consideration of the design, operational and disposal lifecycles;
- describe and summarise the analyses, development, verification and validation performed to identify, analyse, evaluate, treat and retain risks;
- [Tender Only] The [Tenderer] shall describe the implementation of the ASAL, CSAL and ESAL framework for at least one constraint in each generalised category, type or class of constraint proposed. The [Tenderer] shall describe the set of categories, types or classes by which they have categorised the proposed constraints.
- [Contract Only] The [Contractor] shall describe how the safety objectives, and safety assurance requirements of the contract SOR have been achieved, and to provide the arguments and evidence to show the satisfaction of ASAL/CSAL/ESAL criteria for each defence/constraint.

The [Tenderer/Contractor] is free to propose how the information is presented (tabular or using an argument notation such as goal structuring notation), the emphasis on understanding how constraints will be assured and the evidence presented to demonstrate that.

Conclusions and Recommendations. The [Tenderer/Contractor] shall:

- assess the results of the safety assurance program and establish how confidence on identification, analysis, evaluation, treatment and retention of risks has been established to acquirer satisfaction; and
- provide recommendations for treatments of risks where the confidence in knowledge of risks has not been established.

DID NUMBER: ENG-522

TITLE: SAFETY ASSURANCE SUMMARY

DESCRIPTION AND INTENDED USE

The purpose of the Safety Assurance Summary is to describe and justify the tolerability of limitations of evidence with respect the claims and evidence assurance criteria.

The [Tenderer/Contractor] uses the document to describe and justify the tolerability of limitations of evidence with respect the claims and evidence assurance criteria.

The [Acquirer] uses the document to evaluate the tolerability of limitations of evidence with respect the claims and evidence assurance criteria.

INTERRELATIONSHIPS

The Safety Assurance Summary is subordinate to the following data items, where these data items are required under the tender/contract:

- Tenderer/Contractor Engineering Management Plan
- Development Plan
- System Safety Program Plan

The Safety Assurance Summary inter-relates with the following data items, where these data items are required under the tender/contract:

- Requirements Data
- Design Data
- Safety Lifecycle Data
- Hazard Log

PREPARATION INSTRUCTIONS

Generic Format and Content

The data item shall comply with the general format, content and preparation instructions contained in the section entitled “General Requirements for Data Items”.

Specific Content

System Overview

The [Tenderer/Contractor] shall provide a descriptive overview of the system and architecture to which this plan relates.

Achievement of Claims and Attributes of Software Lifecycle Products

The [Contractor] shall describe the attributes that have been assured, for each software lifecycle product, for each constraint described in the [Conceptual Architectural Suitability Document / Architectural Assurance Document]. The [Contractor] shall describe the Claims Safety Assurance Level (CSAL) established for each constraint as per [Table 20].

Assessing the Evidence

The [Contractor] shall describe how the evidence produced from the application of the [Contractor] proposed methods and techniques has assured the tolerability of limitations in evidence with respect to relevance, trustworthiness and results, for each attribute of each software lifecycle product, for each constraint described in the [Conceptual Architectural Suitability Document / Architectural Assurance Document].

The [Contractor] shall describe how the evidence produced from the application of the [Contractor] proposed methods and techniques achieves the Evidence Safety Assurance Level (ESAL) requirements for tolerability of limitations as defined in [Table 24]; for each attribute of each lifecycle product [per Appendix B], at the CSAL [Table 19] for each constraint.

Appendix D – Survey Evaluation Form and Results

Due to constraints on the size of this document, the survey evaluation form and raw results analysed in Chapter 10 have not been included within this thesis. Instead, this appendix lists the survey evaluation information provided on the enclosed CD.

- Survey Evaluation Pack
- Survey Evaluation Form
- Consolidated Survey Evaluation and Workshop Results

Appendix E – Review of Historical Projects

The following table summarises a review of numerous Australian Defence Force projects (avionics and software aspects) with respect to safety and software assurance. The review examined the project files at the Defence Materiel Organisation project offices and/or at the Directorate General Technical Airworthiness. The results are summarised to adhere to security classification and commercial restrictions.

	Paradigm			AA	Safety and Software Assurance				Risk		Safety Architecture[41]						Assurance Evidence					Contractual Impact		
Project	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
AIR 5077 (B737 AEW&C)	Acq	Yes	CC	FAA[46]	[47]	No	[48]	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes/ Lim[49]	Yes/ Lim	Yes/ Lim	Yes/ Lim	Yes/ Lim	No	Yes	Yes

[41] This column describes if evidence was found of these measures being employed. It does not indicate the overall adequacy of the measures. Where limitations are identified, this was based on there being project office documentation pertaining to the issue.

[42] ADF Design Acceptance Strategy involves recognition of prior certification by another civil or military airworthiness authority.

[43] For a commercial contract, refers to the standard identified by the SOR and SOW. For FMS, refers to the contract between the US Government and the prime US contractor.

[44] Refer to prior footnote.

[45] Refers to risk retention by an official instrument such as an Issue Paper.

[46] FAA oversight of baseline type and modifications to civil equipment.

Project	Paradigm			AA	Safety and Software Assurance				Risk		Safety Architecture[41]						Assurance Evidence					Contractual Impact		
	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
AIR 5216 (C-130J)	Acq	Yes	CC	USAF	USM	No	-	-	Yes	No	Lim	Yes	Lim	Yes	Yes	Lim	Yes	Lim	Lim	Lim	Lim	No	Yes	No
AIR 5276 (AP-3C)	Mod	-	CC	-	USM	No	-	-	Yes	No	Lim	Yes	Lim	Lim	Lim	Lim	Yes	Yes	Yes	Yes	Yes	No	Yes	No
AIR 5349 (F/A-18F)	Acq	Yes	FMS	USN	USM	No	-	-	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Lim[50]	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR 5367 (LIF HAWK 127)	Acq	Yes	CC	RAF	UKM	No	UKM	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
AIR 5376 (F/A-18 HUG)	Mod	-	FMS	USN	USM	No	-	-	No	Yes	Lim	Yes	Yes	Yes	Yes	Yes	Yes	Lim[51]	Lim	Yes	Lim	No	Yes	Yes

[47] ARP4754 for baseline type and modifications to civil certified equipment, MIL-STD-882C for military modifications.

[48] RTCA/DO-178B for baseline type and modification to civil certified equipment, no software assurance standard for military modifications

[49] Limitations are with respect to military capability systems, and this applies to requirements, design, implementation and verification evidence columns also.

[50] ITAR restrictions prevented ADF review of requirements, design, implementation and verification evidence.

[51] ITAR restrictions prevented ADF review of requirements, design, implementation and verification evidence.

Project	Paradigm			AA	Safety and Software Assurance				Risk		Safety Architecture[41]						Assurance Evidence					Contractual Impact		
	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
AIR 5391 (F-111 Interim EWSP)	Mod	-	CC	-	USM	No	-	-	Yes	Yes	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR 5398 (F-111 AGM-142)	Mod	-	CC/ FMS	USAF[52]	USM	No	-	-	Yes	Yes	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR 5402 (KC-30A)	Acq	Yes	CC	EASA[53] / CASA[54]	CiSS	No	CiS	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes/ Lim[55]	Yes/ Lim	Yes/ Lim	Yes/ Lim	No	Yes	Yes
AIR 5418 (F/A-18 FOSOW)	Mod	-	CC/ FMS	USAF[56]	USM	No	-	-	Yes	Yes	Lim	Yes	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Yes	Yes	Yes

[52] Weapon acquisition only.

[53] EASA certification of the baseline aircraft type.

[54] CASA (Spanish) oversight of the military modifications.

[55] Limitations are with respect to on and off-board mission/flight planning system, and this applies to requirements, design, implementation and verification evidence.

[56] Weapon acquisition only.

	Paradigm			**AA**	**Safety and Software Assurance**				**Risk**		**Safety Architecture**[41]						**Assurance Evidence**					**Contractual Impact**		
Project	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
AIR5440 (C-130J Block 5.4)	Mod	-	CC	USAF	USM	No	-	-	Yes	Yes	Lim	Yes	Lim	Yes	Yes	Lim	Yes	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR5440 (C-130J Block 6.1)	Mod	-	CC	USAF / RAF	USM	No	-	-	Yes	Yes	Lim	Yes	Lim	Yes	Yes	Lim	Yes	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR5440 (C-130J Block 7.0)	Mod	-	FMS	USAF / RAF	USM	No	-	-	Yes[57]	Yes[58]	Lim	Yes	Lim	Yes	Yes	Lim	Yes	Lim	Lim	Lim	Lim	No	Yes	Yes
AIR 8000 (C-17)	Acq	No	FMS	USAF	USM	No	-	-	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
AIR 87 (ARH Tiger)	Acq	Yes	CC	DGA	CiSS	No	CiS	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
AIR 9000 (MRH-90)	Acq	Yes	CC	DGA	CiSS	No	CiS	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes

[57] Program not completed at time of thesis, but risk retention is likely.
[58] Program not completed at time of thesis, but risk retention is likely.

	Paradigm			AA	Safety and Software Assurance				Risk		Safety Architecture[41]						Assurance Evidence					Contractual Impact		
Project	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
AIR 9000 (CH-47D T55 Engine)	Mod	-	FMS	US Army	USM	No	-	-	Yes	Yes	No	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	No	Yes	Yes
JP129 (I-VIEW TUAV)	Acq	Yes	CC	-	USM	No	-	-	Yes[59]	Yes[60]	Yes	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Yes	Yes	Yes
MIS910 (PC-9/A EFIS-GPS Upgrade)	Mod	-	CC	FOCA	CiSS	No	CiS	Yes	Yes	No	Yes	Yes	Lim	Yes	Lim	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
SEA 1405 (S-70B-2 FLIR and ESM)	Mod	-	CC	-	USM	Yes	-	-	Yes	Yes	No	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	No	Yes	Yes
SEA 1411 (SH-2G (A))	Acq	Yes	CC	-	-	No	-	-	Yes[61]	Yes[62]	No	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Yes	Yes	Yes

[59] Program never released to service.
[60] Program never released to service.
[61] Program never released to service.

Project	Paradigm			AA	Safety and Software Assurance				Risk		Safety Architecture[41]						Assurance Evidence					Contractual Impact		
	New Aircraft Acquisition / Modification	Acquisition includes modification of existing type	Prime Commercial Contract / Foreign Military Sales	Airworthiness Authority Involvement[42]	Contracted[43] Safety Standard	Safety Argument Notation used to express Safety Case	Contracted[44] Software Assurance Standard	Compliance with Software Assurance Standard	Risk retention (elevated risks based on product issues)[45]	Risk retention (elevated risks based on uncertainty)	Aircraft SOR requires Fail Safe Design	Fault Prevention employed	Fault Tolerance employed	System Level Fault Tolerance	LRU Level Fault Tolerance	Software Level Fault Tolerance	Requirements evidence	Design evidence	Implementation evidence	Verification of requirements	Verification of implementation	Cost/Schedule impact due to safety architecture limitations	Cost/Schedule impact due to product limitations	Cost/Schedule impact due to safety evidence limitations
S-70A-9 and CH-47D GPS	Mod	-	SPO	-	-	No	-	-	Yes	Yes	No	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	Lim	No	Yes	Yes

Key to annotations:
Acq – New Aircraft Acquisition
Mod – Modification to existing ADF Type
CC – Commercial Contract
FMS – Foreign Military Sales

SPO – System Program Office conducts integration

CASA – Spanish Construcciones Aeronauticas, S.A.
CiSS – Civil Paradigm Safety Assurance Standards – ARP 4754 and ARP 4761
CiS – Civil Paradigm Software Assurance Standards – RTCA/DO-178A/B
DGA – French General Directorate for Armament
EASA – European Aviation Safety Agency

FAA – Federal Aviation Administration
FOCA – Swiss Federal Office of Civil Aviation
RAF – Royal Air Force
USAF – United States Air Force
USM – United States Military Safety Assurance Standards – MIL-STD-882C/D
UKM – United Kingdom Safety Assurance and /or Software Assurance Standard – DefStan 00-56 Iss 2 or DefStan 00-55 Iss 2
USN – United States Navy

Lim – Limitations of achievement thereof. Note to be a limitation, the issue must feature in project office documentation and have been subject to decision on treatment or otherwise.

Table 43: Review of Historical Projects

[62] Program never released to service.

Glossary

AA	Airworthiness Authority
AC	Advisory Circular (FAA)
ACE	Actuator Control Electronics
ACP	Assurance Claim Point
ACQ	Acquisition
ADF	Australian Defence Force
ADF	Automatic Direction Finding
ADIRU	Air Data and Inertial Reference Unit
AEL	Assurance Evidence Level
AEW&C	Airborne Early Warning and Control
AFCS	Automatic Flight Control System
AFDC	Autopilot Flight Director Computers
AGM	Air to Ground Missile
AIMS	Airplane Information Management System
ALARP	As Low As Reasonably Practicable
AMC	Acceptable Means of Compliance (EASA)
AMU	Avionics Management Unit
AOA	Angle of Attack
ARH	Armed Reconnaissance Helicopter
ARINC	Aeronautical Radio, Incorporated
ARM	Argumentation Meta-model
ASAL	Architectural Safety Assurance Level
ASIL	Automotive Safety Integrity Level
AVMUX	Avionic Multiplex Bus
BBN	Bayesian Belief Network
BIT	Built In Test

BITAR	Buying Information To Avoid Risk
BIU	Bus Interface Unit
CAA	Civil Aviation Authority (United Kingdom)
CAE	Claims-Argument-Evidence (Adelard)
CAS	Control Augmentation System
CASA	Civil Aviation Safety Authority (Australia)
CASA	Spanish Construcciones Aeronauticas, S.A.
CASR	Civil Aviation Safety Regulation
CC	Commercial Contract
CD	Compact Disc
CDRL	Contract Data Requirements List
CIT	Combined Interrogator Transponder
CSAL	Claims Safety Assurance Level
CM	Configuration Management
CNBP	Communication Navigation Breaker Panel
CNI-MS	Communication Navigation Identification – Management System
CNI-SP	Communication Navigation Identification – System Processor
CPM	Core Processor Modules
CPU	Central Processing Unit
CS	Certification Specification
DAL	Design Assurance Level
DCG	Data Conversion Gateway
DDI	Digital Display Indicator
DEFSTAN	Defence Standard (United Kingdom)
DEL	Direct Electrical Link
DGA	General Directorate for Armament (French)
DID	Data Item Description

DoD	Department of Defense (United States of America)
DMC	Digital Map Computer
DME	Distance Measuring Equipment
DMO	Defence Materiel Organisation (Australian Department of Defence)
EAL	Evaluation Assurance Level
EASA	European Aviation Safety Agency
ECAM	Electronic Centralised Aircraft Monitor
EDIU	Engine Data Interface Unit
EEPE	Electrical/Electronic/Programmable Electronic
EGI	Embedded GPS / INS
ENG	Engineering (in relation to Data Item Descriptions)
EOI	Expression of Interest
ESAL	Evidence Safety Assurance Level
ESM	Electronic Surveillance Measures
EWSP	Electronic Warfare Self Protection
FAA	Federal Aviation Administration (United States of America)
FADEC	Full Authority Digital Engine Controller
FCDC	Flight Control Data Concentrators
FCPC	Flight Control Primary Computers
FHA	Functional Hazard Assessment
FLIR	Forward Looking Infra-Red
FM	Formal Methods
FMGC	Flight Management Guidance Computer
FMS	Flight Management System
FMS	Foreign Military Sales
FOCA	Federal Office of Civil Aviation (Switzerland)
FOSOW	Follow-On Stand-Off Weapon

FPTA	Fault Propagation and Transformation Analysis
FPTN	Fault Propagation and Transformation Notation
FSEU	Flap Slat Electronics Unit
FTA	Fault Tree Analysis
GPS	Global Position System
GSN	Goal Structuring Notation
HF	High Frequency
HUD	Heads-Up Display
HUG	Hornet Upgrade Program
IBIT	Initiated Built In Test
ICAO	International Civil Aviation Organization
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFF	Identify Friend or Foe
ILS	Instrument Landing System
INS	Inertial Navigation System
IO	Input Output
IOM	Input Output Module
ISO	International Organisation for Standardisation
IT	Information Technology
ITAR	International Traffic in Arms Regulations
JSSSC	Joint Software System Safety Committee
LIM	Limitations
MAA	Military Airworthiness Authority
MC	Mission Computer
MCDU	Multipurpose Control and Display Units
MECH	Mechanical

MIL-STD	Military Standard (United States)
MOD	Modification
MoD	Ministry of Defence (United Kingdom)
MRH	Multi-Role Helicopter
MSO	Military Standard Order
NAA	National Airworthiness Authority
ND	Navigation Display
OMG	Object Management Group
OO	Object Oriented
PBIT	Periodic Bit
PDF	Probability Distribution Function
PFC	Primary Flight Computers
PRA	Probabilistic Risk Assessment
PSEU	Proximity Switch Electronics Unit
RAAF	Royal Australian Air Force
RAF	Royal Air Force
RPEQ	Registered Professional Engineer of Queensland
RT	Remote Terminal
RTCA	Radio Technical Commission for Aeronautics
SAARU	Standby Attribute and Air Data Reference Unit
SACM	Structured Assurance Case Meta-model
SAEM	Software Assurance Evidence Meta-model
SAL	Safety Assurance Level
SCEFC	Spoiler Control / Electronic Flap Computers
SEAL	Safety Evidence Assurance Level
SHRI	Software Hazard Risk Index
SIL	Safety Integrity Level

SPO	Systems Program Office
SPR	System/Software Problem Report
SSA	System Safety Assessment
SSEI	Software Systems Engineering Initiative
SSSH	Software System Safety Handbook
TACAN	Tactical Air Navigation
T&E	Test and Evaluation
TSO	Technical Standard Order
TUAV	Tactical Unmanned Aerial Vehicle
UFC	Up Front Controller
UHF	Ultra High Frequency
UK	United Kingdom
USAF	United States Air Force
USN	United States Navy
V&V	Verification and Validation
VHF	Very High Frequency
VOR	Very High Frequency Omnidirectional Ranging

Bibliography

Adelard, 2008. *Safety Case Structure.* [Online]
Available at:
http://www.adelard.com/web/hnav/resources/iee_pn/approach/safetyCase_structure.htm
[Accessed 12 Mar 2012].

Air Lift Systems Program Office, 2011. *EMERALD Engineering Decision Record Database - MRD IRNs,* Richmond: Defence Materiel Organisation.

Air Lift Systems Program Office, 2011. *EMERALD Engineering Decision Records - 2011 IRNs pertaining to Oxygen Regulator Failures,* Richmond: Defence Materiel Organisation.

Airbus, 1999. *A330 Flight Deck and Systems Briefing for Pilots, STL 472.755/92, Issue 4.* Toulouse, Airbus.

Airsearch, 2008. *FAA Consultant DER Directory.* [Online]
Available at: http://www.airresearch.com/der/DER_Dir.pdf
[Accessed 04 Apr 2012].

Ankrum, T. & Kromholz, A., 2005. *Structured Assurance Cases: Three Common Standards.* Heidelberg, presented at Ninth IEEE International Symposium on High-Assurance Systems Engineering 12-14 Oct 05.

ASTM International, 2005. *G88-05 Standard Guidance for Designing Systems for Oxygen Service.* West Conshohocken, USA: ASTM International.

Australian Department of Defence, 2006. *Safety Engineering in the Procurement of Defence Systems, Issue 2.* Canberra: Australian Government.

Australian National Audit Office, 2009. *Audit Report No.41 2008-09 Performance Audit,* Canberra: The Auditor -General.

Australian Transport Safety Bureau, 2007. *In-flight upset, 240km NW Perth, WA, Boeing Co 777-200, 9M-MRG,* Canberra: Australian Government.

Australian Transport Safety Bureau, 2008. *In-Flight Upset, 154 km West of Learmonth, WA, 7 October 2008, VH-QPA, Airbus A330-303,* Canberra: Australian Government.

Australian Transport Safety Bureau, 2012. *Aviation Safety Investigations and Reports.* [Online]

Available at: http://www.atsb.gov.au/publications/safety-investigation-reports.aspx
[Accessed 11 Jul 2012].

Aviation Glossary, 2012. *Certification.* [Online]
Available at: http://aviationglossary.com/certification/
[Accessed 2013 Sep 13].

Avizienis, A., Laprie, J., Randell, B. & Landwehr, C., 2004. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on Dependable and Secure Computing,* 1(1), pp. 11-33.

Azzalini, A., 1996. *Statistical Inference - Based on the Likelihood.* 1st ed. Florida: Capham and Hall / CRC Press.

Bartley, G., 2001. Chapter 11 - Boeing B-777: Fly-By-Wire Flight Controls. In: C. Spitzer, ed. *The Avionics Handbook.* Williamsburg(Virginia): CRC Press.

Berdie, D., Anderson, J. & Niebuhr, M., 1986. *Questionnaires: Design and Use.* 2nd ed. Metuchen, N.J. USA: The Scarecros Press.

Bloomfield, R. & Bishop, P., 2010. *Safety and Assurance Cases: Past, Present and Possible Future - an Adelard Perspective.* London, Making Systems Safer, Proceedings of the Eighteenth Safety-Critical Systems Symposium, Bristol, UK, 9-11 Feb 2010, Springer-Verlag, pp. 51-67.

Board of Professional Engineers Queensland, 2013. [Online]
Available at: http://www.bpeq.qld.gov.au/iMIS15/BPEQ/
[Accessed 03 Mar 2013].

Boehm, B., 2008. Making a Difference in the Software Century. *IEEE Computer,* 41(3), pp. 78-84.

Bondavalli, A. & Simoncini, L., 1990. *Failure Classification with Respect to Detection,* Cairo: First Year Report, Task B: Specification and Design for Dependability, Volume 2, ESPRIT BRA Project 3092 Predictably Dependable Computing Systems, Second IEEE Workshop on Future Trends of Distributed Computing Systems.

Briere, D., Favre, C. & Traverse, P., 2001. Chapter 12 - Electrical Flight Controls, From Airbus A-320/330/340 to Future Transport Aircraft: A Family of Fault-Tolerant System. In: C. Spitzer, ed. *The Avionics Handbook.* Williamsburg(Virginia): CRC Press.

Briere, D. & Traverse, P., 1993. *AIRBUS A320/A330/A340 Electrical Flight Controls - A Family of Fault Tolerant Systems.* Toulouse, France: IEEE 0731-3071/93, FTCS-23.

Digest of Papers, The Twenty-Third International Symposium on Fault-Tolerant Computing.

Buus, H. et al., 1995. *777 Flight Controls Validation Process.* Cambridge, USA: IEEE 0-7803-3050-1/95, presented at Digital Avionics Systems Conference, 14th DASC, 5-9 Nov 1995 .

CENELEC, 2001. *EN50128 - Railway Applications : Communications, Signalling and Processing Systems. Software for Railway Control and Protection Systems.* Brussels: European Committee for Electrotechnical Stanardisation.

Certification Services, Inc., 2012. *CSI - Certification Services, Inc. Aviation Safety. From the ground up..* [Online]
Available at: http://www.certification.com/
[Accessed 03 Apr 2012].

Civil Aviation Authority, 2003. *Air Traffic Services Safety Engineering - Part B, Section 3, Systems Engineering - SW01 Regulatory Objectives for Software Safety Assurance in ATS Equipment.* London: Civil Aviation Authority.

Clements, P. et al., 2010. *Documenting Software Architectures: Views and Beyond.* 2nd ed. Boston: Addison-Wesley.

Committee on Certifiably Dependable Software Systems, 2007. *Software for Dependable Systems: Sufficient Evidence?.* Washington, D.C.: The National Academies Press.

Commonwealth of Australia, 1920. *Air Navigation Act.* Canberra: Australian Goverment.

Commonwealth of Australia, 1988. *Civil Aviation Act.* Canberra: Australian Government.

Commonwealth of Australia, 2012. *DEF (AUST) 5657 Australian Cost Schedule Control Systems Criteria; Implementation Guide.* [Online]
Available at: http://www.defence.gov.au/dmo/esd/evm/DefAust5657.cfm
[Accessed 04 Apr 2012].

Defence Aviation Safety Authority, 2011. *Defence Instructions (General) OPS 02-2 Defence Aviation Safety Program - AMDT NO 2.* Canberra: Australian Department of Defence.

Defence Materiel Organisation , 2012. *Procurement and Contracting - ASDEFCON Suite of Tenders and Contracting Templates.* Canberra: Australian Department of Defence.

Defence Materiel Organisation, 2010. *Discussion Paper - Contracting 'Cost' Models & Performance Based Contracting Concepts.* [Online]
Available at:
http://www.defence.gov.au/dmo/gc/Contracting/pbcd/Contracting_CMPBCC.pdf
[Accessed 04 Apr 2012].

Defense Contract Management Agency, 2012. *Types of Contracts / Instruments.* [Online]
Available at: http://guidebook.dcma.mil/18/ContRecRevconttypes.htm
[Accessed 04 Apr 2012].

Department of Computer Science, 2004. *HRM: Hazard and Risk Management & Safety Cases - Lecture Notes.* York: University of York.

Devore, J., 2011. *Probability and Statistics for Engineering and Science.* 8th ed. Stamford: Cengage Learning Inc.

Directorate General Technical Airworthiness, 2010. *AAP7001.053 - Technical Airworthiness Management Manual - AL1.* RAAF Williams - Laverton: Australian Department of Defence.

Directorate General Technical Airworthiness, 2010. *AAP7001.054 - Airworthiness Design Requirements Manual.* RAAF Williams - Laverton: Australian Department of Defence.

Docker, T., 2011. *A Project Manager's View of Safety-Critical Systems.* Southampton, in Proceedings of the Nineteenth Safety-Critical Systems Symposium, 8-10 Feb, Springer-Verlag.

Driscoll, K. & Hoyme, K., 1992. *The Airplane Information Management System: An Integrated Real-time Flight-deck Control System.* Phoenix, USA, IEEE 1052-8725/92, in Proceedings of Real-Time Systems Symposium, 2-4 Dec 1992.

EASA, 2011. *Certificartion Specifications and Acceptable Means of Compliance for Large Aeroplanes.* Koeln: European Aviation Safety Agency.

Eccles, M., 2007. *Deploying Safety Critical Standards Internationally, Joint Strike Fighter Program Presentation.* [Online]

Available at: http://sstc-online.org/2007/index.cfm?fs=pres&aid=1888&ld=530 [Accessed 13 Sep 2013].

Edwards, C., Lombaerts, T. & Smaili, H., 2010. *Fault Tolerant Flight Control: A Benchmark Challenge.* 2010 edition ed. Berlin: Springer.

Ezhilchelvan, P. & Shrivastava, S., 1989. *A Classification of Faults in Systems,* Great Britain: University of Newcastle upon Tyne, Technical Report.

FAA, 1996. *Technical Standard Order TSO-C129a Airborne Supplemental Navigation Equipment Using the Global Positioning System (GPS).* Washington D.C.: Federal Aviation Administration.

Federal Aviation Administration, 1988. *Advisory Circular AC 25.1309-1A System Design and Analysis.* Washington D.C.: United States Department of Transportation.

Federal Aviation Administration, 2003. *Order 8110.49 Software Approval Guidelines.* Washington D.C.: United States Department of Transportation.

Fenelon, P. & McDermid, J., 1994. *New Directions in Software Safety: Causal Modelling As An Aid To Integration.* Gaithersburg, COMPASS '94, Ninth Annual Conference on Computer Assurance, 27 Jun - 01 Jul 1994, High Integrity Systems Engineering Group, University of York.

Fenn, J. & Jepson, B., 2005. *Putting Trust into Safety Arguments.* Southhampton, in Proceedings of the Thirteenth Safety-critical Systems Symposium, 8-10 Feb 2005, Springer.

G-48 Technical Committee, 2008. *GEIA-STD-0010:2008 Best Practices for System Safety Program Development and Execution.* USA: TechAmerica.

Girard, M. & Sharpe, P., 1999. *F/A-18 Testing of Flight Control System Reversion to Mechanical Backup.* Snowmass at Aspen, USA, in Proceedings of Aerospace Conference, IEEE (Volume:5).

Graydon, P., Knight, J. & Green, M., 2010. *Certification and Safety Cases.* Minneapolis, USA, presented at International System Safety Conference 30 Aug - 03 Sep 2010.

Habli, I. & Kelly, T., 2007. *Achieving Integrated Process and Product Safety Arguments.* Bristol, in Proceedings of the Fifteenth Safety-critical Systems Symposium, 13–15 Feb 2007, Springer.

Haddon-Cave, C., 2009. *The Nimrod Review - An independent review into the broader issues surrounding the loss of RAFNimrod MR2 Aircraft XV230 in Afghanistan in 2006.* London: The Stationery Office.

Hammett, R., 2001. *Design by Extrapolation - An Evaluation of Fault Tolerant Avionics.* Cambridge, Massachusetts: Presented at the 20th Digital Avionics Systems Conference, The Charles Stark Draper Laboratory, Inc..

Harris, P., 2006. *An Introduction to Law.* 7th ed. United Kingdom: Cambridge University Press.

Hawkins, R., Kelly, T., Knight, J. & Graydon, P., 2011. *A New Approach to Creating Clear Safety Arguments.* Southampton, in Proceedings of the Nineteenth Safety-Critical Systems Symposium, 8-10 Feb 2011, Springer.

Hawkins, R. & McDermid, J., 2009. *SSEI-TR-0000041 Software Safety Evidence Selection and Assurance,* York, Great Britain: Software Systems Engineering Initiative.

Head of Certification Experts Department, 2011. *Certification Memorandum EASA CM - SWCEH - 002 Software & Complex Electronic Hardware Selection Issue 01.* Koeln : European Aviation Safety Agency.

Hitt, E. & Mulcare, D., 2001. Chapter 28 - Fault-Tolerant Avionics. In: C. R. Spitzer, ed. *The Avionics Hanbook.* Williamsburg: CRC Press.

Holloway, C., 2012. *Towards Understanding the DO-178C / ED-12C Assurance Case.* Edinburgh, Presented at the 7th IET International Conference on System Safety, 15-18 Oct 12.

Hornish, R., 1994. *777 Autopilot Flight Director System.* Phoenix, USA, IEEE 0-7803-2425-0/94, Presented at the 13th Digital Avionics Systems Conference, AIAA/IEEE .

IEC, 1998. *IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems.* Geneva: International Electrotechnical Commission.

IEC, 2010. *IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems Edition 2.0.* Geneva: International Electrotechnical Commission.

IEEE Computer Society, 1995. *J-STD-016-1995 Standard for Information Technology Software Life Cycle Process Software Development Acquirer-Supplier Agreement.* Los Alamitos, USA: The Institute of Electrical and Electronics Engineers.

IEEE, 1991. *IEEE 610.12-1990 ISSS Standard Glossary of Software Engineering Terminology.* Los Alamitos, USA: IEEE Standards Board, The Institute of Electrical and Electronics Engineers.

International Program Office, 2010. *Stage of Involvement #2 Software Development Audit Report for the C-130J Block 7.0,* Dayton, Ohio: Partner Nations and United States Government Audit Team.

ISO/IEC, 1998. *ISO/IEC 15026:1998 Information Technology - System and Software Integrity Levels.* Switzerland: International Organization for Standardization, International Electrotechnical Commission.

ISO/IEC, 2008. *ISO/IEC 12207:2008 Systems and Software Engineering - Software Lifecycle Processes.* Geneva: International Organization for Standardization, International Electrotechnical Commission.

ISO/IEC, 2009. *ISO/IEC 15408 Information Technology - Security Techniques - Evaluation Criteria for IT Security.* Geneva: International Organization for Standardization, International Electrotechnical Commission.

ISO, 2009. *ISO 31000:2009 Risk Management - Principles and Guidelines.* Geneva: International Organization for Standardization.

ISO, 2011. *ISO 26262:2011 Road Vehicles - Functional Safety.* Geneva: International Organization for Standardization.

Joint IECCA and MUF Committee on MASCOT, 1987. *The Official Handbook of MASCOT - Version 3.1.* Malvern: Royal Signals and Radar Establishment.

Joint Services Software Safety Committee, 1999. *Software System Safety Handbook.* Washington D.C.: United States Department of Defense.

Joint Software Systems Safety Engineering Workshop, 2010. *Joint Software Systems Safety Engineering Handbook,* Indian Head, USA: Naval Ordnance Safety and Security Activity.

JSF Program, 2013. *F-35 Lightning II - History.* [Online]
Available at: http://www.jsf.mil/history/his_jsf.htm
[Accessed 22 Apr 2013].

Kelly, T., 1998. *Arguing Safety - A Systematic Approach to Managing Safety Cases,* York, UK: PhD Thesis, Department of Computer Science, University of York.

Kelly, T., 2007. *Reviewing Assurance Arguments - A Step-By-Step Approach.* Edinburgh UK, Workshop on Assurance Cases for Security — The Metrics Challenge at the International Conference on Dependable Systems and Networks.

Kelly, T., 2008. Are Safety Cases Working?. *Safety Critical Systems Club Newsletter,* 17(2), pp. 31-33.

Kelly, T., 2008. *Can Process-Based and Product-Based Approaches to Software Safety Certification be Reconciled?.* London, Improvements in System Safety, Springer, pp. 3-12.

Kelly, T., McDermid, J. & Weaver, R., 2005. *Goal-based safety standards: opportunities and challenges.* San Diego, Presented at the 23rd International System Safety Conference, System Safety Society.

Kinnersly, S., 2011. *Safety Cases - what can we learn from Science.* Southampton, in Proceedings of the Nineteenth Safety-Critical Systems Symposium, 8-10 Feb 2011, Advances in Systems Safety, Springer-Verlag.

Kopetz, H. & Bauer, G., 2003. The Time-triggered Architecture. *Proceeding of the IEEE,* 91(1), pp. 112-126.

Kossiakoff, A., Sweet, W., Seymour, S. & Biemer, S., 2011. *Systems Engineering Principles and Practice.* 2nd ed. New Jersey: John Wiley & Sons Inc..

Kowal, B., Scherz, C. & Quinlivan, R., 1992. *C-17 Flight Control System Overview.* Dayton, USA, Proceedings of the IEEE 1992 Aerospace and Electronics Conference, pp. 24-31.

Leveson, N., 1995. *Safeware: System Safety and Computers.* Reading, Massachusetts: Addison Wesley.

Lindsay, P. & McDermid, J., 1997. *A systematic approach to software safety integrity levels,* Brisbane: Software Verification Research Centre, The University of Queensland.

Littlewood, B., 2007. *Limits to Dependability Assurance - A Controversy Revisited.* London, 29th International Conference on Software Engineering, Centre for Software Reliability, City University.

Littlewood, B., Strigini, L., Wright, D. & Courtois, P., 1998. *Examination of Bayesian Belief Network for Safety Assessment of Nuclear Computer-based Systems,* Brussels: ESPRIT DeVa Project 20072.

Marks, P., 2008. Flight of the Software Bugs. *New Scientist,* pp. 26-36.

Mason, J. & Friedman, A., 2004. *The Professionalisation of UK Professional Associations: Governance, Management and Member Relations.* UK: Professional Associations Research Network.

Mayes, G., 2013. *Because: How to Analyse and Evaluate Ordinary Reasoning, Section 5: Argument and Explanation, Department of Philosophy, Sacramento State University.* [Online]
Available at:
http://www.csus.edu/indiv/m/mayesgr/Phl4/Because/Part5ArgumentExplanation.htm
[Accessed 05 Aug 2013].

McDermid, J., 2001. *Software Safety: Where's the Evidence?.* Brisbane, Australia, In Proeedings of Sixth Australian Workshop on Industrial Experience with Safety Critical Systems and Software, CPRIT.

McDermid, J., 2008. Risk, Uncertainty, Software and Professional Ethics. *Safety Systems: The Safety-Critical Systems Club Newsletter*, January, 17(2).

McDermid, J., 2010. *Conversation with J.A. McDermid (PhD Suprvisor)* [Interview] 2010.

McDermid, J., 2012. *Personal Discussion on ALARP* [Interview] 2012.

McDermid, J. & Kelly, T., 2006. Software in Safety Critical Systems: Achievement and Prediction. *Nuclear Future,* 03(03).

McDermid, J. & Pumfrey, D., 2001. *Software Safety: Why is there no Consensus?.* York: University of York.

McDermid, J. & Rae, A., 2012. *Goal-Based Safety Standards: Promises and Pitfalls.* Bristol, in Proceedings of the Twentieth Safety-Critical Systems Symposium, 7-9 Feb 2012, Springer.

Menon, C., Hawkins, R. & McDermid, J., 2009. *SSEI-BP-000001 Interim Standard of Best Practice on Software in the Context of DS 00-56 Issue 4,* UK: Software Engineering Initiative.

Ministry of Defence, 1996. *Defence Standard 00-56 Safety Management Requirements for Defence Systems Part 1: Requirements, Issue 2.* Great Britain: UK Defence Standardisation.

Ministry of Defence, 1997. *Defence Standard 00-55 Requirements for Safety Related Software in Defence Equipment, Issue 2.* Great Britain: UK Defence Standardisation.

Ministry of Defence, 2007. *Defence Standard 00-56 Safety Management Requirements for Defence Systems Part 1: Requirements, Issue 4.* Great Britain: UK Defence Standardisation.

Ministry of Defence, 2007. *Defence Standard 00-56 Safety Management Requirements for Defence Systems Part 2: Guidance on Establishing a Means of Complying with Part 1, Issue 4.* Great Britian: UK Defence Standardisation.

Morgan, M., 2001. Chapter 29 - Boeing B-777. In: C. Spitzer, ed. *The Avionics Handbook.* Williamsburg(Virginia): CRC Press.

Murphy, J. J., Ericson, J. M. & Zeuschner, R. B., 2003. *The Debators Guide.* 3rd ed. Chicago: Southern Illinois University Press.

National Archives and Records Administration, 2012. *Title 14 Aeronautical and Space, Code of Federal Regulations, Chapter I, Federal Aviation Administration, Department of Transportation, Subchapter C - Aircraft, Part 25 Airworthiness Standards: Transport Category Airplanes.* USA: United States Government.

National Transportation Safety Board, 2013. *Accident Reports.* [Online]
Available at: http://www.ntsb.gov/investigations/reports.html
[Accessed 09 Jul 2012].

Nola, R. & Sankey, H., 2007. *Theories of Scientific Method: An Introduction (Philosophy and Science).* 1st ed. Canada: McGill-Queen's Unversity Press.

NTSB, 2006. *Safety Report NTSB/SR-06/02 Safety Report on the Treatment of Safety-Crtical Systems in Transport Airplanes,* Washington, D.C.: National Transportation Safety Board.

Nuttall, J., 2002. *An Introduction to Philosophy.* 1st ed. Cambridge UK: Polity Press.

Object Management Group, Inc., 2010. *Argumentation Metamodel (ARM) - FTF - Beta 1.* Needham, USA: OMG Document Number: ptc/2010-08-36.

Object Management Group, Inc., 2010. *Software Assurance Evidence Metamodel (SAEM) - FTF - Beta 1.* Needham, USA: OMG Document Number: ptc/2010-08-37.

Object Management Group, Inc., 2012. *Catalog of OMG Modernization Specifications - Structured Assurance Case Metamodel (SACM).* [Online]
Available at: http://omg.org/technology/documents/modernization_spec_catalog.htm
[Accessed 02 Apr 2012].

Object Management Group, 2010. *ptc/2010-08-36 Argumentation Metamodel (ARM) FTF - Beta 1.* Needham: Object Management Group.

Oppenheim, A., 2001. *Questionnaire Design, Interviewing and Attitude Measurement.* New ed. London, Great Britain: Continuum.

Origin Consulting Limited, 2011. *GSN Community Standard Version 1.* York: Origin Consulting (York) Limited.

Oxford University Press, 2010. *Oxford Dictionary of English.* 3rd Edition ed. Great Britain: Oxford University Press.

Palmer, J., 1997. *Traceability.* Los Alamitos, USA, IEEE Computer.

Pop, D. & Kahler, R., 1992. *C-17 Flight Control Systems Software Design.* Seattle, USA, IEEE 0-7803-0820-4/92, Presented at 11th Digital Avionics Systems Conference, IEEE/AIAA.

Potocki de Montalk, J., 2001. Chapter 30 - New Avionics Systems - Airbus A330/A340. In: C. Spitzer, ed. *The Avionics Handbook.* Williamsburg: CRC Press.

Praxis Critical Systems, 2001. *REVEAL - A Keystone of Modern Systems Engineering,* Bath, UK: S.P0544.19.1, Issue 1.2, Praxis Critical Systems.

Pumfrey, D., 1999. *The Principled Design of computer System Safety Analyses,* York, UK: PhD Thesis, Department of Computer Science, University of York.

Rae, A., Alexander, R. & McDermid, J., 2012. *The Science and Superstition of Quantitative Risk Assessment.* Proceedings of PSAM 11 & ESREL 2012, International Association of Probabilistic Safety Assessment and Management, IAPSAM.

Redmill, F., 2000. *Safety Integrity Levels - Theory and Problems.* Southampton, in Proceedings of the Eighth Safety-critical Systems Symposium, Springer.

Robinson, G., 2000. *Practical Strategies for Experimenting.* West Sussex: John Wiley & Sons.

Royal Australian Air Force, 2005. *AAP7211.031-1 Flight Manual C-130J-30.* RAAF Williams - Laverton: Australian Department of Defence.

Royal Australian Air Force, 2007. *Australian Air Publication (AAP) 1000-D Australian Air Power Manual.* RAAF Williams - Laverton: Australian Department of Defence.

Royal Australian Air Force, 2008. *AAP7213.006-1-NFM-000 Flight Manual AF/A-18A and AF/A-18B.* RAAF Williams - Laverton: Australian Department of Defence.

Royal Australian Air Force, 2012. *(AT)A1-F18AC-570-100 Principles of Operation Integrated Flight Controls.* RAAF Williams - Laverton: Australian Department of Defence.

Royal Australian Air Force, 2012. *(AT)A1-F18AC-745-100 Principles of Operation Multipurpose Display Group.* RAAF Williams - Laverton: Australian Department of Defence.

RTCA Inc., 1992. *RTCA/DO-178B: Software Considerations in Airborne Systems and Equipment Certification.* Washington D.C.: RTCA Inc..

RTCA Inc., 2011. *RTCA/DO-178C: Software Considerations in Airborne Systems and Equipment Certification.* Washington D.C.: RTCA Inc..

RTCA Inc., 2011. *RTCA/DO-333 Formal Methods Supplement to DO-178C and DO-278A.* Washington D.C.: RTCA Inc..

RTCA Inc., 2012. *SC-205 (Joint with EUROCAE WG-71) Software Considerations.* [Online]
Available at: http://www.rtca.org/comm/Committee.cfm?id=55
[Accessed 01 Jan 2013].

Rushby, J., 1993. *Formal Methods and the Certification of Critical Systems,* Menlo Park, California: Computer Science Laboratory, SRI International.

SAE Aerospace, 2010. *Aerospace Recommended Practice 4754A - Guidelines for Development of Civil Aircraft and Systems.* Warrendale: SAE International.

SAE International, 1996. *Aerospace Recommended Practice 4754 - Certification Considerations for Highly Integrated or Complex Aircraft Systems.* Warrendale, USA: Society of Automotive Engineers, Inc..

Simpson, H., 1986. The MASCOT Method. *Software Engineering Journal,* 1(3), pp. 103-120.

Simpson, H., 1994. *Architecture for Computer Based Systems.* Stockholm, Proceedings of the IEEE Workshop on Engineering of Computer Based Systems.

Simpson, H., 1996. *Layered Architecture(s): Principles and Practice in Concurrent and Distributed Systems.* Monerey, USA, Proceedings of the 8th IEEE Symposium on Parallel and Distributed Computing Processing.

Simpson, H. & Jackson, K., 1979. Process Synchronisation in MASCOT. *The Computer Journal,* 22(4), pp. 332-345.

Storey, N., 1996. *Safety-Critical Computer Systems.* 1st ed. Essex: Pearson Prentice Hall.

The IET, 1999. *Safety Competency and Commitment - Competency Guidance for Safety-relate System Practitioners.* Stevenage, UK: The Institute of Engineering and Technology.

The IET, 2007. *Competency Criteria for Safety-related System Practitioners.* Stevenage, UK: The Institute of Engineering and Technology.

The Macquarie Library, 2002. *The Macquarie Concise Dictionary.* Third Edition ed. Macquarie(NSW): The Macquarie Library.

Think Defence, 2010. *Step Forward (again) Lord Levene.* [Online]
Available at: http://www.thinkdefence.co.uk/2010/08/step-forward-again-lord-levene/
[Accessed 04 Apr 2012].

Toulmin, S., 1958. *The Uses of Argument.* UK: Cambridge University Press.

Uczekaj, J., 1995. *Reusable Avionics Software - Evolution of the Flight Management System.* Cambridge, USA, IEEE 0-7803-3050-1/95, 14th Digital Avionics Systems Conference.

UK Health and Safety Executive, 2013. *ALARP at a Glance.* [Online]
Available at: http://www.hse.gov.uk/risk/theory/alarpglance.htm
[Accessed 01 Jan 2013].

United Kingdom Goverment, 1974. *Health and Safety at Work etc. Act 1974.* London: United Kingdom Goverment.

United States Air Force Space Command, 2009. *Space and Missile Systems Centre Standard SMC-S-21 Technical Reviews and Audit for Systems, Equipment and Computer Software.* Washington D.C.: United States Department of Defense.

United States Department of Defense, 1988. *DOD-STD-2167A - Defense Systems Software Development.* Washington DC: United States Government.

United States Goverment, 2012. *Title 48 - Federal Acquisition Regulations System, Chapter 1 - Federal Acquisition Regulation, Subchapter C - Contracting Methods and Contract Types, Part 15 Contracting By Negotiation.* USA: United States Goverment.

United States of America, 2012. *Code of Federal Regulations, Title 14, Aeronautical and Space.* Washington DC: United States Government.

University of York, 2004. *CAS: Computers and Software and ISA - Lecture Notes.* York, UK: Department of Computer Science.

US DoD, 1993. *MIL-STD-882C System Safety Program Requirements.* USA: United States Department of Defense.

US DoD, 1994. *MIL-STD-498 Software Development and Documentation.* USA: United States Department of Defense.

US DoD, 1995. *MIL-STD-1521B Technical Reviews and Audits for Systems, Equipment, and Computer Software.* USA: United States Department of Defense.

US DoD, 2000. *MIL-STD-882D Standard Practice for System Safety.* USA: United States Department of Defense.

US DoD, 2011. *MIL-STD-882E Department of Defence Standard Practice: System Safety.* USA: United States Department of Defense.

Van de Ven, A., 2007. *Engaged Scholarship - A Guide for Organizational and Social Research.* Oxford: Oxford University Press.

Wade, M., 2009. *TAR Requirements for Major Projects Course Notes.* Melbourne, Australia: Directorate General Technical Airworthiness.

Weaver, R., 2003. *The Safety of Software - Constructing and Assuring Arguments,* York, UK: PhD Thesis, Department of Computer Science, University of York.

Weaver, R., Fenn, J. & Kelly, T., 2003. *A Pragmatic Approach to Reasoning about the Assurance of Safety Arguments.* Canberra: 8th Australian Workshop on Safety Critical Systems and Software, Conferences In Research and Practice in Information Technology.

Witwer, B., 1995. *Systems Integration of the 777 Airplane Information Management Systems (AIMS): A Honeywell Perspective.* Cambridge, USA, IEEE 0-7803-3050-1/95, 14th Digital Avionics Systems Conference.

Yea, Y., 1996. *Triple-Triple Redundant 777 Primary Flight Computer.* Aspen, IEEE 0-7803-3196-6/96, Proceedings of Aerospace Applications Conference.

Yea, Y., 2001. *Safety Critical Avionics for the 777 Primary Flight Controls System.* Daytona Beach, USA, IEEE 0-7803-7034-1/01, 20th Digital Avionics Systems Conference.