

SCSC
For Everyone Working in System Safety

ISSN 2754-1118
Vol 5, Issue 1

The Safety-Critical Systems Club

SAFETY-CRITICAL SYSTEMS eJOURNAL

This collation page left blank intentionally.

Foreword to the 2026 First Issue

Welcome to the first issue of the fifth volume of the Safety-Critical Systems eJournal, published by the Safety-Critical Systems Club (SCSC) with Publication Number SCSC-214.

There is a new feature in the papers of the on-line version of this volume, and all subsequent, intended to make it more accessible to screen readers, as used by blind or partially-sighted readers to convert text to speech (or Braille). Screen readers cannot parse an image, so descriptive "Alt Text" will now be provided for each figure in a paper. When a screen reader gets to an image, it will tell the user that an image is present, and then read out the Alt Text. The on-line version of the eJournal has actually had Alt Text since Volume 1 in the "wrapper", i.e. everything except for the papers themselves, which are presented as pdf files. In a conventional pdf file reader, the Alt Text will be visible when the mouse pointer is hovered over the image (but note that some browser-based pdf readers do not support this feature).

The Word template, which includes style guidelines for authors, has been updated to include Alt Text: see https://scsc.uk/journal/index.php/scsj/libraryFiles/downloadPublic/4. If you are not already registered as an author, please go to https://scsc.uk/journal and click on "Register" at the top right of the page.

This issue contains three papers:

1. Derek Fowler, in his paper, "*IEC 61508 Viewpoint on System Safety in the Transport Sector: Part 4 — A Safety Assurance Perspective*", presents a generic safety-assurance framework, based on classical systems-engineering practices and rigorous requirements-engineering principles, against which the IEC 61508-1 safety-assessment processes (and their equivalents in European Air Traffic Management and Rail safety standards) are judged in terms of their potential contribution to the safety case for typical safety-related transport applications.
2. Fred Pollard, Martin Rendell, and Daniel Finnigan, in "*Perils of Precedent in Nuclear Software Qualification*", question the use of some Independent Confidence-Building Measures without justification, other than precedence of use, in the qualification of control and instrumentation systems for UK nuclear power stations, and propose a possible alternative approach.
3. Martin Rendell and Fred Pollard evaluate the practicability of using Statistical Testing for complex systems, like nuclear reactor protection systems, covering each of the core conditions underpinning the successful execution of the technique, in "*Statistical Testing: Qualitative Arguments for a Quantitative Outcome?*". They conclude that while the overall method *can* provide useful qualitative evidence, statistical testing itself cannot provide high-confidence quantitative reliability estimates for complex systems.

My thanks go to the authors for contributing their papers, and also to the anonymous peer-reviewers (at least three per paper) for suggesting improvements. Apologies also to those reviewers who made some recommendations that were not taken up.

In contrast to last year's cover art depicting a fusion reaction producing power, this year we have a nuclear fission power station, Sizewell B in Suffolk, UK (with Sizewell C being constructed nearby).

John Spriggs, SCSC Journal Editor
September 2026

This collation page left blank intentionally.

ISSN 2754-1118 (Online) — ISSN 2753-6599 (Print)

IEC 61508 Viewpoint on System Safety in the Transport Sector

Part 4 — A Safety Assurance Perspective

Derek Fowler

Independent Safety Engineering Consultant, Bracknell, UK

Abstract

This article is the fourth in a series of papers that address the potential application of international generic functional-safety standard IEC 61508 to the transport sector, with a focus on the crucial requirements/specification phases of the system safety lifecycle in Part 1 of the Standard (IEC 61508-1). During the peer reviews of the three predecessor articles, questions were raised about the degree and extent to which some European Air Traffic Management and Rail safety-assessment standards and practices actually complied with the fundamental principles of IEC 61508-1; however, at the time, it had not been the intention of any of those three articles to address such questions, and it was left to readers to reflect on the matter. This article now returns to the subject of safety compliance, but in the context of Safety Assurance, and starts by developing a generic safety-assurance framework, based on classical systems-engineering practices and rigorous requirements-engineering principles, against which the IEC 61508-1 safety-assessment processes, and their equivalents in European Air Traffic Management and Rail safety standards, are judged in terms of their potential contribution to the safety case for typical safety-related transport applications.

1 Introduction

1.1 Background

In the first of four related articles, Fowler (2022) proposed ‘a way of thinking’ about safety assessment of the various safety-related systems deployed in the Transport sector in general — but especially in commercial-aviation and rail applications — that was based closely on the application of the key principles and safety lifecycle set out in Part 1 of IEC 61508 (hereinafter referred to as IEC 61508-1).

A follow-up article took the example of an innovative Air Traffic Management (ATM) operational concept and used it to outline how an IEC 61508-1 approach to safety assessment could be applied effectively to the ATM sector, and what the results thereof might look like; that approach started with the concept of the movement of aircraft in the airspace being (what IEC 61508-4 defines as) the Equipment Under Control (EUC) (Fowler and Fota 2023).

A third article then took a similar approach but with the example of a new, moving-block Communications-based Train Control (CBTC) system for a hypothetical metropolitan

© Derek Fowler 2026.

railway; this was used to outline how an IEC 61508-1 approach to safety assessment could be applied effectively to the Railway sector in general, and what the results thereof might look like; that approach started with the concept of the movement of trains around the railway being the EUC (Fowler and Graebner 2023).

This, the fourth and final article in the series, takes a somewhat different, argument-based, assurance perspective motivated in part by:

- The findings of Fowler (2015), which suggested that neither the CENELEC set of railway standards (in particular the 1999 version of EN 50126 and the 2003 version of EN 50129), nor most national ATM safety-assessment standards[1], applicable at **that** time, complied with even the most basic principle of the IEC 61508-1 safety lifecycle;
- An understanding that significant changes in ATM and Rail safety standards have been made since 2015; and
- The following, rather bold, statement, in the 2017 version of EN 50126-1:

> *"EN 50126 forms part of the railway sector specific application of IEC 61508. Meeting the requirements in this European Standard, together with the requirements of other suitable* [unspecified] *standards, is sufficient to ensure that additional compliance to IEC 61508 does not need to be demonstrated".*

So, what does compliance to IEC 61508 mean in practice and, in any event, what would be the value of the IEC 61508-1 lifecycle processes to system safety in the transport sector? Clearly, addressing these two rhetorical questions would require a rigorous and independent suitable benchmark on which to base informed decisions. It was decided to develop, and to present herein, an Argument-based Safety Assurance Framework for that purpose.

1.2 Historical Context

1.2.1 Hard Lessons

Over the period 1970 to 2002, there were nearly 40 major fatal accidents reported worldwide, involving more than 8,000 immediate fatalities, and across three industry sectors, as follows[2]:

- Aviation: at least 17 major accidents, involving more than 3,700 fatalities;
- Chemical and Process Industries: at least 12 major accidents, involving more than 4,000 fatalities;
- Rail: at least 8 major accidents, involving more than 550 fatalities.

Individually, these accidents shaped modern approaches to systems engineering, human factors, software assurance, safety culture, and risk management. However, the concept of a Safety Management System (SMS) did not emerge all at once; rather, it evolved gradually through the 1970s, 1980s and 1990s, as industries moved away from viewing accidents as purely technical failures and toward also addressing management, organisational and human factors. Some of the earliest recognisable SMS-like approaches appeared in response to

[1] In retrospect, that comment would have been better confined to European countries, although the author was not aware of any countries that would have had better safety assessment standards at that time!

[2] These are intended as indicative figures only, and were obtained from an AI-assisted survey. They exclude harmful outcomes caused by deliberate acts of terror, etc.

accidents in high-risk industries such as nuclear power, aviation, chemical processing, offshore oil and gas, and railways. For example:

- **Tenerife Airport Disaster** (1977), in which two aircraft colliding during take-off, brought focus to the need for improved crew communication and cockpit resource management;
- **Three-Mile Island Nuclear Plant Accident** (1979), in which cooling-system failure caused a partial reactor meltdown, led to enhanced operator training, human factors engineering, and the creation of the US Institute of Nuclear Power Operations to promote safety culture;
- **American Airlines Flight 191 Crash** (1979), which remains one of the most studied aviation disasters in history, because it demonstrated how maintenance-related damage can trigger catastrophic system failures. It is frequently cited in aviation safety training and accident investigations as an example of the importance of maintenance standards, engineering redundancy, and regulatory oversight;
- **Bhopal (India) Chemical Plant Disaster** (1984), which involved the accidental release of 40 tons of methyl-isocyanate gas, leading to around 3,000 immediate deaths, tens of thousands of deaths in the longer term, and up to half a million people with permanent disabilities. The accident triggered a multi-decade global evolution in how governments and corporations manage industrial hazards, covering the period 1985 to 2024, and involving the relevant authorities of Canada, India, the EU, the UK, and the USA (Willey et al. 2005) (Cogent Skills 2026) (Mannan et al. 2005).
- The report on the **Challenger Space Shuttle Disaster** (1988), in which failure of an O-ring seal in the right solid rocket booster at liftoff allowed hot gases to escape and damage the external fuel tank, leading to the disintegration of the orbiter, cited flawed engineering decisions, inadequate risk communication, and managerial pressures to maintain launch schedules as key causal factors, and led to sweeping changes in NASA's safety culture, management, and design protocols;
- **Piper Alpha Oil Platform Disaster** (1988), in which a maintenance error caused gas-condensate leak and ignition, and led to triggering multiple explosions, and fire spreading rapidly as adjacent gas pipelines ruptured, engulfing the platform in flames within minutes — of 226 personnel on board, only 61 survived. The Cullen Inquiry report into the disaster (Cullen 1990) fundamentally revolutionised offshore safety by recommending that operators be legally required to produce a comprehensive Safety Case as a means of assuring itself that its operations are [acceptably] safe. In this way, it shifted the regulatory regime from a prescriptive checklist of rules, to a "goal-setting" approach where operators must actively identify risks and demonstrate how they manage them; and
- **Charkhi Dadri Mid-Air Collision** (1996), in which a Saudia Boeing 747 and a Kazakhstan Airlines Ilyushin Il-76 collided at 14,000 feet over India due to language barriers, poor English comprehension by the Kazakh crew, and antiquated ground radars. Following this disaster, the International Civil Aviation Organization (ICAO) made the Airborne Collision Avoidance System (ACAS) mandatory for all major commercial aircraft worldwide. ACAS allows aircraft to communicate directly with each other in mid-air and issue automated "climb" or "descend" commands to pilots, with the intention of completely bypassing airborne and ground-based human error — but see Sub-section 4.4.2 below!

All of the above, and others, provide important context for this article, in terms of an evolution of safety management during the last three decades of the twentieth century, which moved gradually **from**:

- Investigating accidents **after** they occur to determine their causes and prevent similar incidents from happening again; **to**
- Identifying hazards **before** accidents happen to proactively recognize potential sources of harm and implement preventive measures; **and to**
- Managing risk **continually** throughout a system's lifecycle by regularly assessing, monitoring, and controlling risks from design and development through operation, maintenance, and decommissioning.

1.2.2 A New Generic Approach

In 1985, the International Electrotechnical Commission (IEC) initiated an evaluation of the viability of a generic standard for Programmable Electronic Systems (PES) used in safety applications, and in 1995 distributed a draft, known as "IEC 1508", which introduced early concepts of Safety Integrity Levels (SIL) and lifecycle frameworks to specific manufacturing sectors. In the period 1998 to 2000, the seven parts of the first full edition of what became IEC 61508, "*Functional safety of electrical/ electronic/programmable electronic [E/E/PE] safety related systems*", were released and became probably the most widely accepted, international generic standard on functional safety. Although its roots can be traced to process industries, the intention behind the Standard had always been, and still is, to provide a solid, comprehensive basis for adaptation to a very wide range of industry sectors.

IEC 61508 is about Functional Safety, which it defines as "*part of the overall safety, relating to the EUC and the EUC Control System, that depends on the correct functioning of the safety-related systems and other risk-reduction measures*" — and which leads to its very specific approach to systems safety assessment. This concept was described in some detail in Fowler (2022) but, because it is not always understood, yet has a major influence on the Safety Assurance structure developed herein, the key points are worth restating here, as follows:

- IEC 61508 is founded on the notion of an EUC that is defined as "*equipment, machinery, apparatus or plant used for manufacturing, process, transportation, medical or other activities*", and which is inherently hazardous to human life and/or the environment;
- The EUC might have an associated, separately identifiable, EUC Control System that "*... responds to input signals from the [EUC] and/or from an operator, and generates output signals causing the EUC to operate in the desired manner*";
- Safety-related Systems (SRSs) are developed in order to implement the Safety Functions that are necessary to mitigate the pre-existing[3], EUC Hazards, thereby achieving, or maintaining, an "*acceptably safe state for the EUC*";
- Safety Integrity is "*the probability of an SRS satisfactorily performing the specified Safety Functions, under all the stated conditions, within a stated period of time, so as to achieve, and maintain, an acceptably safe state for the EUC*". As emphasised throughout this article, it is NOT simply the likelihood of failure of an SRS.

Note: A more comprehensive introduction to these crucial principles, which underpin IEC 61508, can be found in Redmill's (1999) introduction to the standard; it is as valid and important today as it was at the time!

[3]The hazards that are inherent in the EUC are known as "pre-existing" herein since they pre-date the existence of the SRS(s), and to distinguish them from any hazards that might be introduced by failure of said SRS(s).

1.2.3 The Leveson Contribution

During the period of this IEC 61508 activity, and beyond, Professor Nancy Leveson of MIT had been working in what was then a new area of software safety and reliability, including a major, detailed study into a variety of spacecraft and aircraft accidents that in some way involved software. A helpful overview of this seminal work, and its conclusions, was presented at the 19th International System Safety Conference, in 2001 (Leveson 2001), and, in addressing the rhetorical question as to how many of the accidents were caused by software reliability problems, it referred to:

> *"… confusion between software reliability and safety: most accidents* [studied] *involve software that is doing exactly what it was designed to do, but the designers misunderstood what behavior was required and would be safe, i.e. it reliably performs the wrong function*[s]*".*

More generally she concludes that:

> *"I have noticed … that, over the years, system safety engineering has increasingly drifted toward using reliability engineering techniques and away from classic system safety approaches and has, in particular, adopted this approach for software. This trend may simply be a result of lack of knowledge about software, or it may reflect a lack of appropriate tools to assist in applying system safety approaches to software. A further influence may be that computer science has always been concerned with computer [hardware] reliability and has focused almost exclusively on this quality. Only recently has safety become an issue. Therefore, almost all existing software engineering techniques focus on software reliability, i.e. assuring that the software correctly or reliably satisfies the specified* [functional] *requirements (which* [themselves] *may be incomplete, incorrect, or unsafe)."*

In other words, her findings very much articulate the problems that the early contributors to IEC 61508 set out to address.

It is very disappointing, therefore, that nearly 20 years later Professor Leveson found it necessary to describe, in an Association of Computing Machine (ACM) opinion piece (Leveson, 2020), four major misconceptions which suggest that:

> *"… confusion and misinformation* [still] *abound about software and safety …. Too many incorrect beliefs are being promoted, which are inhibiting progress and, in some cases, unnecessarily costing lives".*

Most importantly, she concludes that:

> *"… the most effective approach to dealing with safety of computer-controlled systems is to focus on creating the safety-related requirements. System and software requirements development are necessarily a system engineering problem, not a software engineering problem. The solution is definitely not in building a software architecture (design) and generating the requirements later, as has been surprisingly suggested by some computer scientists"!*

Again, that is fully in tune with the provisions of IEC 61508-1, and is what this article, (in particular, Section 4 below) is all about.

1.2.4 Developments in European Air Traffic Management Safety Management

From the early 1990s, the UK National Air Traffic Services (NATS) pioneered the early development of a formal SMS, having been mandated by the UK government to operate independently from its regulatory body in the UK Civil Aviation Authority (CAA). That SMS contained all the recognised elements of an SMS; however, like many equivalent system safety-assessment processes at the time, and quite unlike what emerged subsequently in IEC 61508, followed traditional reliability-engineering methods and was based on an assumption that system failures would occur in a random fashion. At the time, such an approach would have been reasonable, since Air Traffic Management (ATM) functional safety rested primarily on the skills and experience of human Air Traffic Controllers, and the hardware tools on which they relied contained little or no software safety functions.

In the late 1990s the UK CAA, being increasingly concerned about the new generation of software-intensive safety-related systems being introduced into ATM, commissioned the development of a set of comprehensive safety objectives, known as "SW01", and first published them formally in December 2002, in what has now become document CAP[4] 670 (UK CAA 2019). SW01 was, and still is, widely regarded as seminal work, but with one stated significant deficiency:

> *"This document assumes that software safety requirements have been derived from a full risk and safety analysis of the system. This will have established the overall safety requirements that have been refined and allocated in the design to software."*

Whereas it rightly claims that "*This is a commonplace system safety process and is described in standards and guidelines such as IEC 61508 Part 1 …*", some system-safety practitioners might find it disappointing that no equivalent, system-level safety objectives have been published.

Meanwhile, on a wider, EUROCONTROL[5] front, parallel developments were taking place. Until the mid-1990s, it had been recognised that safety was an implicit but key feature of ATM operations. Technical systems were functionally quite basic, and the primary role of ATM — that is, reducing the risks that are inherent in aviation, especially the risk of mid-air collisions, or high-energy collisions between aircraft on the ground — was vested in highly skilled, well-motivated human operators (controllers) working to well-defined procedures.

With the increased use of technology, and the introduction of formal safety management systems in ATM during the second half of the 1990s, EUROCONTROL started on the development of its very-detailed "Safety Assessment Methodology" (SAM) (EUROCONTROL 2006a).

The original intention was to base the SAM on the well-established commercial-aircraft-certification procedures, first published in 1996, SAE ARP 4754A and SAE ARP 4761.

SAE ARP 4754A details the engineering processes used to develop civil aircraft and systems. It focuses on:

- bridging high-level aircraft functions down into specific system requirements.
- ensuring the right system is being built and that the system is built correctly.

[4] Civil Aviation Publication

[5] The European Organisation for the Safety of Air Navigation

- defining Development Assurance Levels (DALs) to manage the rigour of hardware and software design.

SAE ARP4761 provides the methods and guidelines to execute the safety assessments mandated by SAE ARP4754A. It works in lockstep with the development process to:

- classify the severity of potential failure conditions using Functional Hazard Assessments (FHA).
- analyse system architecture using techniques like Fault Tree Analysis (FTA) and Failure Modes and Effects Analysis (FMEA) to prove safety requirements are met.
- identify common-cause or cascading failures that could bypass independent systems.

As it turned out, very little of SAE ARP 4754A was adopted for the SAM and so what was promulgated by EUROCONTROL was an entirely failure-based approach that utilised traditional reliability-engineering techniques — contrary to the principles underpinning IEC 61508 and those expounded by Nancy Leveson at the time, and since.

It was not until about 2008, in the face of the enormous challenges presented by the European Commission's Single European Sky ATM Research & Development (SESAR) programme, that EUROCONTROL, with others, started to develop a far more rigorous and capable system safety assessment method (SESAR 2018a) with guidance for its application (SESAR 2018b), which was much more closely aligned with the principles behind SAE ARP 4754A and IEC 61508-1 — see later in Sections 5 and 6 below.

1.2.5 Developments in European Rail Safety Management

During the 1990s, the privatisation and cross-border integration of railways created a critical need for universal safety benchmarks. In response, the European Committee for Electrotechnical Standardization (CENELEC) drafted a sector-specific adaptation of the generic functional safety standard IEC 61508. The original standard was approved as EN 50126:1999, "*Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS)"*, and introduced the well-known 14-phase System (V-model) life cycle to the rail industry.

This was followed by the addition of two related standards:

- EN 50128, "*Software for Railway Control and Protection Systems"*, which defines processes and technical requirements for developing, verifying, validating, and maintaining software used in railway control and protection systems; this was first published formally in 2001; and
- EN 50129, "*Railway Applications – Communication, Signalling and Processing Systems – Safety Related Electronic Systems for Signalling"*, which defines the requirements for the acceptance and approval of safety-related electronic systems used in railway signalling, and was first published formally in 2003[6].

The problem with EN 50126, in particular, was that it was based on (or could be interpreted as being based on) what Nancy Leveson would describe as "*reliability-engineering techniques*" and, as suggested by Fowler (2015), did not comply with even the most basic principles of IEC 61508-1.

Since then, there have been several major updates to the above CENELEC standards:

[6] But published earlier (1998) as a European "pre-standard" (ENV 50129)

- **EN 50126**, in its latest, 2017, version has been split into two parts: EN 50126-1, which is the administrative and managerial backbone of the lifecycle. It governs the entire timeline and covers all four RAMS pillars; and EN 50126-2, which is a highly specialized, technical engineering manual dedicated exclusively to Safety;
- The original 2001 edition of **EN 50128** was a straightforward railway adaptation of IEC 61508-3. It then gradually shifted from being primarily a software development standard to being a broader software assurance standard; with more emphasis on organisational competence, management processes, tool confidence, verification independence, and demonstrating confidence in the overall software engineering process, rather than focusing predominantly on implementation techniques[7];
- The 2003 edition of **EN 50129** established the basic framework for producing a safety case for safety-related electronic signalling systems. In 2018, a major revision aligned it with the EN 50126:2017 lifecycle, updated the safety case requirements, and incorporated guidance from CENELEC documents CLC/TR 50451 and CLC/TR 50506. What is particularly relevant to this article are the safety case aspects, which could be summarised as shifting from "*produce these documents*" to "*demonstrate a convincing safety argument*".

This then brings us back to the question of whether the claim that "*EN 50126 … together with the requirements of other suitable* [unspecified] *standards, is sufficient to ensure that additional compliance to IEC 61508 does not need to be demonstrated*" was justified when it was made by CENELEC in 2017.

In the abstract world of IEC 61508, the EUC is the *problem* since, by definition[8], it is inherently hazardous. The SRS, on the other hand, is the *solution* since, by definition, it provides the Safety Functions that mitigate the hazards that exist within the domain of the EUC. The first step in trying to apply IEC 61508 to a specific sector is to determine a suitable practical interpretation of the "EUC" — in the worked example of Fowler and Graebner (2023), we saw that "*the movement of rail traffic*" was the most suitable interpretation for the rail sector. Similarly, and bearing in mind that the rail example given therein is based on a major upgrade to existing railway operations, it was considered appropriate for the legacy railway infrastructure to be interpreted as being the EUC Control System[9]. By the same token, the SRSs were constituent parts of the new automatic train control system that was being developed for the hypothetical railway.

In the practical world of the above CENELEC standards, things seem (to the author of this article at least) to be less clear. Those standards do not use the term EUC explicitly; rather they refer to, the much more flexible notion of the *System under Consideration* (SuC). Furthermore, whereas the concept of a hazard as a "*condition that could lead to an accident*" (as per (EU) No 402/2013) is on the face of it roughly equivalent to the IEC 61508 definition of "*a potential source of harm*", only the CENELEC usage, in order to comply (EU) No 402/2013 makes the following strong link between hazards and "the system" — presumed to be the SuC[10]:

[7] In 2023, the 2020 edition was superseded by EN 50716 in what might viewed as a refinement and modernisation rather than a complete rethink of railway software assurance.

[8] In IEC 61508-4

[9] Of course, other interpretations might be equally valid provided that, if properties of part, or all, of the EUC Control System had to be relied upon in the achievement of an acceptably safe state for the EUC, then those aspect of the EUC Control System would have to be treated as being safety-related, in accordance with the Standard.

[10] The ORR (2024) guidance document effectively confirms the application of (EU) No 402/2013 to the UK, post-"Brexit".

> *"The risk assessment process starts with the system definition (which can use information from the preliminary system definition). This provides the key details of the system that is being changed - its purpose, functions, interfaces and the existing safety measures that apply to it. In most cases, the hazards which need to be analysed will exist at the boundary of the system with its environment."*

Herein lies a dilemma. If we were to define the SuC as being the whole railway — i.e. equivalent to the EUC — then the hazard-identification process would (or should) start with the hazards that are inherent in railway operations, as per an IEC 61508-1 approach. However, if the SuC were to be defined as being, say, a new automatic train control system that was being introduced to the railway, what "hazards" would we then be identifying?

It seems that the CENELEC standards are aimed more at a change to a railway, rather than at the whole railway operation, whereas IEC 61508 is clearly aimed at the whole operation in whatever sector to which it is to be applied. There *should* be, however, no contradiction here because, for example, in order to specify a complete set of safety requirements for a new automatic train control system, one should *first*:

- have a complete understanding of the EUC, the EUC Control System and the Operational Environment and the interactions between them and the SuC (i.e. the new system);
- know the parameters of all of the pre-existing operational (or EUC) hazards that the new system would have to mitigate;
- assess all of the impacts that the change (i.e. the introduction of the new system) would have on the existing operating rules (e.g. signalling principles) for the railway, captured in what IEC 61508 calls *Overall Safety Functions / Overall Safety Requirements.*

However, there is a real danger here that too literal an interpretation of the term SuC would lead to a mindset that the hazards to be assessed would be those caused by the SuC — in effect, the problem Leveson (2001) consistently warned against and reaffirmed more recently in Leveson (2020). As we will see in, *inter alia*, the case history outlined in section 4.4.3 below, we could easily end up with a set of system safety requirements that specified only a tolerable failure rate for the system, and not also the functionality that the system would need into order to mitigate the pre-existing, railway-level, operational hazards.

Of course, this is all about assurance activities, and perhaps, therefore, the question should be, not whether the processes in EN 50126 *et alia* comply with the corresponding processes of IEC 61508, but whether, by careful interpretation of the system under consideration and adherence to the standards themselves, the results would then produce assurance evidence sufficient to stand up to the demands of a rigorous safety argument. It follows, therefore, that this question would best be addressed.

1.3 Scope

Like its predecessors, this article focuses mainly on the crucial, early phases of the IEC 61508-1 safety lifecycle — i.e. on the *Specification* of Overall Safety Requirements (Phases 1 to 4), and the high-level *Design* of the System (Phases 5, 9 and 11). This is because most of the key principles underpinning IEC 61508 — i.e. those universal principles set out in Part 1 of the Standard, which govern the determination of the required risk-reducing properties of Safety-related Systems — take effect during these earlier phases, whereas the subsequent implementation and operating phases are less specific to the Standard.

Indeed, the truth of this assertion is borne out by Leveson (2020), as quoted in Sub-section 1.2.3 above and, in Sub-section 4.4 below, by the three past examples of where the ignorance of the above principles appears to have led to disastrous consequences!

1.4 Aims and Objectives

The aim of this paper is to develop a generic Safety Assurance structure, against which the efficacy of transport system safety-assessment standards could be judged.

This aim is addressed through the following objectives:

4. the development of an Argument-based Safety Assurance framework, compatible with the safety lifecycle of IEC 61508-1, but based predominantly on well-founded requirements-determination and systems-engineering practices, and focussed on safety-requirements specification and high-level design;
5. assessment of the strengths and shortcomings of IEC 61508-1 processes in the development of evidence to support the arguments for safety-requirements Specification and high-level Design; and
6. assessment of the efficacy of the illustrative products of the application of IEC 61508-1 to the ATM sector (Fowler and Fota 2023) and Rail sector (Fowler and Graebner 2023), in providing evidence to support the arguments for system safety-requirements Specification and high-level Design.

2 Safety Assurance Principles and Rationale

2.1 Safety Assurance Definition

Although the term "Safety Assurance" is widely used across many safety-related industry sectors, there does not appear to be a single, internationally accepted definition of the term. A recent AI-assisted search of its usage suggested the following, which is judged to serve the purposes of this article very well:

> *Safety assurance is the planned and systematic set of arguments, activities and evidence, which demonstrate, with justified confidence, that a product, service, organisation, or functional system is acceptably safe, in a given context.*

2.2 The Need for an Argument-based Safety Assurance Approach

In Annex 5 to its guidance on the introduction of AI into organisations (ICO 2022), the UK Information Commissioner's Office draws an interesting comparison between assurance-case approaches, "*… as have historically arisen in safety-critical domains …*", and which it strongly recommends for the introduction of AI into organisations, and what it calls "*the older, more reactive and prescriptive methods that are now increasingly challenged by the complex and rapidly evolving character of emerging AI technologies*".

It goes on to explain that "*An assurance case is a set of structured claims, arguments, and evidence which gives confidence that an AI system will possess the particular qualities or properties that need to be assured…*" and that this "*…would involve providing an argument,*

supported by evidence, that a system possesses the properties that will allow it to function safely, securely, reliably, etc., given the challenges of its operational context".

Whereas the "*older prescriptive approach stressed the application of one-size-fits-all general standards that specified how systems should be built and often treated governance as a retrospective check-box exercise*", it suggest that "*argument-based assurance takes a different tack, …from the inception of an AI project and plays an active role at all stages of the design and use lifecycle. It begins with high-level normative goals derived from context-anchored impact and risk-based assessment for each specific AI application and then sets out structured arguments demonstrating:*

- *how such normative requirements address the impacts and risks associated with the system's use in its specified operating environment;*
- *how activities undertaken across the design and deployment workflow assure the properties of the system that are needed for the realisation of these goals; and*
- *how appropriate monitoring and re-assessment measures have been set up to ensure the effectiveness of the implemented controls*".

It notes that "*… argument-based assurance cases have historically arisen in safety-critical domains … because it is a reasonable way to structure and document an anticipatory, goal-based, and procedural approach to innovation governance, … in contrast to the older, more reactive and prescriptive methods that are now increasingly challenged by the complex and rapidly evolving character of emerging AI technologies*".

It is for very similar reasons that EUROCONTROL had previously used argument-based safety assurance for the introduction of Reduced Vertical Separation Minima for aircraft across most of European airspace in 2002 (Fowler and Tiemeyer 2002), as has the European Commission's Single European Sky ATM Research & Development programme since its inception (Fowler et al. 2009).

It is also the reason for selecting such an approach for this article, for assuring the safety of other highly complex systems / operations!

2.3 Safety Assurance Model

Figure 1 is a Safety Assurance Model, which has been adapted from Fowler *et alia* (2009), and fits the above definition very well.

The green box represents the whole of the Argument structure, with all of its contextual annotations, each and every strand of which has to be provided with supporting Evidence to show that it is valid (i.e. true). An example notation for such a structure is described in Section 3 below.

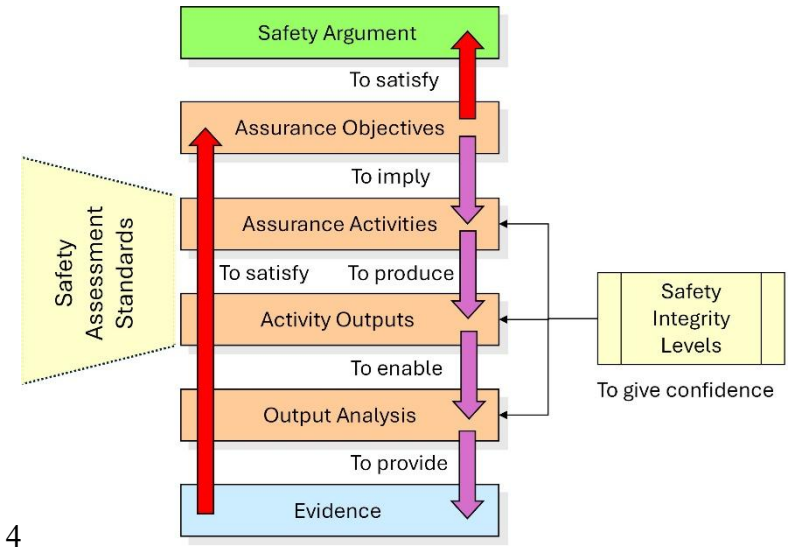


Figure 1~ Safety Assurance Model

Evidence is any artefact or item of information that supports the truth of an Argument and that is shown to be relevant, credible, sufficient, traceable, and verifiable.

Assurance Objectives state what the Evidence has to demonstrate in order to satisfy the associated Argument statement[11] — in effect, therefore, they are actually the lowest level of decomposition of the Argument. The Assurance Objectives specified in Sections 5 and 6 below were adapted in part from CAP670 SW01 (UK CAA 2019), which sets out a very comprehensive set of "*Regulatory Objectives for Software Safety Assurance in ATS*[12] *Equipment*".

The Assurance Objectives imply the *Assurance Activities*, i.e. the safety-assessment processes, that are necessary and sufficient to satisfy each Assurance Objective. The *Activity Outputs*, i.e. the results of completing the Assurance Activities, form the basis of the *Evidence*, subject to the addition of the results of Activity *Output Analysis* that show (with the appropriate degree of rigour) how the Evidence satisfies the related *Assurance Objective*.

Figure 1 also suggests that safety-assessment standards, such as IEC 61508, usually focus on Assurance Activities and Assurance Outputs, with some attention also to Output Analysis. IEC 61508-1 does *not*, however, address what are defined above as Assurance Objectives — its use of the term Objective, is very different in that, in most cases, they are phrased as things that have to be done, i.e. Assurance Activities, *rather* that things that have to be demonstrated.

A major positive contribution of IEC 61508 is its use of Safety Integrity Levels (SILs)[13] to define, *inter alia*, the *rigour* demanded of the system-development processes and the safety-assessment processes: Assurance Activities, Outputs and Analysis.

[11] Strictly speaking, the way they are usually worded does not preclude a further decomposition of the Argument should that prove necessary.

[12] Air Traffic Services

[13] Sometimes referred to as Safety Assurance Levels (SALs)

3 Developing an Argument-based Safety Assurance Structure

3.1 Structure and Notation

Safety assessments are often done within the context of a Safety Case which might, like a legal case, comprise two main elements:

- a set of arguments, i.e. statements which claim that something is true (or false); and
- supporting evidence to show that each argument is actually true.

Safety arguments are normally set out hierarchically. In this case, an *adapted form* of Goal-structuring Notation (GSN) has been chosen for the purpose, as shown in Figure 2. GSN is simply a graphical representation of an argument / evidence structure and usually starts with the top-level *claim* (Arg 0) that something is (or will be) tolerably (or acceptably[14]) safe; this is then decomposed such that it is true if, and only if, the next-level *argument* statements (in this case Arg 1 to 4) are themselves *all* true. The *strategy* text (St001) should explain the rationale for that decomposition.

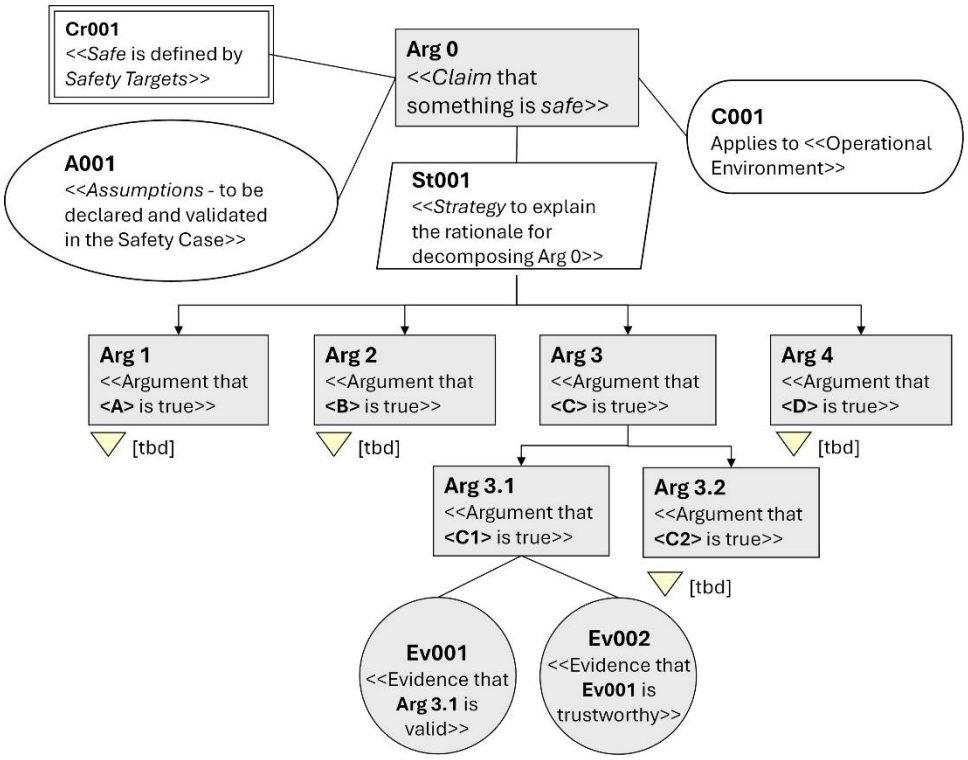


Figure 2 ~ A Generic High-level Safety Argument

The claim (Arg 0) is supported by vital contextual information, as follows:

- what is meant by *acceptably safe* is defined by means of *safety targets* (see *constraint* Cr001);
- the *context* (C001) for the claim must include a description of the operational environment for which the claim is being made — it will be seen in Sub-section 4.1 how critical this is to the validity of the claim;

[14] i.e. Tolerably safe plus ALARP (As Low as Reasonably Practicable) (UKMoD 2016).

- *assumptions* (e.g. A001) are sometimes used for material on which a claim or argument depends but the validity of which might be in some doubt; it is vital that such assumptions are shown to be reasonable and that they are logged and tracked through to the end of the safety lifecycle.

The arguments would then be further sub-divided until a level is reached at which documented evidence, of a manageable size, could be produced to show that the related argument is valid.

Figure 2 identifies two main types of Evidence, in support of Arg 3.1:

- Evidence that shows directly that a particular objective has been achieved — in this case, that Arg 3.1 has been satisfied — and is referred to as *Direct* evidence (Ev001);
- Evidence that shows that the Direct evidence is trustworthy, i.e. that it can be relied upon. This is referred to as *Backing* evidence (Ev002).

Appendix A hereto presents a comprehensive set of principles for developing safety-assurance structures and supporting evidence, which have been adapted from EUROCONTROL (2006b), and are recommended to be followed in the development of assurance structures (and have been followed herein!).

3.2 Transport Illustration of a Top-level, Lifecycle Safety Assurance Structure

Figure 3 is an illustration of a top-level safety-argument structure, and has been worded such that it could be easily adapted to most safety-related transportation upgrades.

It should be noted that the top-level Claim is deliberately made for an acceptable level of safety of the operations enabled by the System, not for the System itself — the necessity for this distinction should become clear in the detailed discussion in Section 4.

The question then is how to ensure that the next level of the safety argument would be complete and rigorous — for this, an approach is proposed that follows a reasonably conventional, 5-stage systems-engineering development process, and starts with a high-level safety argument for each stage, as follows (Fowler et al 2009).

Arg 1 — argument that the System (or system of systems) to be introduced has been specified to be safe, i.e. the overall safety requirements (or *Specification;* see Section 4) meet the appropriate *Safety Targets* for the Operational Environment completely and correctly.

Arg 2 — argument that the high-level *Design* of the System satisfies its *Specification*, correctly and completely.

Arg 3 — argument that the *Implementation* of the *Design*, in the built-and-tested System, satisfies the *Design* requirements correctly and completely.

Arg 4 — argument that the *Transition* from current signalling system to the new System will be safe, i.e. the known risks during this process have been reduced to a level that is As Low as Reasonably Practicable (UKMoD 2016).

Arg 5 — argument that the System will be shown to continue to operate safely throughout its Operational Service.

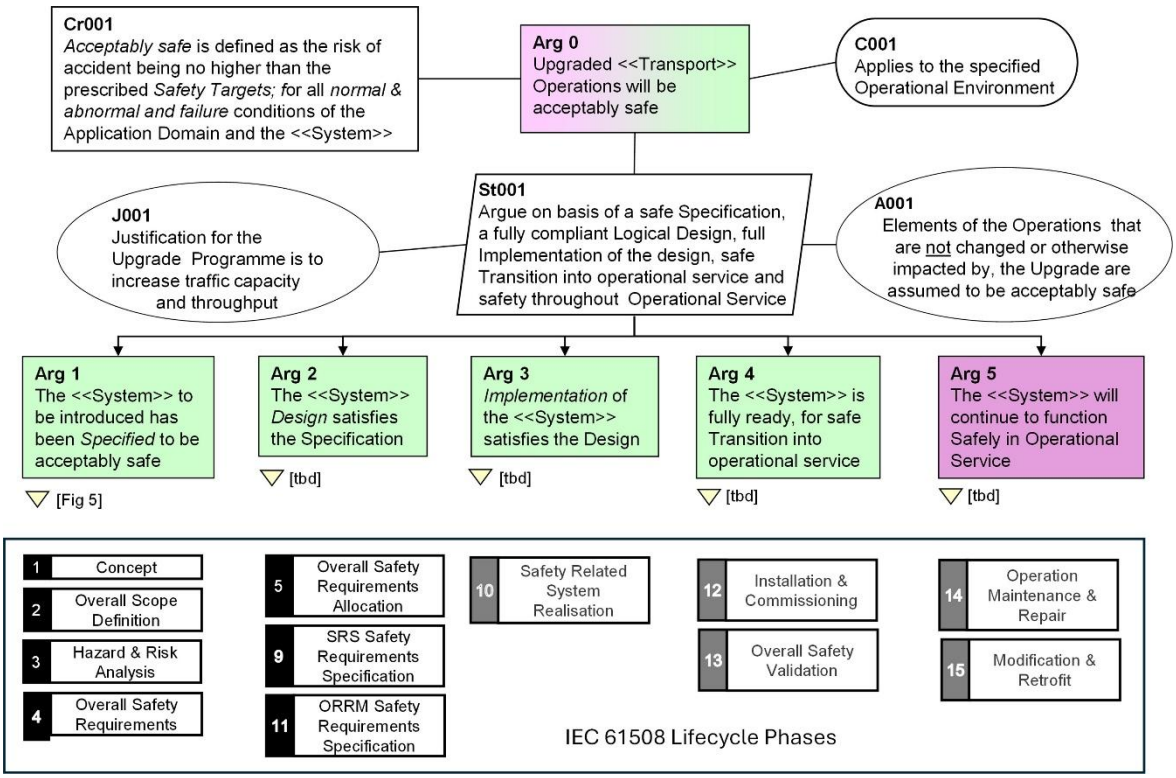


Figure 3 ~ Generic Top-level Safety Argument for a Transport Application

For ease of reference only, the general relationship with IEC 61508-1 safety-lifecycle phases (except for three planning phases and Phase 16-*Decommissioning*) is shown at the foot of the argument diagram, the phases that are outside the scope of this article being shown in grey. It is important to note, however, that the Safety Argument was *not* derived from those IEC 61508-1 phases, and its purpose is to provide a generic model for assessing the ability of transportation safety-engineering standards (and of IEC 61508-1 itself!) to provide incontrovertible direct and backing evidence of adequate system functional-safety properties.

In line with the scope of the three previous articles, the further decomposition of the argument, discussed in Sections 5 and 6 below, focuses entirely on Arg 1 and Arg 2, especially the former. Since those two arguments are based heavily on sound systems-engineering practices, they are described in Section 4 below, along with some examples of the pitfalls of not following them.

4 Systems Safety Engineering — Good Practice and Pitfalls

This section describes good practices in systems engineering on which the subsequent decomposition of Arguments 1 and 2 are based, and presents some real-life examples of the perils of ignoring them.

4.1 Requirements Engineering Concepts

In his seminal work on *The World and the Machine* (Jackson 1995), Dr Michael A Jackson set out crucial scientific principles regarding requirements capture. He summed up the perceived requirements-capture problem, as follows:

> *"We are concerned both with the world, in which the machine serves a useful purpose, and with the machine itself. The competing demands and attractions of these two concerns must be appropriately balanced. Failure to balance them harms our work".*

The concepts underlying his Problems Frame Approach were adopted by the then Praxis Critical Systems Ltd.[15] in the development of its REVEAL® Requirements Engineering method, including the simple, but rigorous, Requirements Engineering model shown in Figure 4 (Hammond et al. 2001).

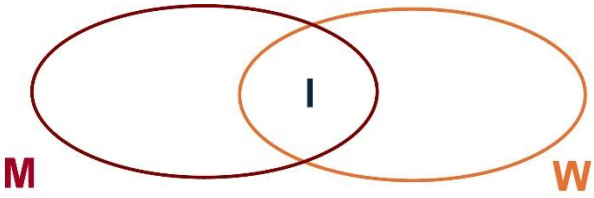


Figure 4 ~ The World and the Machine

Essentially, *Machines* (M) are built in order to bring about some improvement to the *World* (W). In these terms, a Machine can be any socio-technical artefact that is to be constructed, and need not be physical, e.g. it could be a process.

Whenever a Machine is introduced to a World, it interacts with that World through a shared *Interface* (I), and items that might occur in this interface are those phenomena that are shared between the World and the Machine; they are *shared* because they are in some sense visible to both the World and the Machine, i.e. they represent the Machine's view of the World, and vice-versa. It is through these shared phenomena that the Machine seeks to influence the behaviour of the World. Other shared phenomena could be controlled by the World and responded to by the Machine.

The following definitions are based on those provided by Jackson (1995):

- *Requirements* (ℛ) are a collection of statements about phenomena in the World that we want the Machine to *help* make true — the implication is that it is not solely the Machine that brings about the Requirements;
- a *Specification* (𝒮) is a collection of statements that describes a Machine's required external behaviour. A Specification statement refers only to shared phenomena in the Interface, and can only constrain those shared phenomena that the Machine itself could control;
- the *Application Domain* covers those aspects of the World that are relevant to the improvement that we wish the Machine to achieve, and its properties — that we know (or assume) to be true — are termed *Domain Knowledge* (𝒟).

[15] Later Altran UK Ltd and Capgemini.

This recognition of the rôle of Domain Knowledge in the Requirements Engineering process leads Jackson to suggest the following relationship, which must hold if a Machine is to be introduced to a World to bring about some Requirements:

$$\mathcal{D}, \mathcal{S} \vdash \mathcal{R}$$

Hammond *et alia* (2001) call this a "*satisfaction argument*", which should be read as follows:

> *"Using the relevant properties of the Application Domain ($\mathcal{D}$), when combined with the Specification ($\mathcal{S}$) of the behaviour of the Machine to be constructed, it is possible to show ($\vdash$) that the Requirements ($\mathcal{R}$) will hold",*

and is the foundation of Arg 1, as discussed in Section 5 below.

4.2 Rich Traceability

Of course, we must demonstrate that the $\mathcal{D}, \mathcal{S} \vdash \mathcal{R}$ does actually hold true in practice and, for this, Hammond *et alia* (2001) recognise that a traditional, basic requirements-traceability approach would not be adequate because it would simply link:

> *"… items from one artefact to the corresponding items from a previous artefact, or to the source of the items. For example, a specification statement may be linked to all the requirements that it helps to satisfy. However, there is no explanation or justification of the links and so their semantics is often unclear. Typically, all that a link between two items can suggest is that changes in one may require changes to the other".*

Instead, Hammond proposes a "Rich Traceability" approach, "*… which focuses on **how** an item is satisfied, by enhancing the use of simple links with additional structure and justification*".

In essence, *rich traceability* is a more comprehensive, multidimensional way of capturing, linking, and analysing traceability information across the lifecycle of a product, system, or process, by:

- capturing different types of artifacts (requirements, design, code, test cases, risks, regulations, defects, etc.).
- supporting multiple relationship types (not just "*satisfies*" or "*tests*", but also "*conflicts with*", "*depends on*", "*refines*", "*duplicates*", etc.).
- providing contextual meaning, enabling you to understand why the link exists, what kind of relationship it is, and how strong or weak the relationship is.
- allowing for bidirectional navigation and impact analysis — so if one artifact changes, one can see all the related artifacts and assess ripple effects.
- use of visualizations (graphs, matrices, networks) instead of flat lists, helping teams spot gaps, overlaps, and dependencies.

These are very important principles, which are fundamental to any well-structured safety argument. GSN does provide for this (Dick 2002) to some extent but as it is only a notation, it needs a set of rules to supplement it — hence, for example, the good-practice principles for structuring safety arguments and evidence, set out at Appendix A hereto.

4.3 Application to Safety Requirements

Figure 5 below is from Fowler and Pierce (2012) and shows how the Jackson (1995) model of Figure 4 above can easily be applied to a safety environment, as follows:

- the *Application Domain* now covers the IEC 61508 concept of an *EUC*, and the *Operational Environment* in which it is deployed — each has its own *Domain Knowledge* (𝒟), which we know (or assume) to be true;
- the term *System* (comprising equipment, people and procedures) replaces the *Machine*;
- the primary Satisfaction Argument (𝒟 , 𝒮 ⊢ ℛ) remains the same, but a Design argument (𝒜 ⊢ 𝒮) has been added[16]; and
- although not shown explicitly, the *Requirements* would include the basic safety targets that would need to be satisfied in the *Operational Environment*, e.g. tolerable maximum risk of accidents and/or hazards in the *Operational Environment*.

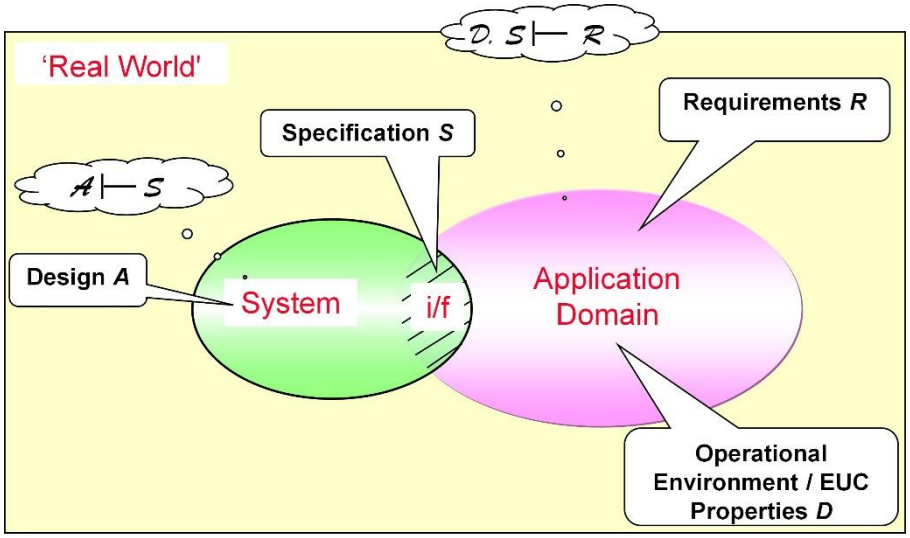


Figure 5 ~ Safety Interpretation of the Jackson Requirements-capture Model

It is this model that is used in developing Argument 1 and Argument 2, in Sections 5 and 6, respectively, below.

Before that, however, it would be worth pausing to consider three egregious past examples of where the ignorance of the above principles and practices appears to have led to disastrous consequences.

4.4 Some Lessons from History

The examples chosen to illustrate in this Sub-section the perils of ignoring the above principles and practices are:

- the Ariane 5 rocket explosion;
- the Überlingen mid-air collision; and
- the failure of a major London Underground Ltd. (LUL) re-signalling project.

[16] 𝒜 is Design; 𝒟 has already been used for Domain Knowledge.

It is important to note, however, that *none* of these case studies are intended to be complete discussion of everything that went wrong — rather, they are focussed on those causal factors that can be seen to have strong links with the themes in this article.

4.4.1 Ariane 5 Rocket Explosion

This first example involves the maiden flight of the Ariane 5 rocket (Ariane 501), on 4th June 1996, which failed catastrophically within 40 seconds after initiation of the flight sequence, and which incurred a direct cost of at least $370 million.

According to Dowson (1997) the sequence of events can be summarised as follows:

- *"An outside-expected-range value, derived from the launcher's actual horizontal velocity, led to an 'unhandled exception' in the Inertial Reference System (SRI[17]) software. This caused the backup "hot standby" SRI to fail approximately 37 seconds after launch initiation (H0).*
- *Seventy-two milliseconds later, the active SRI (running identical software) failed for the same reason.* [a common-mode failure]
- *In the absence of a working backup, the failure of the active SRI caused it to transmit diagnostic bit patterns to the main On-Board Computer.*
- *The On-Board Computer, misinterpreting the diagnostic input as valid data, ordered full nozzle deflections of the solid boosters and the Vulcain main engine.*
- *These deflections created aerodynamic loads that separated the boosters from the main stage, (correctly) triggering the launcher's self-destruct system at approximately H0+40 seconds".*

The Inquiry Board Report (Lions 1996) concluded, in essence, that poor software engineering practices were responsible for the "*software failure*" that caused the catastrophe; however, both Mark Dowson (1997) and Nancy Leveson (2001) disagree with the IRB's use of "*software failure*", since the software did exactly what it had been designed to do. Dowson (1997), in particular, offers a "*closer analysis of the Inquiry Report*" that "*reveals a rather different picture*", as follows:

7. "*... the early part of the flight trajectory of Ariane 5 differs from that of Ariane 4 and results in considerably higher horizontal velocity values.*" (Inquiry Board Report, page 4).
8. "*The design of the Ariane 5 SRI is practically the same as that of an SRI which is presently used on Ariane 4, particularly as regards software.*" (Inquiry Board Report, page 3). "*...As should now be obvious, the software functioned precisely according to its specification and design, and no additional software-related measures — such as ensuring that all code explicitly included a handler for "others" conditions, improved QA, e.g. better test coverage, or use of improved approaches to software fault tolerance — could have ensured that some similar catastrophe did not occur.*"

Dowson then goes on to say that there are "*then, ... two astonishing aspects to the Inquiry Report*"; *viz.*:

[17] French initialism

9. *"… the report reveals that the expected flight trajectory of Ariane 5 was **deliberately** excluded from consideration in the design of a key component involved in its control; and*
10. *"… the Inquiry Board (which included software engineers) characterized the cause of the consequent flight failure as a software problem.*

"Unfortunately, it is all too easy to imagine how the first of these happened in an industrial context: pressures of time and budget, if-its-not-broke-don't-fix-it arguments, etc. The second is harder to comprehend, and perhaps it is wiser not to speculate on its causes without better insight into the procedures followed by the Inquiry Board.

*So, how should we characterize the causes of failure of Ariane flight 501, and how can future such failures be avoided? At the very least, the failure was a systems engineering failure. As we all know, software can't be designed in isolation, and the system that it is part of includes its operating environment. …A well-defined (and well designed) process for engineering systems such as Ariane 5 should make it impossible to exclude any significant aspect of the system **operating environment** from consideration. …A good process needs to regulate not only how systems are designed and developed, but also how high-level decisions about that design and development are arrived at. Only by adopting and following a process that encompasses these considerations, will sound engineering practice be able to triumph over immediate pressures of time and budget".*

In other words, failure to understand that, even if the *Requirements*, *Specification* and *Design* of the software for Ariane 5 did not change from those of Ariane 4, **changing the properties of the Application Domain,** i.e. of the rocket, could (and DID!) negate the satisfaction arguments $\mathcal{D}$, $\mathcal{S} \vdash \mathcal{R}$, with catastrophic results!!

4.4.2 Überlingen Mid-air Collision

According to the subsequent accident-investigation report (BFU 2004), a Boeing 757 (B757) cargo aircraft flying north, in cruise, at flight level (FL) 350, requested FL 360, and was incorrectly cleared to climb to this level. This created a crossing conflict with a Tupolev Tu-154 passenger aircraft[18] flying on an approximately westerly track at the same flight level. For various reasons, detailed by BFU (2004), the Air Traffic Controller in the Zürich centre did not notice this conflict for some time. When the Controller did eventually notice the conflict, he instructed the Tu-154 aircraft to descend to FL 350 thus: "*BTC 2937…descend flight level…350, expedite, I have crossing traffic*". Seven seconds after the start of this instruction, the ACAS on each aircraft issued a Resolution Advisory (RA). The B757 received a descend RA, and the Tu-154 received a climb RA, i.e. in the opposite direction to the Controller's instruction. The B757 flight crew promptly obeyed their RA while the Tu-154 flight crew, after a short on-board discussion, chose to obey the Controller's instruction.

The B757 flight crew reported their RA (badly!) 23 seconds after the RA was issued, and 16 seconds after the Controller had issued a second (more urgent) instruction to the TU154 to "expedite descent". The Tu-154 flight crew did not report the RA, but did acknowledge the Controller's second descent instruction, in response to which the Controller confirmed that "*… we have traffic at your 2 o'clock position, now at* [FL]*360*". Both aircraft continued to descend and collided. Around 20 seconds before the collision, the B757 ACAS issued an

[18] Forty-six of the passengers on the TU154 were Russian schoolchildren on a school trip organised by a local UNESCO committee to the Costa Daurada beach area of Catalonia. in Spain.

increase descent RA and at 8 seconds the Tu-154 ACAS issued an increase climb RA. At no stage during the sequence did ACAS generate a reversal command to either aircraft.

BFU (2004) correctly focussed more on the many egregious failings on the part of the responsible Swiss authority (skyguide[19]), the management at the Zurich Control Centre, and the (sole) Controller on duty at the time, than on why the ACAS, as a "socio-technical system", failed to prevent the accidents despite the fact that the technical functioning of the ACAS *equipment* in both aircraft had been correct throughout. It is the latter question that occupied the minds of EUROCONTROL experts after the accident; leading to publication of many reports, including the final version of an argument-based ACAS Safety Case (EUROCONTROL 2011), which concerns us here.

During the development of the Safety Case, it became clear that ACAS operations were defined in a considerable number of disparate documents promulgated by ICAO, other international / national regulatory and standards bodies, ATM service providers, aircraft operators, and manufacturers (Thomas and Fowler 2013).

The problem was that, even at the highest level of definition, each of these documents captures only an individual aspect of ACAS operations. Consequently, there existed no conceptual description of ACAS operations as a whole that could be used as a starting point for assuring that the detailed provisions were themselves correct, coherent and consistent with each other, or for verifying that any future changes to ACAS operations would be compatible with the original intent of the system — in the terms of this article, what was needed, but was t*otally absent*, was a *Specification for ACAS*.

The solution was to *reverse-engineer* the ACAS document set to yield a coherent definition of ACAS operations, including a set of basic operational rules, some examples of which are shown in Table 1 below (Thomas and Fowler 2013).

Table 1 ~ Examples of ACAS Basic Operational Rules

Ref.	Safety Objective
1	In the event of an imminent mid-air collision, one or both of the aircraft involved shall initiate a vertical manoeuvre such that the collision is avoided
2	Collision-avoidance manoeuvres shall be initiated in a timely manner and in a way that does not generate positive or negative 'g' forces that would over-stress the airframe or injure the aircraft occupants
3	Collision-avoidance action shall be unaffected by interference from air traffic services
4	Collision-avoidance action shall not interfere with the provision of air traffic services to non-involved aircraft
5	Collision-avoidance action shall take priority over the need to resolve situations that present a lower risk to the aircraft

An important part of the safety case work was, therefore, to check compliance of the existing ACAS design — especially the operational procedures — against the rules. Unsurprisingly, given that those procedures were written without the benefit of the rules, i.e. a Specification, the analysis revealed both a number of serious non-compliances with the rules; and inconsistencies within the procedures themselves — two examples of which are as follows:

[19] Swiss Air Navigation Services Ltd.

Firstly, the fact that, although Controllers were told not to intervene in the event of an RA (ICAO 2007), the only way they could know that an RA was happening was by a report from the Flight Crew, and yet such a report was low down in the Flight Crew's priorities during an RA event (ICAO 2006).

Secondly, Flight Crew procedures (ICAO 2006) state, in Section 3.1.2, that:

"*Nothing in the procedures specified in section 3.2 …shall prevent pilots-in-command from exercising their best judgement and full authority in the choice of the best course of action to resolve a traffic conflict or avert a potential collision*",

whilst section 3.2 itself clearly states that:

"*… in the event of an RA, pilots shall:*

> *"1) respond immediately by following the RA as indicated, unless doing so would jeopardize the safety of the aeroplane;*
>
> *"2) follow the RA even if there is a conflict* [sic] *between the RA and an air traffic control … instruction to manoeuvre".*

Sadly, Überlingen was not an isolated incident. Just 18 months before the Überlingen collision, a conflict occurred between two Japan Airlines (JAL) aircraft, a Boeing 747 climbing outbound from Tokyo to Okinawa, and a DC-10 inbound to Tokyo from South Korea near the city of Yaizu. In response to an STCA[20] alert, the air traffic controller instructed the B747 to descend, about three seconds before ACAS RAs were issued in both aircraft. The ACAS equipment on board both aircraft functioned correctly, giving a climb RA to the descending B747, and a descend RA to the DC-10. The DC-10 flight crew obeyed the descend RA, but the B747 crew continued its descent, in accordance with the prior air traffic controller instruction, but in contradiction to the climb RA (EUROCONTROL 2022). Fortunately, a mid-air collision was avoided, but only by a few feet!

EUROCONTROL (2011) poses the rhetorical question: "*Would it be unreasonable to suggest that the actions of the B747 pilot in the Yaizu incident and of the TU154 pilot in the case of the Überlingen accident … as falling within what is permitted by PANS-OPS section 3.1.2 rather than contravening PANS-OPS section 3.2?*"

From the way that the above rules are expressed (especially their complete independence from the technical solution), it should be apparent that, in terms of this article:

- the Rules would form a major part of the *Specification*, if one had existed;
- the Aircraft in the airspace (*Operational Environment*) would be the EUC; and
- Air Traffic Services would be part of the *EUC Control System*.

The problem was that ACAS had been brought into service, and had spent many years *in* service, without the benefit of a *Specification*, seemingly allowing operating procedures, i.e. a vital part of the wider-system *Design*, to be developed without an appropriate degree of rigour.

It has to be said, though, that whilst the post-Überlingen safety analysis produced a set of rules, which enabled inconsistencies in the procedures to be revealed, we will see, in Sub-section 5.4 below, that it is crucial that analysis techniques be applied in order to ensure that the *Specification* itself is produced with the appropriate degree of rigour, in terms of its correctness, completeness, and coherency. Unfortunately, it seems that ACAS was

[20] Short-Term Conflict Alert (STCA) is a radar-based tool that warns air traffic controllers of aircraft being too close to each other.

introduced by the Commercial Aviation sector without the benefit of a proper safety assessment, on the basis that — as a safety net, of last resort — it would be better to have ACAS than not have it at all. Such a naïve view would of course overlook the possibility of ACAS creating a new *system-generated* hazard, due to dysfunctional operation of the wider system — a hazard that would not otherwise have occurred.

A clear case of failing to appropriately balance "*the competing demands and attractions*" of the world and the machine (Jackson 1995), with tragic consequences.

4.4.3 Railway Signalling Contract Abandonment

4.4.3.1 Background

The following is a direct quote from an article in the International Railway Journal, of 3rd January 2014 (IRJ 2014):

> *"London Underground (LUL) and Bombardier Transportation have announced that a contract for re-signalling of the* [Sub-Surface Railway (SSR)] *comprising the Circle, District, Hammersmith & City, and Metropolitan lines with communications-based train control (CBTC) will be relet"*
> *"Bombardier* [had been] *awarded the £354m contract in June 2011 and was expected to supply its Cityflo 650 moving block automatic train control system which it has already successfully deployed on lines 1 and 6 of the Madrid Metro, increasing capacity by 30%...".*
> *"However, Transport for London (TfL) says* [in Jan 2014] *as the work progressed it became clear that the fit between the system offered and the Underground network is such that the contract is not going to deliver what is needed, in the time required, while remaining cost-effective. "We have been working closely with Bombardier to find a way forward on what is one of the most challenging and complex pieces of work on the Tube," says Mr Mike Brown, LUL's managing director. "However, it has become apparent to both parties that for the work to be completed within or close to the planned deadline, we need to push on with works with another contractor…."*

In August 2015, LUL announced the award of a £760m contract to the Thales Group to "*modernise the signalling and train control system on* [LUL's] *Circle, District, Metropolitan and Hammersmith & City lines*" (IRJ 2015).

In March 2016 the London Assembly, Budget and Performance Committee, issued a scathing report (London Assembly 2016) on the project failure, which, in the words of its chairman (John Biggs) concluded that, inter alia:

"TfL has grossly mismanaged its signalling contract with Bombardier, which means that anyone taking the District, Circle, Metropolitan, or Hammersmith & City lines will have to wait five years longer than originally expected to see the improvements. And there is a staggering cost associated with this mismanagement, which leaves TfL with £886 million less to spend on its capital programme than it thought it had".

As a systems-safety engineer of nearly 20 years' experience, 18 months of which was as a member of the LUL SSR project safety-engineering team, prior to the cancellation of Bombardier's contract, the author of this article makes no comment on the above political

and commercial aspects of the project but does offer the following personal insight into the technical problems that beset the project, from the perspective of this article[21].

4.4.3.2 A Safety-engineering Viewpoint

It is self-evident that the primary purpose of any railway signalling system is to prevent accidents, and that the primary properties of such a system that determine its ability to prevent accidents are those related to its *functionality and performance*. It should therefore be equally self-evident that producing assurance of correct functionality and performance of a safety-related system is a vital part of any system safety assessment; it is also entirely consistent with the concept of safety functions in IEC 61508-1 and was a specific requirement of rail safety standard EN 50129, at the time, but *only* of Annex B, section B.2 thereto.

Of course, in specifying and designing a signalling system, it is necessary also to ensure that the (extremely low) likelihood of a signalling system causing, or failing to prevent, accidents is acceptable. The primary properties of a signalling system that determine this likelihood are its *reliability and integrity*; producing assurance of this is also a requirement of the above IEC and CENELEC standards.

Bombardier's 2011 SSR re-signalling contract with LUL was founded on supply of its CITYFLO650 system, with a safety-assurance strategy that seemed (from Bombardier's documents at the time) to be based on the following, unsubstantiated assumptions:

- that a system designed for use, and deployed, on Shenzhen and Madrid metros would meet the uniquely complex needs of the SSR with "*some upgrades and enhancements*";
- that evidence based on the development and in-service experience of Shenzhen and Madrid products would be of sufficient scope and quality to support *cross-acceptance* arguments for application to the SSR;
- that compliance with (generic, *process-based*) CENELEC standards would provide sufficient safety assurance for CITYFLO650-enabled, moving-block operations across the SSR network;
- that evidence of the functional correctness of a SIL 4 system design can be based almost entirely on a simple DOORS-based traceability process, and testing, without rigorous design analysis.

By 2013, it was becoming increasingly clear, to LUL's SSR safety team at least, that none of the above assumptions were valid, and that, in relation to the thrust of this article, Bombardier's approach to the project was seriously flawed, as follows:

Firstly, Bombardier had little or no understanding of the unique complexity of the SSR *Application Domain*, i.e. the EUC, its *Control System* (the existing railway infrastructure) and their *Operational Environment*, compared with the very basic Madrid and Shenzhen metros, until LUL (somewhat belatedly), produced a detailed description thereof.

Secondly, in accordance with its contract, Bombardier developed a very detailed structured Safety Argument, for the introduction of CITYFLO650 to the SSR. That was a somewhat strange responsibility to place on an *equipment supplier* since the responsibility for the assuring the safety of SSR *operations* would have lain with LUL as the service operator.

[21] This insight is a summary of a highly critical technical report, written by the author of this article, and submitted to LUL two months before the Bombardier contract was cancelled.

Worst still, the Safety Argument was based almost entirely on compliance with the processes of the extant CENELEC rail safety standards, with the result that:

- it contained no valid claims for the functional safety of the CITYFLO650 *products*;
- it was based on safety-assessment processes that addressed only the reliability of the CITYFLO650 products, to the exclusion of their safety functionality and performance;
- it failed to recognise the crucial role that the properties of the *Application Domain* play (or should play) in a system safety argument (see Arg 1.2 at Sub-section 5.3 below).

Thirdly, apart from some high-level operational requirements provided by LUL, there was no equivalent of a *Specification* (see Arg 1.3 at Sub-section 5.4 below) or high-level Design Analysis (see Arg 2 at Section 6 below) — in fact, no functional-safety requirements existed, at any level, for the application of the CITYFLO650 system to the SSR. Bombardier did eventually produce a detailed set of textural functional-requirements statements that might have been applicable at the physical-Design level, but it was entirely devoid of any analysis and any requirements covering system behaviour.

Fourthly, much of Bombardier's product-level safety-assurance strategy seems to be based on the *Cross-acceptance* of individual products from their Shenzhen and/or Madrid baselines. Whilst such an approach was allowed in principle under CENELEC standards, the requirements for cross-acceptance, as laid down in guidance document CLC/TR 50506, sections 4.4.1a) to d), f), and g), were so demanding that it is unlikely that any of Bombardier's products (with the possible exception of *Ebilock* interlocking) would have been suitable for Cross Acceptance. Indeed, an independent "gap analysis" of CITYFLO650 software in 2012 regarding the SIL 4 RATP and VATP sub-systems' software from the Shenzhen baseline, revealed critical shortcomings in the software-development processes.

In 2012-13, in a serious attempt to correct these *ab initio* flaws, LUL's SSR project safety team set out to:

- produce its own, rigorous Safety Argument for the introduction of CITYFLO650-enabled, moving-block operations on the SSR. In outline, this argument was similar to that presented herein, but was further decomposed in full detail, across the whole of the safety lifecycle;
- produce a detailed description of the *Application Domain* for the SSR, which showed *inter alia* that, because of track sharing between the SSR, other LUL lines and some Network Rail lines, the CITYFLO650 system would have to: be rolled out across 12 so-called Migration Areas; accommodate 8 different train types; support 5 different ATP systems; and interface with 6 different signalling systems, across at least 15 signalling boundaries; and
- carry out an operational hazard analysis and develop, what are known herein as, a full Specification and a technology-independent, high-level Design, addressing more than 100 operational scenarios and covering *normal*, *abnormal* and *failure conditions*. The article by Fowler and Graebner (2023) is based almost entirely on that work.

Regrettably, all this came too late to save the project and served only to highlight the yawning gap between the system offered by Bombardier and what the SSR required in order to meet its operational and safety needs, leading to cancellation of the contract on 31st December 2013.

It could be said that, from the outset, LUL's and Bombardier's approach to the project seems to have been focussed mainly on how the existing CITYFLO650 system could be used on

the SSR, and nowhere near enough on first understanding what that system would have to do in order to meet the operational and safety needs of the SSR — this imbalance was completely at odds with the warning of Jackson (1995) that "*Failure to balance* [the competing demands and attractions of the World and the Machine] *harms our work*" — quite an understatement, in the context of LUL's loss of £886 million for zero benefit to its business and its customers (London Assembly 2016)!

In Sections 5 *et sequitur*, Arguments 1 and 2 are developed in more detail, based on the principles and lessons discussed above, and the results used to assess the safety-assurance role and capabilities of IEC 61508-1 in the specification of safety-related systems for the transport sector.

5 Decomposition of Arg 1

5.1 Preamble

Unsurprisingly, the first level of decomposition of Arg 1 comes straight from the *satisfaction argument* of Hammond *et alia* (2001), set out in Sub-section 4.1 above, and the logical relationship $\mathcal{D}, \mathcal{S} \vdash \mathcal{R}$ first proposed by Jackson (1995).

For that relationship to hold, and thus for Arg 1 to be true, we must show that each of the three sub-arguments, as captured in *adapted* GSN form in Figure 6, is true. These sub-arguments are discussed in detail in Sub-sections 5.2 to 5.4, respectively.

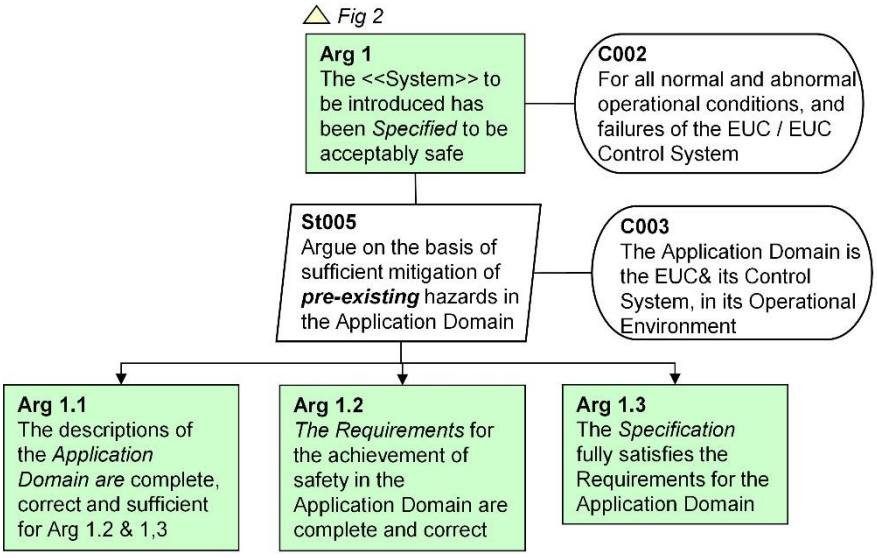


Figure 6 ~ First Level of Decomposition of Argument 1

It is important to note that the $\mathcal{D}, \mathcal{S} \vdash \mathcal{R}$ relationship is three-way, and that the process of satisfying Arg 1, through satisfaction of sub-arguments Args 1.1 to 1.3, is necessarily iterative — see also the discussion on rich traceability in Sub-section 4.2 above.

5.2 Arg 1.1 Domain Knowledge — Completeness & Correctness

5.2.1 Strategy

The *Strategy* for satisfying Arg 1.1 is to provide arguments and evidence which show that the properties of the *Application Domain* have been identified and documented completely, correctly and sufficiently, to support the satisfaction of Arg 1, at this stage — and later to support the satisfaction of Arg 2 to Arg 5.

In IEC 61508-1 terms, it would be necessary to show that inherent properties of the following Application Domain entities have been described fully and correctly, and are sufficient to support the *Requirements* in Arg 1.2 and the *Specification* in Arg 1.3:

- the EUC, e.g. the movement of aircraft through the airspace (Fowler and Fota 2023) or the movement of trains around the railway (Fowler and Graebner 2023);
- the Operational Environment and the EUC Control System; and
- all of the national and international safety regulations that are applicable to the subject of the overall Safety Argument.

5.2.2 Assurance Objectives for Arg 1.1

Credible arguments and evidence shall be provided to demonstrate the achievement of each of the ten Assurance Objectives defined in the pair of tables below.

Table 2 ~ Direct Assurance Objectives for Arg 1.1

ID	What has to be Demonstrated
1.1.1	The Domain properties address all the needs of the system Specification
1.1.2	The properties, and their attributes, accurately and completely reflect the actual (or assumed) properties of the Application Domain
1.1.3	Each Application Domain property description includes either: • a specification for each of the appropriate attributes of the property, or • a valid justification that the attribute is not applicable
1.1.4	The properties, and their attributes, form a coherent set and are specified in sufficient detail and clarity to enable the system Specification, Design and Implementation to achieve the required level of safety

Table 3 ~ Backing Assurance Objectives for Arg 1.1

ID	What has to be Demonstrated
1.1.5	The form of representation of the properties is appropriate to the Application Domain and allows an adequate determination of the Domain Requirements and/or system Specification and subsequently of the system high-level Design and Implementation, as applicable

ID	What has to be Demonstrated
1.1.6	The methods and techniques used are appropriate for the type, and safety-integrity level, of the properties / attributes concerned
1.1.7	The methods and techniques used are capable of supporting the identification of all properties of the Application Domain under all conditions: i.e. *normal*, *abnormal* and *failure* states of the EUC / EUC Control System
1.1.8	The methods and techniques have been applied by suitably qualified and experienced staff *Note: staff are deemed to be suitably qualified and experienced if they understand the methods used, are experienced in using them, and understand the required properties / attributes, and the operational and system contexts*
1.1.9	Any tools, used in the identification, description and configuration management of the Application Domain properties, have been verified and validated, as appropriate to the safety-integrity level of the properties concerned
1.1.10	A process that is independent of the means by which the properties were derived in the first place has demonstrated the validity of the properties, as appropriate to the safety-integrity level of the properties concerned.

5.2.3 Supporting Evidence for Arg 1.1 from IEC 61508

IEC 61508-1 states, in its subclause 7.2.1, that the "*objective* [sic] *of the requirements of this subclause is to develop a level of understanding of the EUC and its environment (physical, legislative, etc.) sufficient to enable the other safety lifecycle activities to be satisfactorily carried out*".

It then sets out three "requirements" [sic] that are relevant here [22]:

> *"7.2.2.1 A thorough familiarity shall be acquired of the EUC, its required control functions and its physical environment.*
> *7.2.2.4 Information about the current safety regulations (national and international) shall be obtained.*
> *7.2.2.6 The information and results acquired … shall be documented".*

In assurance terms, this is quite weak. Whereas some indication is given about what needs to be done, i.e. the Assurance *Activities* involved in deriving the Application Domain properties, there is nothing in the IEC 61508-1 text (subclause 7.2.1) that requires the completeness and correctness of those properties to be demonstrated, as required by the much more stringent Assurance *Objectives* of Sub-section 5.2.2 above.

5.2.4 An ATM Perspective on Arg 1.1

Sub-section 3.1 of Fowler and Fota (2023) provides a summary of the range of data that would be required initially in order to characterise the Operational Environment, the EUC and the EUC Control System for the Point Merge operational concept. Because that ATM application is a generic view of Approach operations, rather than a case study for a specific

[22] Three other requirements are set out in IEC 61508-1, subclause 7.2.2; however, as they refer to the identification of hazards, they are covered herein under Arg 1.2.

airport, the range of data is necessarily illustrative; however, the reference (*unlike* IEC 61508-1) does emphasise the need for the Application Domain data, and their properties, to be verified for completeness, correctness, consistency and sufficiency to support Arg 1.3 and, subsequently, Arguments 2, 3, 4 and 5.

With those provisos, Sub-section 3.1 of Fowler and Fota (2023), offers a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.1 for a typical new ATM operational concept. In reality of course:

- the actual data would need to be more detailed and specific to the airport concerned. However, for the purposes of satisfying Arg 1, only data related to the Specification would be required at this stage in the safety-assessment process — data related the Design and the Implementation of the System would be required later, in support of Arg 2 and Arg 3 respectively; and
- the completeness, correctness, and consistency of the actual data would need to be demonstrated, to the level of integrity — in IEC 61508 terms, the Safety Integrity Level (SIL) — required of the data, in the System context.

Furthermore, all data would need to be maintained, throughout all phases of the safety lifecycle outlined in Figure 3 above.

5.2.5 A Rail Perspective on Arg 1.1

Sub-section 4.1 of Fowler and Graebner (2023) provides a summary of the range of data that would be required initially in order to characterise the Operational Environment, the EUC and the EUC Control System. Because the Railway-signalling application is a generic view of Moving Block operations, rather than a case study for a specific railway, the range of data is necessarily illustrative; however, the reference (*unlike* IEC 61508-1) does acknowledge the crucial need for the Application Domain data, and their properties, to be verified for completeness, correctness, consistency, and sufficiency in support of Arg 1.3 and, subsequently, Arguments 2, 3, 4 and 5.

It should be noted, though, that there is an error in Sub-section 4.1 of Fowler and Graebner (2023), in that track and station parameters should have been listed under the "EUC Control System" in Sub-section 4.1.4, rather than as properties of the (Operational) Environment in Sub-section 4.1.3.

With those provisos, Sub-section 4.1 of Fowler and Graebner (2023), provides a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.1 for a typical new moving-block railway-signalling system.

In practice the full data set would be huge, and would probably include items that would require a SIL 4 attribution; and assurance of the completeness, correctness, coherence, and integrity of the whole data set, throughout the development and operational life of the CBTC System, would be a significant challenge.

5.3 Arg 1.2 Application Domain Requirements

5.3.1 Strategy

Traditionally, *Requirements* for the safety of operations are mandated by the relevant regulatory body, rather than by the responsible operating authority, and such *Safety Targets* would usually relate only to a maximum tolerable likelihood of Harm[23].

The usefulness of such targets, in *a priori* safety assessments is quite limited because of the *a posteriori* nature, and necessarily extremely low frequency of occurrence, of such harmful events. A far more practicable approach would be to place limits on the frequency of occurrence of the Hazards such that the regulatory Safety Targets for Accidents would thence be met. Furthermore, as IEC 61508-1[24] makes very clear, Hazards are associated with the EUC, and it follows that the tolerable rates of occurrence of the Hazards must fall within the Application Domain *Requirements*, not within the System *Specification* (Arg 1.3).

> ***Note:*** *Further justification for the inclusion of such pre-existing Hazards here can be seen from an example of a Train Derailment accident, for which Fowler and Graebner (2023) states that the hazardous precursor would be "HP#3 - Train speed exceeding capabilities of the track infrastructure…"; i.e. the occurrence of the hazardous state, HP#3, depends on* ***two*** *properties: the* ***speed*** *at which the train, i.e. the EUC, is travelling and the* ***physical properties*** *of the track (i.e. part of the EUC Control System), e.g. its curvature, that define its ability to accommodate a train travelling at that speed. It is asserted, therefore, that what IEC 61508-1 describes as EUC Hazards are actually emergent properties of the Application Domain, not properties of the System; the only exception to this are hazards caused by incorrect or spurious operation of the System — as happened in the case of the Überlingen mid-air collision, outlined in Sub-section 4.4.2 above.*

Therefore, the *Strategy* for satisfying Arg 1.2 is to provide arguments and evidence which show that:

11. a valid set of *Safety Targets*, which specify the maximum tolerable risk of an accident, have been determined, in accordance with national regulatory requirements;

    Note: The Safety Targets could be specified for all accidents together or for each accident type separately.

12. all potential hazards associated with the EUC, in its Operational Environment, covering all accident types, and within the scope of all potential risk-reduction measures, have been identified;
13. the probability that each Hazard, if not mitigated, would lead to an accident has been determined;
14. the tolerable hazard occurrence rate (THOR) for each and every identified Hazard, has been determined such that the corresponding Safety Target is met.

Hence the *Requirements* for the safety of operations in the Application Domain comprise simply a description of, and THOR for, each and every in-scope EUC Hazard.

---

[23] Defined in IEC 61508-4 as "*...death, physical injury or damage to the health of people or damage to property or the environment*".

[24] For example, in IEC 61508-1 subclause 7.4.3.2.

5.3.2 Assurance Objectives for Arg 1.2

Credible arguments and evidence shall be provided to demonstrate the achievement of each of the 15 Assurance Objectives defined in the tables below.

Table 4 ~ *Direct* Assurance Objectives for Arg 1.2

ID	What has to be Demonstrated
1.2.1	Safety Targets, which state the maximum tolerable risk of an accident, have been specified
1.2.2	The Safety Targets have been determined, in accordance with the relevant national regulatory requirements
1.2.3	The Safety Targets have been derived for all accidents as a whole, or for each accident type separately, as appropriate
1.2.4	All potential hazards associated with the EUC, in its Operational Environment, covering all accident types, and within the scope of all potential risk-reduction measures, have been identified
1.2.5	The consequences associated with each of the identified Hazards have been determined in terms of the potential accident outcome(s)
1.2.6	The probability of each unmitigated Hazard leading to an accident has been determined
1.2.7	The hazard-analysis process has addressed all reasonably foreseeable circumstances
1.2.8	The hazard-analysis results are sufficient to support the derivation of the Application Domain safety *Requirements* and the System safety *Specification*
1.2.9	For each and every identified Hazard, a THOR has been derived such that the Safety Target for the related Accident(s) could be met

Table 5 ~ *Backing* Assurance Objectives for Arg 1.2

ID	What has to be Demonstrated
1.2.10	The methods and techniques used in the derivation and verification of the Safety Targets are appropriate to the Application Domain
1.2.11	The processes, tools and techniques that are used in the Hazard analysis, are appropriate to the Application Domain, and allow an adequate analysis of the THORs
1.2.12	The methods and techniques used in the derivation and verification of the THORs are appropriate to the Application Domain, and allow an adequate analysis of the *Specification*
1.2.13	All methods, processes, tools and techniques used in the development of Direct evidence for Arg 1.2 are appropriate to the specified safety integrity levels
1.2.14	All methods, processes, tools and techniques used in the development of Direct evidence for Arg 1.2 have been correctly applied

ID	What has to be Demonstrated
1.2.15	Only suitably qualified and experienced staff have applied the methods, processes, tools and techniques used in the development of Direct evidence for Arg 1.2

5.3.3 Supporting Evidence for Arg 1.2 from IEC 61508

IEC 61508-1, clause 7.4, clearly recognises the need for a Hazard and Risk Analysis to be caried out, and sets three "*objectives*" [sic] for it:

15. "… *to determine the hazards, hazardous events and hazardous situations relating to the EUC and the EUC Control System, in all modes of operation, and for all reasonably foreseeable circumstances* …";
16. "… *to determine the event sequences leading to the hazardous events* …"; and
17. "… *to determine the EUC risks associated with the hazardous events* …".

Being a generic, multi-sector safety standard, IEC 61508-1 cannot be expected to detail how a tolerable level of risk would be determined, despite the need for knowing this being crucial to its fundamental concept of Necessary Risk Reduction". Thus, while the first two "*objectives*" [sic] strongly imply the need to demonstrate the completeness and correctness of the Hazard and Risk Analysis results, the overall process involved lacks the context of what would be a tolerable level of risk for the EUC and a tolerable rate of occurrence of each of EUC hazard — thus there is there is little in the IEC 61508-1 text (clause 7.4) that explicitly requires the completeness and correctness of the Hazard and Risk Analysis results to be demonstrated, as required by the much more stringent Assurance Objectives of Sub-section 5.3.2 above.

The third objective seems to be related to subclause 7.5.2.4 of IEC 61508-1, which states that:

> *"...overall safety integrity requirements shall be specified in terms of either:*
> *– the risk reduction required to achieve the tolerable risk,* ***or***
> *– the tolerable hazardous event rate so as to meet the tolerable risk".*

Since the strategy for Arg 1.2, set out in Sub-section 5.3.1 above, is based on the latter of these options, objective #3 above is not required; indeed, subclause 7.4.2.10 of IEC 61508-1 helpfully recognises the need to "*... consider... the tolerable risk for each hazardous event*".

5.3.4 An ATM Perspective on Arg 1.2

Sub-section 3.3 of Fowler and Fota (2023) describes five accident types relevant to Approach airspace, which has been adapted from ICAO (2011)[25] and, in each case, involves death or serious injury to one or more of those on board. It then presents (from SESAR (2018b)) five related EUC Hazards[26] and a mapping back to the associated accident types.

Because Fowler and Fota (2023) present a *generic* view of Point Merge operations, rather than a specific case study, it does not quantify either the probabilistic relation between the

[25] ICAO (2011) is intended for use in *a posteriori* categorisation of actually occurrences, rather than *a priori* safety assessment—hence the need for some adaptation

[26] SESAR (2018b) uses the term "*pre-existing*" hazards for what IEC 61508-4 defines as EUC hazards.

Hazards and Accidents, or the Tolerable Hazard Occurrence Rate for each Hazard. Instead, it refers to the Accident Incident Models of SESAR (2018a), from which those parameters could be derived.

Nevertheless, Sub-section 3.3 of Fowler and Fota (2023), does provide a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.2 for a typical new ATM operational concept, at a specific existing airport.

5.3.5 A Rail Perspective on Arg 1.2

Sub-section 4.3 of Fowler and Graebner (2023) describes the various accident types that are relevant to railway operations, and, in each case, would probably involve death or serious injury to one or more of those on board a train, or to the workforce or members of the public on, or in the vicinity of, the track. It presents a comprehensive list of the EUC-related Hazards that are likely to fall within the scope of the System-to-be-specified, in relation to what are seen to be the most credible accident outcome(s). The accident and hazard sets were both derived from doctoral research carried out by Ivan Lucic, and adopted for general use by London Underground Ltd. (Lucic 2015).

Unfortunately, in sticking (perhaps too closely!) to the related IEC 61508-1 processes at this stage, the reference has little evidence to offer in relation to deriving Tolerable Hazard Occurrence Rates; rather, it defers the matter (incorrectly) to the subsequent discussion, on Overall Safety Functions. Otherwise, Sub-section 4.3 of Fowler and Graebner (2023), provides a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.2 for a typical new moving-block railway-signalling system.

5.4 Arg 1.3 Specification — Satisfaction of Requirements

5.4.1 Overview

Jackson (1995) explains that a *Specification* captures the properties that are *shared* by the Application Domain and the Machine, i.e. the System, and acts as a sort of logical interface between those two entities. In essence, it has to specify everything that the System has to *make happen* in the Application Domain in order to satisfy the Requirements thereof, with two provisos:

- it should contain *no* view or assumption regarding *how* the System might do that, in terms of its Design or Implementation, i.e. it should take a "black-box" view of the System; and
- it should be expressed in a way that the System *Design* could ensure that everything that the System has to make happen in the Application Domain, would actually happen.

A more challenging question that the author was faced with. in researching the subject of this article, was what form should a Specification take for transport applications, given that the safety Requirements for the Application Domain would most likely be expressed in the form of THORs for each and every identified EUC Hazard. At first sight, it might appear that a Specification is synonymous with the concept of Overall Safety Functions (OSFs) of IEC 61508-1. However, in applying the OSF concept to the simple example of a pedestrian-safety problem, Fowler (2022) quite reasonably deduced that the main hazard would be "*…moving traffic and pedestrians intending to occupy the same area of the road surface at the same time*", but ended up with a simple functional safety requirement that:

"In the area of potential conflict, the … safety function shall ensure the safe separation of pedestrians and moving road traffic temporally and/or spatially".

Whereas such a requirement would not preclude any particular mitigation strategy or ultimate technological solution and might be, at the highest level, sufficient in itself to mitigate the Hazard, it is not particularly enlightening since all it is really saying is don't let the Hazard happen!

In order to avoid such anodyne outcomes, a somewhat different approach is proposed, as follows.

Firstly, any options that IEC 61508-1 defines as "*Other Risk-reduction Methods*" (ORRMs) should be separated out *before* more-focused and more-detailed requirements for mitigation of the Hazards are derived. This would be a minor departure from IEC 61508-1 but the reasoning for this can be seen in, say, the case of a choice between a pedestrian crossing and a pedestrian underpass[27], for which it would be largely pointless to try to derive a *single* set of high-level safety requirements to cover both options.

Secondly, in order to avoid straying into the area of system Design, the requirements in a Specification should be *rule*-based, rather than functional, i.e. stating what has to *happen* in the Application Domain, *rather* than stating *how* it has to be done. In ATM, these rules would include the *separation minima* for aircraft in the airspace concerned; in the rail sector, the rules would include the *signalling principles* for the railway concerned.

Thirdly, in order to be really meaningful, the *rules* would need to be set in the associated operational context, otherwise it would not be valid to argue that the Specification would be sufficient to satisfy the Requirements "*under all normal, abnormal and failure conditions*" (see Sub-section 5.4.2). In ATM, the question as to where and when separation *rules* should be applied would depend on the operational concept (e.g. Point Merge, as outlined in Fowler and Fota (2023)) and, within that, would vary between the various phases of flight. In the rail sector, there would be significant variation in the *rules* between, say, *fixed-block* and *moving-block* operations and, within that, on different parts of the railway. Specific examples of the application of the above will be provided in Sub-sections 5.4.5 and 5.4.6.

Note: although it is beyond the scope of this article to enter into detailed discussion about the introduction of AI into safety-related applications, it would be important, in such cases, to consider whether the AI would be used as a means of satisfying the existing Specification rules more efficiently or, say, 'machine learning' would be used as a means of developing and applying new, and more-efficient, rules. In the latter case, the safety assurance problem would clearly be very much greater!

5.4.2 Strategy

The Strategy for satisfying Arg 1.3 is to show that the *Specification* correctly states what is necessary and sufficient to achieve tolerable safety, in the Application Domain. Specifically, it is necessary to provide arguments and evidence which show that:

18. The safety *Requirements* that are to be addressed by the *System* have been identified as such, and are *separate* from those requirements to be addressed by what IEC 61508-4 defines as ORRMs;

[27] See Fowler (2022).

19. the *Specification* for the System states correctly what is necessary and sufficient to satisfy those Application Domain safety Requirements that have been allocated to the System, under all *normal*, *abnormal*, and *failure* states of the EUC, the EUC Control System and the Operational Environment, as well as the *transitions* between those states; and
20. the *Specification* states correctly what is necessary and sufficient, and in sufficient detail and clarity, to enable the subsequent system *Design* and *Implementation*, in conjunction with any identified ORRMs, to achieve the required level of safety in the Application Domain.

5.4.3 Assurance Objectives for Arg 1.3

Credible arguments and evidence shall be provided to demonstrate the achievement of each of the 10 Assurance Objectives defined in the tables below.

Table 6 ~ *Direct* Assurance Objectives for Arg 1.3

ID	What has to be Demonstrated
1.3.1	The Application Domain safety *Requirements* have been completely and correctly allocated and apportioned to the System and any ORRMs
1.3.2	Any assumptions about ORRM contributions to the mitigation of Hazards are clear, quantified and justified
1.3.3	At least one *Overall Safety Function* has been developed for each Hazard, clearly and accurately reflecting the System's required contribution to the mitigation of Hazards, at the Specification level *Note: the IEC 61508-1 concept of an Overall Safety Function is used here merely as a convenient way of expressing a high-level function generically – its use is **not** limited to the Standard's meaning of the term*
1.3.4	The rule-based Overall Safety Requirements derived for each Overall Safety Function are sufficient to mitigate the corresponding Hazards in the Application Domain Requirements
1.3.5	The Overall Safety Requirements set out in the Specification adequately specify the required safety behaviour of the System at the Specification level, for all normal, abnormal and failure states of the EUC, the EUC Control System and the Operational Environment, and the transitions between those states
1.3.6	A target Safety Integrity requirement has been specified for each Overall Safety Function, such that the corresponding THORs in the Application Domain safety Requirements would be met *Note: In this context, Safety Integrity is defined as the "…probability of a … safety-related system satisfactorily **performing** the specified safety functions under all stated conditions, within a stated period of time" (IEC 61508-4)* *Thus, the form of these target Safety Integrity Requirements is entirely **different** from what IEC 61508-1 describes as "Safety Integrity Requirements"*

Table 7 ~ *Backing* Assurance Objectives for Arg 1.3

ID	What has to be Demonstrated
1.3.7	The analytic methods, techniques and tools used in the derivation and verification of the Overall Safety Requirements are appropriate for the attributes of those requirements
1.3.8	The analysis notations are appropriate to the Application Domain and the Safety Integrity requirement specified for each Overall Safety Function
1.3.9	The analysis notations allow an adequate analysis of the *Design*
1.3.10	Only suitably qualified and experienced staff have applied the analysis methods, techniques and tools

5.4.4 Supporting Evidence for Arg 1.3 from IEC 61508

As noted in Fowler (2020), the language of Section 7.5 of IEC 61508-1 can be quite difficult in places; nevertheless, the IEC 61508-1 concept of *Overall Safety Functions*, and (within them) *Overall Safety Requirements*, is consistent with the Jackson (1995) description of a *Specification*.

The importance of this IEC 61508-1 concept should not be underestimated; however, clause 7.5 of IEC 61508-1 does not offer nearly enough to satisfy the Assurance Objectives for Arg 1.3, listed in Sub-section 5.4.3 above, mainly because OSFs are defined at too high a level, i.e. at a level above which they are allocated to Safety Related Systems and/or Other Risk Reduction Measures. This means that the OSFs act as little more than a *basic-traceability* path between the A*pplication Domain Requirements* to mitigate Hazards (in Arg 1.2) and the SRSs that provide that mitigation (in Arg 2). The problem with that approach in relation to the transport sectors, is that the analysis needed to demonstrate *satisfaction* of the Requirements is made very much more complex and difficult if it is left entirely to the subsequent high-level *Design* phases of IEC 61508-1 clauses 7.6 and 7.7.

Without Jackson's work, it would be all too easy to underestimate the enormous potential value that clause 7.5 of IEC 61508-1 *could* bring to the development of safety-related systems in the transport sector. Fortunately, there is much in clause 7.7 of IEC 61508-1 (SRS high-level *Design*) that could be adapted and applied to the earlier phase, i.e. to the development and validation of Overall Safety Requirements, in clause 7.5 of the Standard, if the OSFs / Overall Safety Requirements were defined specifically for the System, as proposed above.

Hopefully, Sub-sections 5.4.5 and 5.4.6 below will give some insight into how this could be accomplished in the development of an illustrative *Specification*, for ATM and Rail applications, respectively — the overriding need here being to demonstrate, with the appropriate degree of confidence, that the *Specification* satisfies the Requirements, given the properties of the *Application Domain*.

5.4.5 An ATM Perspective on Arg 1.3

Further to the comments in Sub-section 5.4.4 above, it is now recognised that Sub-section 3.4 of Fowler and Fota (2023), in simply following clause 7.5 of IEC 61508's Overall Safety Function processes at this stage, falls well short of what would be needed to satisfy Arg1.3

for the Point Merge ATM application. Whereas it derives a comprehensive list of Overall Safety Functions, and Overall Safety Requirements, there is no attempt, in Fowler and Fota (2023), to validate those requirements in terms of their completeness, correctness, consistency or sufficiency, in relation to mitigation of the associated Hazards, under all *normal*, *abnormal,* and *failure* conditions!

That said, the work described later, in Sub-section 3.6 of Fowler and Fota (2023), presents a quite detailed, and much more comprehensive and rigorous analysis, which was originally aimed at adherence to the *Design* processes specified in IEC 61508-1 clause 7.7. It is now clear that this analysis could quite easily be adapted for use at the, earlier, *Specification* level, leading to a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.3 for a typical new ATM operational concept. For a simple operational change like Point Merge, it would also reduce greatly the amount of analysis that would then be needed at the high-level Design stage — see Sub-section 6.5, below[28].

5.4.6 A Rail Perspective on Arg 1.3

Further to the comments in Sub-sections 5.4.4 and 5.4.5 above, it is now recognised that Sub-section 4.4 of Fowler and Graebner (2023), in simply following clause 7.5 of IEC 61508's Overall Safety Function processes at this stage, also falls well short of what would be needed to satisfy Arg1.3, for the CBTC moving-block application. Whereas, it derives (from an independent, authoritative source, IEC 62290-1:2014) a very comprehensive list of Overall Safety Functions, and Overall Safety Requirements, there is no attempt, in Fowler and Graebner (2023), to validate them in terms of their completeness, correctness, consistency or sufficiency, in relation to mitigation of the associated Hazards, under all *normal*, *abnormal*, and f*ailure* conditions!

Fortunately, the work described in Sub-section 4.6 of Fowler and Graebner (2023) presents a quite detailed, and much more comprehensive and rigorous, analysis that was originally aimed at adherence to the Design processes specified in IEC 61508-1 clause 7.7. It is now clear that the various analysis techniques that are described in Sub-section 4.6 of Fowler and Graebner (2023) could (and should) be adapted for use *also* at the earlier, *Specification* level and, in so doing, would give a reasonable illustration of what would need to be done in order to provide adequate Evidence in support of Arg 1.3 for a typical new moving-block railway-signalling system; in so doing, it would also simplify somewhat the analysis that would then be needed at the high-level Design stage — see Sub-section 6.6, below.

6 Argument 2 — System Design

6.1 Overview

Argument 2 asserts that the high-level *Design* of the System satisfies its *Specification*, correctly and completely, under all *normal*, *abnormal,* and *failure* conditions. Whereas the *Specification* sets out the rules about what *must happen* in the *Application Domain*, the

[28] The author intends to present a revised version of the analysis in Sub-section 3.4 of Fowler and Fota (2023) in a 2027 issue of the Safety-critical Systems eJournal

Design specifies the Safety Functions that *must be performed* by the System, such that that the rules are satisfied, in the *Application Domain*, under all the stated conditions.

Furthermore, in order to provide an adequate basis for Argument 3, i.e. the argument that the *implementation* of the *Design*, in the built-and-tested System, satisfies the *Design* requirements correctly and completely, the *Design* would typically be based on a high-level *Architecture* of the System, specifying:

- one or more *Logical* (rather than physical) elements — what IEC 61508-4 defines as Safety Related Systems (SRSs) — and the interactions between them; and
- the individual *Safety Functions* to be performed by each SRS.

In the case of functionally complex applications, e.g. railway signalling systems, it would usually be necessary to complement the *Logical Architecture* with separate, *Functional Models* of the system, with cross-checking between these two views of the *Design*.

6.2 Strategy

The *Strategy* for satisfying Arg 2 is to show that the high-level *Design* correctly states what is necessary and sufficient to satisfy the System *Specification*, whilst providing a necessary and sufficient set of requirements for the achievement of a tolerable level of safety in the physical Implementation of that *Design*.

Specifically, it is necessary to provide arguments and evidence which show that:

21. the *Design* for the System states correctly what is necessary and sufficient to satisfy the Overall Safety Requirements specified in the System *Specification*, under all *normal*, *abnormal*, and *failure* states of the EUC, the EUC Control System and the Operational Environment, as well as the transitions between, and recovery from, those states (**Arg 2.1**); and
22. the *Design* states correctly what is necessary and sufficient, and in enough detail and clarity, to enable the subsequent implementation of the System *Design* in the *Implementation*, to achieve the required level of safety for the *Application Domain*, for all *normal*, *abnormal,* and *failure* conditions (**Arg 2.2**).

6.3 Assurance Objectives for Arg 2

Credible arguments and evidence shall be provided to demonstrate the achievement of each of the 17 Assurance Objectives defined for Arg 2.1, in Table 8 and Table 9; and the 11 Assurance Objectives defined for Arg 2.2, in Table 10 and Table 11.

Table 8 ~ *Direct* Assurance Objectives for Arg 2.1

ID	What has to be Demonstrated
2.1.1	A necessary and sufficient set of Safety-related Systems (SRSs) has been identified in response to the Overall Safety Functions and Overall Safety Requirements of the System *Specification*
2.1.2	The safety requirements set out in the System *Specification* have been completely and correctly allocated and apportioned to the set of SRSs, such that the Overall Safety Requirements for each Overall Safety Function are capable of being satisfied

ID	What has to be Demonstrated
2.1.3	At least one *Safety Function* has been developed for each SRS, clearly and accurately reflecting the System Design's required contribution to the rule-based requirements at the Specification level *Note: the terms Safety Function and SRS are as defined in IEC 61508-4 — see Sub-section 1.4, above.*
2.1.4	Each *Safety Function* has been correctly identified as being applicable to an SRS. operating in low demand, high demand or continuous modes of operation
2.1.5	The Functional Safety Requirements specified for each *Safety Function* are sufficient to satisfy the Overall Safety Requirements allocated to its host SRS
2.1.6	The Functional Safety Requirements specified for each *Safety Function* take full account of all relevant information related to the Application Domain, at this level of the Design
2.1.7	The Functional Safety Requirements for each *Safety Function* adequately specify the required safety behaviour of the System at the *Design* level, for all ***normal*** states of the EUC, the EUC Control System and the Operational Environment, and the transitions between, and recovery from, those states
2.1.8	The Functional Safety Requirements for each *Safety Function* adequately specify the required safety behaviour of the System at the *Design* level, for all ***abnormal*** states of the EUC, the EUC Control System and the Operational Environment, and the transitions between those states, including recovery of the System to *normal* operations
2.1.9	The Functional Safety Requirements for each *Safety Function* adequately specify the required safety behaviour of the System at the *Design* level, for all ***failure*** states of the EUC, the EUC Control System and the Operational Environment, and the transitions between those states, including recovery of the System to *normal* operations
2.1.10	The Functional Safety Requirements for each *Safety Function* adequately specify the required safety attributes of the *Safety Function*
2.1.11	A Safety Integrity Requirement (SIR) has been specified for each *Safety Function*, such that the corresponding target SIR(s) in the System Specification would be met
2.1.12	For SRSs that implement Safety Functions of different Safety-integrity Levels, it has been shown that either: • there would be sufficient independence of implementation between these particular Safety Functions; or • the requirements applicable to the highest relevant Safety Integrity Level apply to **all** Safety Functions

Table 9 ~ *Backing* Assurance Objectives for Arg 2.1

ID	What has to be Demonstrated
2.1.13	The analysis methods, techniques and tools used in the derivation and verification of the Functional Safety Requirements for each Safety Function are appropriate for the attributes of those requirements
2.1.14	The analysis notations are appropriate to the Application Domain and the Safety Integrity Levels derived for each Safety Function
2.1.15	The analysis notations allow an adequate analysis of the *Design*
2.1.16	The allocation of the Safety Integrity Requirements was carried out using appropriate techniques for the combination of probabilities
2.1.17	Only suitably qualified and experienced staff have applied the analysis methods, techniques and tools

Table 10 ~ *Direct* Assurance Objectives for Arg 2.2

ID	What has to be Demonstrated
2.2.1	The System high-level *Design* safety requirements are specified in sufficient detail and clarity to allow the *Implementation* of the System to achieve the required level of safety
2.2.2	The Functional Safety Requirements for each *Safety Functio*n are specified explicitly as such, and are set out in such a way as to be easily distinguishable from other (non-safety) requirements
2.2.3	The System high-level *Design* safety requirements specify the necessary limiting and constraint conditions for the Implementation of the System high-level *Design* due to the possibility of common cause failures
2.2.4	There are no emergent properties of the System high-level *Design* that could allow non-safety functions to interfere with Safety Functions and the Data sources / flows, in the *Implementation* of the System *Design*
2.2.5	All feasible measures have been taken to avoid that necessity of the requirement for a SIL 4 Safety Function in the *Design*
2.2.7	All Functional Safety Requirements and Safety Integrity Requirements for the high-level *Design* are realistic, i.e. are capable of being implemented by known, existing technology
2.2.8	All Functional Safety Requirements and Safety Integrity Requirements for the high-level *Design* are verifiable at the Implementation level

Table 11 ~ Backing Assurance Objectives for Arg 2.2

ID	What has to be Demonstrated
2.2.9	The analysis methods, techniques and tools used in assessment of the suitability of the Functional Safety Requirements for each Safety Function are appropriate for the attributes of those requirements

ID	What has to be Demonstrated
2.2.10	The analysis methods, techniques and tools used in assessment of the suitability of the Functional Safety Requirements are appropriate to the Application Domain and the Safety Integrity Levels derived for each Safety Function
2.2.11	Only suitably qualified and experienced staff have applied the analysis methods, techniques and tools used in assessment of the suitability of the Functional Safety Requirements.

6.4 Supporting Evidence for Arg 2 from IEC 61508

Subclause 7.10.1 of IEC 61508-1 states that the "*objective*" [sic] of Phase 9 — System Safety Requirements Specification — is to, "*define the ...* [SRS] *safety requirements, in terms of the* [SRS] *safety functions ... and the* [SRS] *safety integrity requirements, in order to achieve the required functional safety*".

Subclause 7.10.2.1 then adds that, "*The ...* [SRS] *safety requirements specification shall be derived from the allocation of safety requirements specified in* [Phase 5] *together with all relevant information related to the application. This information shall be made available to the ...* [SRS] *developer*".

Whereas the overall intent is clear, the traceability back to the Overall Safety Requirements (see Sub-section 5.4 above) is weak, and the stated objective of achieving "*the required functional safety*" is (for complex, transport applications, at least) wholly inadequate in terms of essential rich traceability back to the top-level Application Domain Requirements (see Sub-section 5.3 above). These are a further reflection on the weaknesses of Phase 5 of IEC 61508-1, as noted in Sub-section 5.4.4 above.

Otherwise, Phase 9 of IEC 61508-1, and subclause 7.10.2.6 in particular, places great emphasis on the need for a description of the workings of the SRS, including:

- a description of all the Safety Functions, how they work together to satisfy the rule-based requirements of the Overall Safety Functions, and whether they operate in low-demand, high-demand or continuous modes of operation;
- the required performance attributes of each safety function - e.g. timing properties and (for more data-intensive applications than possibly envisaged by IEC 61508) data accuracy, latency, refresh rate, and overload tolerance;
- all interfaces[29] that are necessary to achieve the required functional safety;
- all relevant modes of operation of the EUC and its Control System;
- response of the SRSs to abnormal conditions that might arise in the EUC, its Control System or its Operational Environment;
- all required modes of behaviour of the SRSs — in particular, its failure behaviour and the required response in the event of such failure; and
- the need for the safety requirements at this level to be sufficient to enable a complete and correct implementation of the high-level Design, in the physical system(s).

Indeed, much use has been made of, *inter alia*, clause 7.10 of IEC 61508-1 in framing the Assurance Objectives for Arg 2, in Sub-section 6.3 above.

[29] IEC 61508-1 also includes "operator" interfaces at this level. These have been excluded here on the ground that human operators can by definition form part of a physical SRS and should be left until the physical (or at least logical) design stage of the development lifecycle.

6.5 An ATM Perspective on Arg 2

In relation to an ATM operational concept known as Point Merge, Sub-section 3.5 of Fowler and Fota (2023) suggests how the *Functional Safety Requirements* and *Safety Integrity Requirements*, which were derived for the three *Overall Safety Functions* in Sub-section 3.4 thereto, could be allocated to a total of 13 *Safety-related Systems*, as per clause 7.6 of IEC 61508-1[30].

Sub-section 3.6 of Fowler and Fota (2023) then purports to "*develop safety requirements for those Safety-related Systems, in terms of their Functional Safety Requirements … and* [Safety Integrity Requirements]*, in order to achieve the required functional safety under all normal, abnormal and failure conditions*", as per clause 7.10 of IEC 61508-1.

Whereas these are both reasonable attempts to follow the related IEC 61508-1 safety processes, they also inherit two weaknesses from the Standard, revealed by the safety-argument approach, viz:

- lack of rigour in the Specification, and validation, of the *Overall Safety Functions*, as reported in Sub-sections 5.4.4 and 5.4.5 above;
- the lack of rich traceability, back to Application Domain Requirements, through the Overall Safety Requirements, as implied in the in use of the inadequate phrase "*in order to achieve the required functional safety*".

However, as already been explained, in Sub-section 5.4.5 above, most of the scenario analysis presented in Sub-section 3.6 of Fowler and Fota (2023) would much more usefully have been carried out at the *Specification* level in order to enhance the *Overall Safety Requirements* thereof. This would have required only small changes in the terminology used, e.g. Overall Safety Requirements instead of Functional Safety Requirements, and strict adherence to the rule-based principle of requirements expression in the Specification; there would also be no need to repeat the analysis at the *Safety-related System* level since, as noted in Sub-section 3.6.2 of Fowler and Fota (2023):

> *"In the particular case of Point Merge operations, there are no new SRSs / Safety Functions; rather, the operational concept is based on existing … elements of … what IEC 61508-4 terms the EUC Control System."*

In effect, this would mean that, for Point Merge, the main high-level Design outputs would be the design of the *Precision Area Navigation* (P-RNAV) routes, which make up the Approach patterns, and the human tasks that would need to be carried out (mainly by flight crew and air traffic controllers), and when they were to be carried out, in order to satisfy the rule-based Overall Safety Requirements, under the stated conditions.

Finally, Sub-section 3.6.6 of Fowler and Fota (2023) addresses the question of how failures *internal* to the ATM System impact on the safety of Point Merge operations. It notes that IEC 61508 suggests a Risk Classification Scheme (RCS) as a possible method for deriving Safety Integrity Requirements at this level but, having already cast doubt on the validity of RCSs used traditionally in ATM, it outlines a scheme based on that used on the SESAR Programme (SESAR 2018a), which resolves most, if not all, of those doubts. Although lacking in specific detail regarding failure-assessment processes and outputs, it does at least provide insights into the use of "*appropriate techniques for the combination of probabilities*" as required Assurance Objective #2.1.16, in Table 9 above.

[30] No *Other Risk-reduction Methods* were identified for the Point Merge application described.

6.6 A Rail Perspective on Arg 2

Sub-section 4.5 of Fowler and Graebner (2023) suggests how the eight *Overall Safety Functions*, derived in Sub-section 4.4 thereto, could be allocated to nine *Safety-related Systems* and a number of associated *Other Risk-reduction Methods*[31]. For the purposes of that exercise, the SRSs were not pure invention; rather, they had been adapted directly from the highest-level functional elements set out in the *Urban Guided Transport Management and Command/Control Systems* rail standard, IEC 62290-1.

Sub-section 4.6.3 of Fowler and Graebner (2023) then derives, and describes, a total of 30 illustrative Safety Functions covering the nine SRSs. Again, this task was made easier by adapting the Safety Functions, and their descriptions, from IEC 62290-1. As noted therein, this initial set of Safety Functions covered only those necessary for *normal* operational conditions. Following the same strategy, it was noted that some 350 Functional Safety Requirements would be needed to cover the 30 Safety Functions; Appendix B to Fowler and Graebner (2023) therefore presented, as examples, just the 41 requirements that would apply to the three SRSs needed to support the Overall Safety Function "*Establish & Protect a Safe Route for each Train Movement*".

Recognising that the above examples had been derived solely hierarchically, Sub-section 4.6.4 of Fowler and Graebner (2023) sought to show explicitly:

- how the functions interact with each other, and with the elements of the wider CBTC system and the EUC Control System;
- the information needed by, and produced by, the SRSs and Safety Functions;
- the system states and sequence of events, during a typical "day-in-the-life" of a train; and
- that the requirements constitute a complete, correct and coherent set.

It is noted therein that there is a wide range of techniques for addressing these issues and gives worked examples of the use of:

- State-transition Models of the various operational states of the railway;
- Operational Scenario Analysis;
- Sequence Diagrams for specific scenarios; and
- Activity Diagrams for capturing the interactions between Safety Functions and the Data involved.

Sub-section 4.6.5 of Fowler and Graebner (2023) outlines an equivalent approach for *abnormal* operational conditions, which stem mainly from:

- hazardous events in the operational environment that are not encountered on a day-to-day basis; and
- failure events within the EUC Control System but *outside* the scope of the CBTC system itself, e.g. a failure of a train's traction-control system.

It is noted therein that, in either case, the following need to be addressed:

- what effect the event would have on the continuing functioning of the CBTC system and the consequences for the safe operation of the railway; and

[31] For the three most important OSFs, and the five most important SRSs, the relationships involved in the allocations were one-to-many, making the subsequent derivation of requirements for Safety Functions somewhat more complex than would otherwise have been the case!

- what actions would need to be taken to mitigate the consequences of the event, and how the functionality of the CBTC system (existing or additional) could be used to support such actions.

Finally, Sub-section 4.6.6 of Fowler and Graebner (2023) addresses the question of how failures *internal* to the CBTC System impact on the safety of railway operations. It notes that, as such analysis is "*covered comprehensively in existing rail safety standards, including EN 50126-1*", Sub-section 4.6.6 is limited to addressing key principles relating specifically to the IEC 61508-1 viewpoint. However, although lacking in specific detail regarding failure-assessment processes and outputs, it does at least provide insights into the use of "*appropriate techniques for the combination of probabilities*" as required by Assurance Objective #2.1.16, in Table 9 above.

Overall, the aim of Fowler and Graebner (2023) was to show what the safety assessment of a railway signalling system would look like, if based on IEC 61508-1; given that proviso, Sub-sections 4.5 and 4.6 of that article do a reasonably good job of illustrating what the safety assurance model in Figure 1 refers to as Assurance Activities and Activity Outputs. What it lacks (as does IEC 61508-1 largely) is much detail on the subsequent processes involved in Output Analysis, which are so vital to the submission of Evidence to satisfy the Assurance Objectives and thence to support the related Safety Argument.

7 Conclusions

The 1970s through 1990s was a bad period for international system safety, across a variety of safety-critical industries. Individual reaction to serious accidents led gradually to a worldwide understanding of the need for the development of a holistic approach to safety management and what has become recognisable as typical safety-management systems. What also emerged, in 1990, was an extremely important finding of the Cullen inquiry into the 1998 Piper Alpha oil platform disaster, which "*recommend that operators be legally required to produce a comprehensive Safety Case as a means of assuring themselves that their operations are acceptably safe*" (Cullen 1990).

In parallel with these activities, and in the face of emerging new technologies, the IEC started work on a generic standard for the assessment of Programmable Electronic Systems (PES) used in safety-related applications. Over the period 1998 to 2000, following earlier releases of draft material, the first full editions of the seven parts of IEC 61508 were published. IEC 61508 was, and *still* is, widely accepted as the "gold standard" in Functional Safety, i.e. in the safety assessment of Safety Related Systems whose primary purpose is to provide mitigation of operating hazards that are inherent in many disparate safety-related industries sectors. Its stated purpose has always been to provide a solid, comprehensive basis for adaptation to a very wide range of applications — in that general sense, it has undoubtably been successful, though, at a more detailed level, the completeness and consistency of some of the adaptations of the Standard, released with regard to the original, might be called into question — EUROCONTROL's System Safety Assessment "Methodology" and the CENELEC railway RAMS standard EN 50126:1999 could be considered to be cases in point.

Fast-forwarding to today we see, in the European ATM and Rail sectors, respectively:

- the appearance of a EUROCONTROL produced[32], and recently updated, safety assessment manual, with the somewhat vague title "*Safety Reference Material*", with accompanying "*Guidance*" for applying it; and
- major updates to the CENELEC standards EN 50126:2017, EN 50716:2023 (formerly EN 50128) and EN 50129:2026.

The SESAR (2018a) "*Safety Reference Material*" makes no claim to be compliant with IEC 61508 but, to the firsthand knowledge of the author of this article, it does at least adhere to the fundamental principles of IEC 61508-1, outlined herein.

EN 50126:2017, on the other hand, is quite specific on this point and *implies* that the three CENELEC standards together do comply with IEC 61508 — a huge claim. Whereas the latest versions of the three CENELEC standards do seem to bear close resemblance to IEC 61508[33], they do include what might be viewed as a potential logical inconsistency in that, apparently for reasons of compliance with Regulation (EU) No 402/2013, "*Common Safety Method for Risk Evaluation and Assessment*", they use the variable concept of a "*System Under Consideration*" as the starting point for a safety assessment, rather than the very specific IEC 61508 concept of Equipment Under Control.

Over many decades, Professor Nancy Leveson of MIT has noticed that "*... system safety engineering has increasingly drifted toward using reliability engineering techniques and away from classic system safety approaches ...*" and this has resulted in the fact that accidents she studied had involved software that did exactly what it was designed to do, but for which the designers had misunderstood what *behaviours* were required from the software in order to achieve an acceptable level of safety — in other words, the software reliably performed the *wrong* functions. This exactly what can happen if, when a new system is introduced into an existing operation, that system (instead of the overall operation) becomes the main (or, worse still, sole) focus of the safety assessment process — what Jackson (1995) refers to as, and warns about, an imbalance in "*the competing demands and attractions of the world and the machine*" during the development process.

In any case, what is generally well understood is that all the aforementioned standards are primarily processed-based, and whilst specifying the *assurance activities* to be followed in system safety assessment, do not guarantee that the activity outputs would provide sufficient *evidence* for a *safety case* that showed that the *system under consideration* would enable an acceptable level of safety for the overall operation. For that reason, it was decided that a more productive approach for this article would be to develop and to present a rigorous and independent Safety Argument structure, against which the effectiveness of system safety assessment standards, such as the SESAR Safety Reference Material or the CENELEC series, (and including IEC 61508-1 itself!) could be judged individually, rather than comparing one set of standards with another.

Such a Safety Argument structure is presented herein. Although it is based on the fundamental principles underlying IEC 61508, and generally aligned with the IEC 61508-1 safety lifecycle, it is otherwise largely independent of the provisions of that Standard.

The key, and to some extent unique, features of the Safety Argument framework are as follows:

[32] With others, and on behalf of the European Commission's Single European Sky ATM Research & Development (SESAR) Programme.

[33] Certainly, a *much* closer resemblance that earlier versions see Fowler (2015).

Firstly, the Argument comprises five top-level claims, covering the whole of a typical system lifecycle, and is concerned entirely with *Functional Safety*, i.e. safety relating to an inherently hazardous "Equipment Under Control" ,which, for the purposes of all four articles in this series, has been interpreted as the flow of road, rail or air traffic, in the area of interest and which depends on the *correct functioning* of Safety-related Systems, and/or other means of reducing the associated inherent risks to an acceptable level.

Secondly, the first two claims, (Argument 1 — System Specification, and Argument 2 — System High-level Design) are based on a rigorous Requirements-engineering model, adapted from the seminal work of Dr Michael A Jackson (Jackson 1995) and underpinned by the principles of *Rich Traceability*. The adaptation of Jackson's model makes many vital contributions to Argument 1, in particular, by introducing the concept of a *Specification* that:

- captures the properties shared by the *System* and its *Application Domain* (the latter comprising the EUC, the EUC Control System and the Operational Environment);
- states only what the System has to *make happen* (as opposed what the System has to *do*) in the Application Domain; hence, for the transport sector at least, the Specification expresses its safety requirements as *Rules* to be applied in the Application Domain (i.e. rules of the road, sea, rail or air), *rather than* safety functions to be provided by a *safety-related system*; and
- has to be shown to be necessary and *sufficient in itself* to satisfy the safety Requirements, i.e. the S*afety Targets*, of the Application Domain.

It also forces Argument 1 to be conditional on complete and correct knowledge of the related properties (actual or assumed) of the Application Domain.

A review is included of three cases from the transport sector, in which seeming ignorance of the principles of Jackson (1995) led to catastrophic failures; this has underlined the importance of that work to the framing of a sound Safety Argument from the outset.

Thirdly, the Argument / Evidence structure is founded on a Safety Assurance Model in which the lowest level of decomposition of the Safety Argument, and the highest level in the Safety Assurance Model, are sets of Assurance *Objectives*, which prescribe exactly what has to be *demonstrated* by the Evidence, i.e. by the outputs of the Assurance Activities. These Assurance Objectives are based quite heavily on a system-level interpretation of a very comprehensive set of *Regulatory Objectives for Software Safety Assurance in ATS Equipment*, published by the UK Civil Aviation Authority (UK CAA 2019).

An assessment of the provisions of IEC 61508-1, within the scope of the three predecessor articles, then showed that:

- there is a significant deficiency in Phase 4 of IEC 61508-1 (Overall Safety Requirements), in relation to its *Specification* equivalent; whereas the Standard does mandate the derivation of technology-independent Overall Safety Requirements for each Overall Safety Function, the treatment is somewhat superficial, and there are no explicit requirements to validate the completeness, correctness or sufficiency of the Overall Safety Requirements in relation to the satisfaction of the Application Domain Requirements;
- Phase 5 of IEC 61508-1 (High-level Design) rightly makes demands regarding the rigour and completeness of the Design information; however, it lacks explicit requirements for demonstrating, *via the Specification*, that the high-level Design satisfies the top-level, safety requirements of the Application Domain; this would not be an adequate situation for many safety-related systems in the transport sector, due to their much greater

complexity at the Design level compared with the process industries wherein lie the origins of IEC 61508.

Otherwise, as shown previously in the assessment of the example applications for the ATM sector in Fowler and Fota (2023) and for the Rail sector in Fowler and Graebner (2023), IEC 61508-1 makes a considerable contribution to the definition of Safety Assurance *Activities* but, being a generic, process-based standard, would not satisfy the majority of the Assurance Objectives derived herein.

It is concluded, therefore, that as far as IEC 61508-1 is concerned, the Safety Argument would not be satisfied purely by adherence to the Standard, although the Standard does provide most of what would be required in the way of specifying the Safety Assurance Activities needed to produce sufficient Evidence to support the Safety Argument.

Given that, the question of whether or not "*... [EN 50126], together with the requirements of other suitable standards, is sufficient to ensure that additional compliance to IEC 61508 does not need to be demonstrated*" seems to be not only difficult to justify but also to be largely irrelevant. Therefore, far more important questions would be to what degree, and extent, would compliance with those railway safety standards satisfy the Safety Argument presented herein — and, if not, why not?

Disclaimers

Any opinions expressed herein are those of the author, not necessarily those of any of his past or present employers or clients.

Whilst every effort has been made to provide a comprehensive set of generic Assurance Objectives, in the context of the article, they should be understood as being illustrative and non-exhaustive, rather than as being definitive for any particular application.

Correspondence Address

Corresponding e-mail address: derek-fowler@btconnect.com.

Acknowledgments

The author wishes to acknowledge the help, support, understanding of IEC 61508, and experience of aviation and rail safety, provided by his long-standing colleague Ron Pierce, without which this series of articles would not have come to fruition.

The copyright holder of the quotations from published standards used for illustration in this article is the International Electrotechnical Commission, Geneva. The primary copyright holder for EN 50126, and associated standards, is CENELEC (the European Committee for Electrotechnical Standardization), Brussels. The SESAR 3 Joint Undertaking (SESAR JU) holds the direct copyright ownership of SESAR material; it mandates that while technical project results are technically owned by the participating industry beneficiaries, they are licensed back to the SESAR JU.

References

BFU. (2004). Bundestelle fur Flügunfallundersuchung (German Federal Bureau of Aircraft Accident Investigation) Investigation Report AX001-1-2/02, May 2004. Available from: https://skybrary.aero/sites/default/files/bookshelf/414.pdf. Accessed 11th September 2026.

CLC/TR 50451. (2007). *Railway applications – Systematic allocation of safety integrity requirements.* CENELEC (European Committee for Electrotechnical Standardization), Brussels.

CLC/TR 50506. (2007). *Railway Applications – Communication, signalling and processing systems – application guide for EN 50129.* CENELEC (European Committee for Electrotechnical Standardization), Brussels.

Cogent Skills. (2026). *The Bhopal tragedy and its impact on process safety*. Cogent Skills. https://cogentskills.com/news/the-bhopal-tragedy-and-its-impact-on-process-safety. Accessed 17th July 2026.

Cullen W. D. (1990). *The Public Inquiry into the Piper Alpha Disaster*. HMSO Publications Centre, ISBN 0–10–113102-X.

Dick J. (2002). *Rich Traceability*. ResearchGate, January 2002. Available from: https://www.researchgate.net/publication/239560722_Rich_Traceability. Accessed 17th July 2026.

Dowson M. (1997). *The ARIANE 5 Software Failure*. In ACM SIGSOFT Software Engineering Notes Volume 22, No. 2, March 1997. Available from: https://dl.acm.org/doi/pdf/10.1145/251880.251992. Accessed 17th July 2026.

(EU) No 402/2013. (2013). *Commission Implementing Regulation (EU) No 402/2013 of 30 April 2013 on the common safety method for risk evaluation and assessment and repealing Regulation (EC) No 352/2009*. European Commission, Brussels.

EN 50126-1. (2017). *Railway applications – the specification and demonstration of reliability, availability, maintainability and safety (RAMS), Part 1: Basic requirements and generic process.* EN 50126-1:2017. CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EN 50128. (2020). *Railway applications – Communication, signalling and processing systems – Software for railway control and protection systems.* EN 50128:2020. CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EN 50129. (2003). *Railway applications - Communication, signalling and processing systems - Safety related electronic systems for signalling.* EN 50129:2003, CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EN 50716. (2023). *Railway Applications — Requirements for software development*. EN 50716, 1st Edition, 2023. CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EUROCONTROL. (2006a). *Air Navigation System Safety Assessment Methodology*. EUROCONTROL SAF.ET1.ST03.1000-MAN-01, 3 October 2006. Available at https://skybrary.aero/sam-toolkit. Accessed 13th July 2026.

EUROCONTROL. (2006b). *Safety Case Development Manual*. EUROCONTROL DAP/SSH/091, Edition 2.1, 13 October 2006. Available at https://skybrary.aero/sites/default/files/bookshelf/33520.pdf. Accessed 17th July 2026.

EUROCONTROL. (2011). *ACAS II Post-implementation Safety Case*. EUROCONTROL ALDA Reference 11/03/28-15, Version 2.3, 25 November 2011. Available at https://skybrary.aero/sites/default/files/bookshelf/4967.pdf. Accessed 17th July 2026.

EUROCONTROL. (2022). *ACAS II Bulletin – Near collision over Yaizu*. Available at https://skybrary.aero/sites/default/files/bookshelf/32673.pdf. Accessed 17th July 2026.

Fowler D. (2015). *Functional Safety by Design – Magic or Logic?* In Proceedings of the 23rd Safety-Critical Systems Symposium, Bristol, UK.

Fowler D. (2022). *IEC 61508 Viewpoint on System Safety in the Transport Sector: Part 1 — An Overview of IEC 61508*. Safety-Critical Systems eJournal, Vol. 1, Iss. 2. Available at https://scsc.uk/r176.3:1. Accessed 10th September 2026.

Fowler D, and Fota O. N. (2023). *Safety Assessment of Point Merge Operations in Terminal Airspace — an IEC 61508 Viewpoint*. Safety Critical Club eJournal, Vol. 2, Iss. 1. Available at https://scsc.uk/r183.3:1#page=25. Accessed 10th September 2026.

Fowler D, and Graebner A. (2023). *An IEC 61508 Viewpoint on the Safety Assessment of Railway Control Systems*. Safety Critical Club eJournal, Vol. 2, Iss. 2. Available at https://scsc.uk/r1858. Accessed 10th September 2026.

Fowler D, Perrin E, and Pierce R. (2009). *2020 Foresight — a Systems-engineering Approach to Assessing the Safety of the SESAR Operational Concept*. Proceedings of the 8th USA/Europe Air Traffic Management Research and Development Seminar, Napa, USA

Fowler D, and Pierce R. (2012). *A Safety Engineering Perspective*. In: Cogan B. (editor) *Systems Engineering: Practice and Theory*. IntechOpen. London.

Fowler D, and Tiemeyer B. (2002). *Reduced Vertical Separation in European Airspace – A Case Study in Effective Safety Management*. Proceedings of the 20th International System Safety Conference, Denver, USA August 2002.

Hammond J, Rawlings R, and Hall A. (2001). *Will it Work?* In the Proceedings of the 5th IEEE International Symposium on Requirements Engineering, Toronto, Canada, 2001, ISBN 0-7695-1125-2

ICAO. (2006). *Aircraft Operations Volume I, Flight Procedures*. International Civil Aviation Organisation Doc 8168 OPS/611, 5th Edition, 2006.

ICAO. (2007). *Procedures for Air Navigation Services*. International Civil Aviation Organisation Doc 4444, 15th Edition, 2007.

ICO. (2022). *Explaining decisions made with AI – Annexe 5: Argument-based assurance cases.* 17th October 2022. UK Information Commissioner's Office. Available at https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/artificial-intelligence/explaining-decisions-made-with-artificial-intelligence/annexe-5-argument-based-assurance-cases/. Accessed 2nd June 2026.

IEC 61508. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems*. IEC 61508, in 7 parts[34], 2nd Edition, 2010. International Electrotechnical Commission, Geneva.

IEC 62290-1. (2014). *Railway applications – Urban Guided Transport Management and Command/Control Systems – Part 1: System Principles and Fundamental Concepts*. IEC 62290-1:2014, 2nd Edition, 2014. International Electrotechnical Commission, Geneva.

IRJ. (2014). *London Underground and Bombardier abandon Tube signalling contract*. International Railway Journal, 3 January 2014.

[34]Parts are typically denoted with hyphen and part number, e.g. IEC 61508-3:2010 is Part 3, dealing with software development requirements.

https://www.railjournal.com/financial/london-underground-and-bombardier-abandon-tube-signalling-contract/. Accessed 17th July 2026.

IRJ. (2015). *Thales wins £760m London Underground re-signalling contract.* International Railway Journal, 3 August 2015. https://www.railway-technology.com/news/newsthales-wins-760m-london-underground-re-signalling-contract-4638341/?utm_source=chatgpt.com&cf-view. Accessed 27 October 2025.

Jackson M. A. (1995). *The World and the Machine*. In the Proceedings of the 17th IEEE International Conference on Software Engineering (ICSE), Seattle, WA, 1995, ISBN 0-89791-708-1

Leveson N. G. (2001). T*he Role of Software in Recent Aerospace Accidents.* In the Proceedings of the 19th International System Safety Conference, Huntsville, Alabama, USA, 10-14 September 2001. Available at http://sunnyday.mit.edu/accidents/issc01.pdf. Accessed 17th July 2026.

Leveson N. G. (2020). *Are You Sure Your Software Will Not Kill Anyone?* Communications of the Association of Computing Machinery (ACM), 1st February 2020. Available at: https://cacm.acm.org/opinion/are-you-sure-your-software-will-not-kill-anyone/. Accessed 12th July 2026.

Lions J-L. (1996). *Ariane 5 — Flight 501 Failure*. Report by the Inquiry Board, Paris, France 19 July 1996. Available at: https://esamultimedia.esa.int/docs/esa-x-1819eng.pdf. Accessed 17th July 2026.

London Assembly. (2016). *Transport for London's Signal Failure*. The London Assembly's Budget and Performance Committee Report dated March 2016. Available at: https://www.london.gov.uk/sites/default/files/transport_for_londons_signal_failure.pdf. Accessed 17th July 2026.

Lucic I. (2015). *Risk and Safety in Engineering Processes*. Cambridge Scholars Publishing, January 2015. ISBN (13): 978-1-4438-7077-1.

Mannan M. S, West H. H, Krishna K, Aldeeb A. A, Keren N, Saraf S. R, Liu Y, and Gentile M. (2005). *The legacy of Bhopal: The impact over the last 20 years and future direction*. Journal of Loss Prevention in the Process Industries. Volume 18, Issues 4–6, 2005, Pages 218-224. Available at: https://doi.org/10.1016/j.jlp.2005.06.037. Accessed 24th June 2026.

ORR. (2024). *Common Safety Method for Risk Evaluation and Assessment*. UK Office of Rail and Road, 16th December 2024.

Redmill F. (1999). *An Introduction to the Safety Standard IEC 61508*. Journal of the System Safety Society, Volume 35, No. 1, First Quarter 1999. The International System Safety Society, Britton, SD.

SAE ARP4754A. (1996). *Recommended Practice, Guidelines for Development of Civil Aircraft and Systems*. ARP4754A. SAE International, Warrendale, PA. Available at https://doi.org/10.4271/ARP4754A. Accessed 13th July 2026.

SAE ARP4761. (1996). *Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment*. ARP 4761. SAE International, Warrendale, PA. Available at https://doi.org/10.4271/ARP4761. Accessed 13th July 2026.

SESAR. (2018a). *SESAR Safety Reference Material*. Edition 00.04.01, 14th December 2018. SESAR Joint Undertaking. Available at https://www.sesarju.eu/sites/default/files/documents/transversal/SESAR2020%20Safety

%20Reference%20Material%20Ed%2000_04_01_1%20(1_0).pdf. Accessed 17th July 2026.

SESAR. (2018b). *Guidance to Apply SESAR Safety Reference Material*. Edition 00.03.01, 14th December 2018. SESAR Joint Undertaking. https://www.sesarju.eu/sites/default/files/documents/transversal/SESAR%202020%20-%20Guidance%20to%20Apply%20the%20SESAR2020%20Safety%20Reference%20Material.pdf. Accessed 17th July 2026.

Thomas S. M, and Fowler D. (2013). *Safety Case for the Airborne Collision Avoidance System*. In: Dale C, and Anderson T. (editors) *Assuring the Safety of Systems*, Proceedings of the Twenty-first Safety-Critical Systems Symposium, Bristol, UK, 5-7th February 2013.

UK CAA. (2019). *Regulatory Objectives for Software Safety Assurance in ATS Equipment*. CAP 670 Part B, Section 3, SW01. UK Civil Aviation Authority, Third Issue, Amendment 1/2019, 1 June 2019.

UKMoD. (2016). *Journey to ALARP*. UK Ministry of Defence. June 2016. Available at: https://test.asems.mod.uk/sites/default/files/documents/Journey%20to%20ALARP.pdf. Accessed 17th July 2026.

Willey R. J, Crowl D. A, and Lepkowski W. (2005). *The Bhopal tragedy: its influence on process and community safety as practiced in the United States*. Journal of Loss Prevention in the Process Industries, Volume 18, Issues 4–6, 2005, Pages 365-374. Available at: https://doi.org/10.1016/j.jlp.2005.06.019. Accessed 4th July 2026.

Appendix A. Good Practice in Safety Arguments & Evidence

The following principles (adapted from EUROCONTROL (2006)) are recommended to be followed in the development of an argument and evidence structures.

A.1 Decomposition of Safety Arguments

The decomposition of a top-level Claim into lower-level Arguments provides the essential links between the top-level Claim and the wealth of Evidence needed to show that the overall Claim is valid. In performing this decomposition, it is essential that:

- each Claim in the Argument structure is expressed as a simple predicate, i.e. a statement that can be only true or false;
- the Argument structure does not contain any indirect or inconclusive Claims;
- the means by which a Claim is (or is intended to be) shown to be true is not included in the Claim statement; for example, "*System testing shows that system requirements have been satisfied*" pre-supposes (incorrectly for most high-integrity systems!) that testing would be an adequate means of showing that the system requirements have been satisfied;
- the Arguments at every level of decomposition are necessary and sufficient to show that the parent Claim or Argument is true;
- a valid counterargument, which would negate the parent Claim or Argument, does not exist; for example, a fatal counterclaim to the statement (two bullets) above would be simply "*Testing is an inadequate means of showing that the system requirements have been satisfied*";
- where the rationale for decomposition of a Claim or Argument into lower-level Arguments is not self-evident, it is explained by supporting text, i.e. a Strategy;
- the number of levels of decomposition is appropriate to the complexity of the Safety Case and/or supporting Evidence;
- each branch of the Safety Argument structure is terminated in supporting Evidence;
- a Claim or Argument may be satisfied by either a further level of decomposition or by supporting Evidence, but not both at the same level; and
- there is a clear distinction between, and correct use of, Direct (product-based) and Backing (process-based) Arguments; see section A.3 below.

A.2 "Rules of Evidence"

The corresponding essential rules of Evidence are as follows:

- Evidence must be presented only to the degree and extent necessary to support the related Claim;

  Note: *the issue here is that, in the context of an Argument-based approach, any "Evidence" which is unrelated to a part of that Argument is not only of no value but could also serve as a distraction from those aspects of the Safety Case that are relevant. On the*

other hand, any Evidence which actually undermines the validity of an Argument must not be ignored. The existence of such Evidence must be acknowledged and explained fully in the Safety Case.

- multiple sources of Evidence under a single Claim should not be used as a substitute for further decomposition of the Argument, except where one piece of Evidence is Direct and the other is Backing;
- Evidence must be clear, conclusive and, wherever possible, objective;

  Note: *It is vital to the integrity of the Safety Case that the Evidence be presented in such a way that is clear to the reader that the Evidence does actually show the related Argument to be true, "beyond all reasonable doubt".*

  It is bad practice to present an element of a structured Argument and then refer to a mass of information as "Evidence" to substantiate the Argument.

  Where Evidence is contained in appendices or external documents, a summary justifying the adequacy of the Evidence should be presented (in the Safety Case) along with the associated Argument. It is not sufficient to merely reference the Evidence with statements such as "Evidence to support the Argument is presented in …".

- the type of Evidence — from safety analysis, design, simulation, test, previous usage, standards compliance, etc. — must be appropriate to the Argument and required Safety Integrity (or Assurance) Level;
- "expert" opinion must not be used as Direct Evidence, unless its use is clearly justified in the Safety Case (see, however, below for the relevance of "competence" in Backing assurance);
- the rigour of the Evidence must be appropriate to the required Safety Integrity (or Assurance) Level;
- Evidence must actually relate to the correct configuration of the system under consideration; and
- there must be a clear distinction between, and correct use of, Direct (product-based) and Backing (process-based) Evidence; see section A.3 below.

A.3 *Direct* versus *Backing* Assurance

In constructing a Safety Argument, the following two main types of argument (and supporting evidence) should be presented:

- that which shows that a particular objective has been achieved, i.e. a higher-level Argument or Claim is satisfied; this is referred to as *Direct* assurance;
- that which shows that the Direct assurance is trustworthy, i.e. that it can be relied upon; this is referred to as *Backing* assurance.

Direct assurance may be thought of as that assurance which relies directly on the observable properties of a product, i.e. the output of a process, to show that the product satisfies the requirements placed upon it.

Backing assurance, on the other hand, is obtained from the properties of the processes by which Direct assurance was obtained, and shows that those processes, tools and techniques,

human resources etc were appropriate, adequate and properly deployed. In general, Backing assurance should include:

- the suitability of the processes, tools and techniques etc that are used to generate the evidence obtained from the safety assessment;
- the correct application of those processes, tools and techniques; and
- the competence of the personnel applying those processes, tools and techniques.

The general rules for the use of Direct and Backing assurance are as follows:

- Direct assurance should always be supported by Backing assurance, as necessary to demonstrate the *trustworthiness* of the Direct assurance, as appropriate to the required Safety Integrity Level; and
- Backing assurance should *not* be used other than to demonstrate the *trustworthiness* of the Direct assurance.

ISSN 2754-1118 (Online) — ISSN 2753-6599 (Print)

Perils of Precedent in Nuclear Software Qualification

Fred Pollard, Martin Rendell, and Daniel Finnigan

Rolls-Royce SMR Limited

Abstract

This paper examines the continuing use of the United Kingdom nuclear industry's current precedent in the qualification of software and programmable logic used in nuclear control and instrumentation systems. In this context, software qualification has conventionally relied on a two-legged argument comprising "production excellence" and "independent confidence-building measures" (the latter including independent static analysis, independent dynamic analysis, and statistical testing). Drawing on precedent from Sizewell B and subsequent UK nuclear practice, this paper argues that these measures have too often become in-effect "prescribed" techniques rather than measures selected to answer a clearly stated safety question. This paper reviews the claimed contribution of these techniques to safety and confidence-building, considering their practical and technical limitations, their dependence on developer interpretation, their cost and schedule implications, and their effect on through-life modification. It concludes that post hoc re-verification of completed software can provide little additional safety benefit where modern standards-based development, verification, validation, and independent functional safety assessment have already been performed, and may inhibit future safety improvements to a system by making modification disproportionately difficult. The paper proposes a possible alternative approach to software qualification, but notes that until the goal of qualification is determined, it is not clear how to assess the validity of any qualification performed above and beyond that which is prescribed by international standards.

1 Introduction

1.1 Background

The UK nuclear industry has developed a relatively unique approach to qualification of software in control & instrumentation (C&I) systems since the 1990s, with the original catalyst for the establishment of the approach being the Primary Protection System (PPS) at Sizewell B Nuclear Power Station (SZB) in Suffolk, UK. This has established a UK-nuclear–specific precedent that is frequently used as a basis for software justification within this industry.

This software-qualification approach was inspired by the "special-case procedure" for plant structural components that have very small random-failure rates — systematic defects dominate random failures (May et al. 1995). This procedure established a "multi-legged" safety case for such components. It was considered a helpful analogy for software, which

© Rolls-Royce SMR 2026.

cannot fail randomly — only systematically[35]; thus, a special-case procedure was established for software-based systems as well.

This software-qualification approach, comprising a "two-legged" qualification framework, focuses on demonstrating software quality through:

- a confirmation that the software-based system's development lifecycle and design meet expectations based on relevant international standards (the "production excellence" leg); and
- independent confidence-building activities, such as analysis and test techniques and measures, usually performed on the final version of the software (or integrated system) after it has completed its development lifecycle (the "independent confidence-building measures" (ICBMs) leg) and preferably in a diverse manner to the development lifecycle's equivalent activities[36].

It is the ICBM leg in particular that is often identified as contributing significantly to the cost and schedule of software-based C&I systems in the UK, especially for systems performing Category A nuclear safety functions[37]. The ICBM leg, and the activities contained within it, is the primary focus of this paper's examination of the value of UK-nuclear software-qualification precedent. This is because unlike the production excellence leg, which is informed by useful precedent and international consensus in an evolutionary way through the development of standards and bodies of recognised good practice across safety industries, the ICBM leg is a construct unique to the UK nuclear industry, the basis for which has not significantly changed in nearly four decades (Hunns and Wainwright 1992) (ONR 2020).

The primary references for the two-legged software-qualification approach is guidance found in the Office for Nuclear Regulation's (ONR) Safety Assessment Principles (SAPs) (ONR 2020) and the ONR's technical assessment guide for computer-based systems (ONR 2023).

1.2 Purpose

This paper takes a step back from the software-qualification approach conventionalised within the UK nuclear industry since the 1990s to examine what the authors believe that the current approach achieves in reality, looking beyond the theory laid out in existing literature.

We focus in particular on five techniques which appear in regulatory documentation as additional activities to be performed over and above the activities already performed in a modern, safety-oriented development process:

1. Independent tool review
2. Certification by an independent body

[35] Given the same inputs, same external environment (including the hardware with which it is interacting) and same internal state, software will always generate the same output. Non-determinism in a software system's output, i.e. different output for same input, is due to incomplete definition of the system's behaviour, leading to incomplete software requirements.

[36] ICBM activities are expected by UK regulatory guidance (ONR 2020) (ONR 2023) to be performed specifically on the final post-validation product, rather than on pre-final product versions or on the development process itself. Activities on the development process are the remit of "production excellence" activities, and the expectation is that any pre-validation testing is within scope of the product developer's own processes.

[37] Nuclear safety function categories and system classes are defined in BS EN IEC 61226:2021. Category A functions play a principal role in preventing anticipated operational occurrences or design basis accidents from leading to high-severity consequences, or whose failure could directly lead to an unmitigated accident with high-severity consequences. The class of a system is informed by the categories of function(s) allocated to it and the system's contribution to them.

3. Independent static analysis
4. Independent dynamic analysis
5. Statistical testing

We review these techniques, in terms of their method and theoretical value, and consider how their application as ICBMs contributes to the plant safety case. We consider their interaction with the development lifecycle of the systems themselves, and the plant, and consider how human factors may affect the value to the safety case of these techniques.

1.3 Structure of this Paper

This paper comprises two core sections:

- Section 2 examines the use of the ICBM techniques listed above that appear in UK nuclear precedent, and the benefits that they offer. It also considers the challenges to safety that may arise by focusing effort on post-development verification.
- Section 3 explores the pre-requisites for constructing a useful qualification if it is not to be driven solely by the precedent from previous UK nuclear projects.

Note that in this paper, "software" is used as a catch-all term that includes hardware description language (HDL)-programmed devices such as field-programmable gate arrays (FPGAs), as such devices are considered to have similar relevant properties to conventional software as far as qualification is concerned. This terminology aligns with the UK nuclear consensus, as well as some other safety-critical industries worldwide.

2 Examination of Techniques Used in UK-Nuclear Precedent

2.1 Overview

The background in Section 1 has described the structure of prior demonstrations for nuclear-classified software, but this section will consider the activities that have previously been undertaken as part of these demonstrations. The "production excellence" part of prior demonstrations is not objectionable in principle, in that it addresses the need for the application of internationally-recognised standards and showing that these have been complied with. This form of demonstration appears in all nuclear jurisdictions of which the authors are aware, and so we do not discuss this aspect of the safety demonstration further.

The independent confidence-building leg has previously involved the execution of a series of activities, typically graded by classification. Table 2 in NS-TAST-GD-046, the ONR Technical Assessment Guide for computer-based safety systems (ONR 2023), which is entitled "*Example graded approach to the selection of measures to support justification of commercial off the shelf devices*", only applies to smart devices, and it is not formally a requirement that licensees follow this in any case. In practice, however, this table appears in one form or another with the same list of techniques in several contexts for so-called Computer-Based Systems Important to Safety (CBSIS) in existing facilities. In civil nuclear power plants, the representation of that table typically includes additional gradation with

two separate columns for Class 1 differing only by reliability claim so that more onerous techniques are reserved for a 10^{-4} pfd/pfa claim[38].

The ICBM techniques for a Class 3 claim according to this table are generally achieved by a good development process and the application of regular commercial practice. The differentiator in gradation is the extent to which the following techniques are applied:

1. Independent tool review.
2. Certification by an independent body (supported by evidence).
3. Static analysis of the source code.
4. Dynamic analysis of the source code.
5. Statistical testing.

The application of these techniques as independent activities (i.e. as part of the ICBM leg, separate from and subsequent to the development process) is not mandated by international nuclear standards such as BS EN 61513:2013 (or standards within its family such as BS EN 60880, or BS EN IEC 62138), nor indeed by any other functional safety standard that the authors have examined.

The activities represent precedent, and so it is helpful to consider whether these techniques should be included in a given nuclear software qualification programme. However, the basis for inclusion cannot simply be that it is what has been done in the past, since precedent reflects the technological context in which it was developed. Instead, precedent should be evaluated for any benefit it offers relative to the time, effort and cost involved in undertaking the additional activities in the present context.

For a system delivering nuclear safety functions, benefit can come in two forms: establishing a *safety benefit* and *building confidence*. These are related concepts, but they are not the same.

Safety benefit implies eliminating or reducing hazards, strengthening controls to eliminate risk or reducing the likelihood of error. In essence, safety is improved by an activity when it truly reduces the level of risk.

Building confidence is an exercise in increasing the trust that a system is safe or otherwise fit for purpose. Here, the activity is about changing the understanding of stakeholders, be they a customer, regulator, or even an assurance function within a development organisation. The aim in a safety context is to assure these stakeholders that safety has been achieved by demonstrating that the system will operate correctly.

By developing a system well, applying the safety lifecycle rigorously and introducing safety improvements throughout, strong technical evidence is naturally generated. These activities directly contribute to the safety of the system and any improvements in them should correspondingly produce benefits for safety.

Confidence, however, is established when the system is assessed as a whole, bringing that evidence together into a coherent, system level analysis that explores whether the safety functionality of the plant has been correctly implemented. This holistic review demonstrates that the lifecycle has been applied consistently, that the development activities have genuinely reduced risk, and that the integrated system behaves safely. In this way, good engineering creates safety benefits, and structured assurance creates confidence.

[38] Where pfd = probability of failure on demand, and pfa = probability of failure per annum.

In the context of international standards, the safety benefits are achieved by rigorous specification, design, implementation, verification and validation, coupled with hazard and failure analysis at each stage that establishes potential issues that can be resolved as part of the development lifecycle. These same standards will typically include an independent functional safety assessment (or equivalent) that fulfils the need for confidence-building by performing (BS EN 61508-4:2010):

> [an] *investigation, based on evidence, to judge the functional safety achieved by one or more ... safety-related systems and/or other risk reduction measures.*

The independence here is crucial to provide the confidence, with greater independence typically required as the safety significance of the functionality increases. However, note that this independent assessment from international standards does not re-execute development activities, including verification[39]. Instead, it probes the evidence already generated to form a judgement on the overall safety achieved.

Within international functional safety standards then, we already have the required elements of building safety into the plant and confidence-building for stakeholders. These activities are already effectively required in the UK nuclear regime as part of the "production excellence leg". Since the ICBMs are an additional layer of verification over and above the development activities and an independent assessment of those activities as described above, the remaining question is whether they provide meaningful additional benefits in the form of safety improvement or confidence-building.

2.2 Independent Tool Review and Certification

We can briefly state that tool review is already addressed by international standards, which require various forms of tool qualification. This is already incorporated into both the developer's activities (on the assumption that we have determined that the developer is following modern safety system development practices), and the independent assessment thereof. We therefore assert that performing another independent assessment of tool qualification is redundant, and so adds no safety benefit nor meaningful improvement in confidence.

Table 2 in NS-TAST-GD-046 also mentions certification by an independent body. If applied correctly, certification by an independent body may result in benefit in the form of safety improvements because certification authorities increasingly engage with developers during the process to provide independent advice, thus enhancing the development lifecycle quality. However, in the UK nuclear regime certification alone is not accepted and licensees are expected to review all evidence to confirm themselves that the development of components is adequate and in line with expectations. Therefore, in the UK nuclear context, certification at a component level might at most provide some initial guarantee that the additional independent audits and assessments are likely to be satisfactory, but the certification does not in of itself offer any usable evidence of quality. Consequently, it is judged that, in practice, certification by an independent body would be equivalent to an independent functional safety assessment for matters of significance[40]. This is already part of demonstrating "production excellence", so an exercise in certification would simply demand

[39] Note that in the UK nuclear industry, it is increasingly the case that software-based systems are wholly or partly pre-developed (e.g. smart devices and pre-developed platforms/programmable logic controllers, respectively). Consequently, the UK nuclear qualification approach, with its focus on *post hoc* independent re-verification and re-validation, can have little to no influence on the quality of the underlying software.

[40] Noting that different assessors may place different emphasis on different aspects of the system development.

a second independent assessment, which would be redundant, and so add no safety benefit nor meaningful improvement in confidence.

This leaves static analysis of the source code, dynamic analysis of the source code and statistical testing to consider in the coming subsections. However, first it is instructive to consider the structure of a typical ICBM activity and its impact on safety.

2.3 Structure of an ICBM Assessment Activity

For source-code ICBMs, there are three principal parties involved:

1. The *analyst* is the person or team responsible for performing the independent verification.
2. The *licensee* is the operating organisation responsible for the safety of the plant.
3. The *developer* can be any of the people, team(s) or organisation(s) who have specified, designed, implemented, verified and validated the safety system that contains the software.

The typical relationships and reporting arrangements between these parties are illustrated in Figure 7.

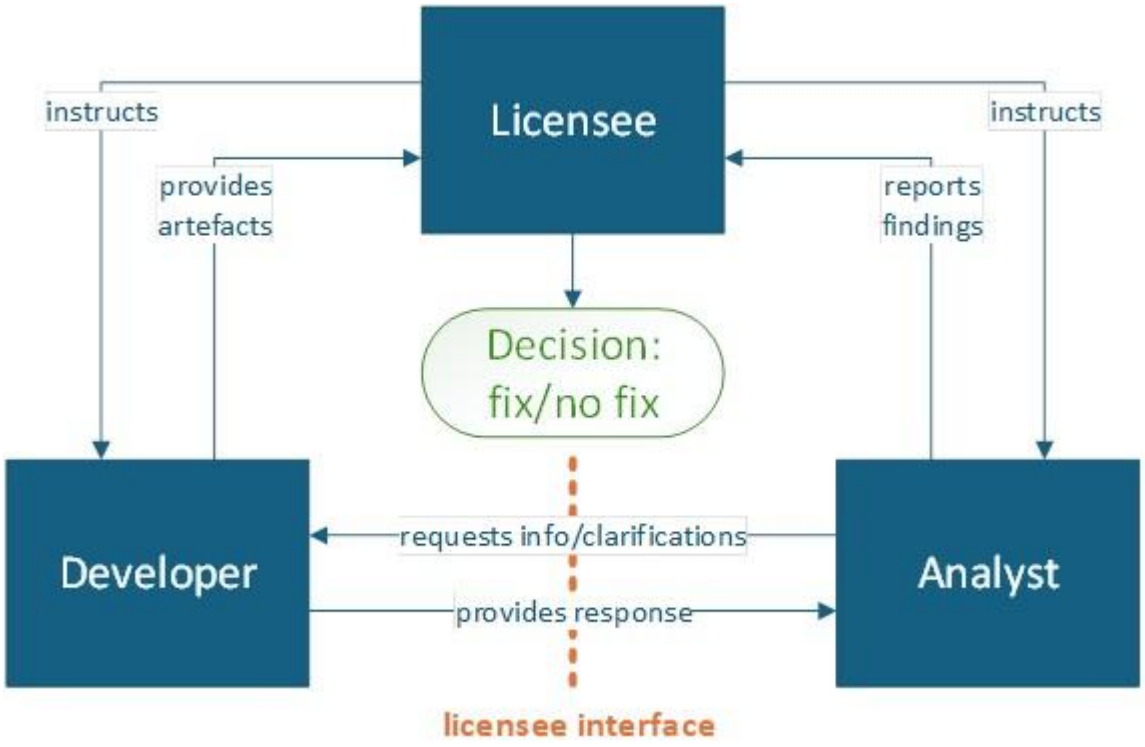


Figure 7 ~ Typical Relationships During ICBM Activities

The analyst typically reports to the licensee who is planning to operate the plant. The developer reports to the licensee as a supplier. Interactions between the analyst and the developer are in practice variable in their formality but good practice is that they should be documented and mediated by the licensee. In the course of their work, the analyst will make findings, which are technical outcomes that the analyst believes are anomalies or defects in the system. These findings are then sentenced in an engagement between the analyst and

developer, chaired by the licensee who makes the final determination on the impact of the finding to safety[41].

This arrangement is unavoidable, because the analyst must be paid, and the licensee is ultimately responsible for safety and so must oversee and ultimately accept all such findings. Efforts have previously been made to have dual roles within a single licensee organisation, but this is less apparent in recent practice, which instead emphasises the independence of the analysts themselves from the developers on the assumed premise that this ensures the result is not affected by bias or mindset.

It is worth highlighting that this arrangement is nonetheless extremely challenging to a good safety culture. The timing of ICBMs, which must by their nature be delayed until the source code has undergone sufficient verification and validation, and which cannot be completed until after system validation, tends to place them on the critical path of a project. Since any defects requiring source code changes would result in a costly and time-consuming modification cycle that would also risk injecting new faults, the licensee is disincentivised to agree to any categorisation of a finding that will result in changes because of cost and schedule pressures. This is not to suggest in any way that any licensee would make a conscious decision to ignore a safety-related finding, but in a case where the effect (perhaps in aggregation) of many findings is unclear, e.g. non-trivial/unidentified interactions, or subject to debate, there is always going to be a pressure that favours a less impactful categorisation.

Having established the reporting structures that are required for such a third-party independent re-verification, the next sub-sections consider the different techniques that have been applied in this way in prior precedent within the UK nuclear industry and examines their benefits to both safety and confidence-building.

2.4 Static Analysis

The term “static analysis” is extremely broad and can refer to any method of analysing software (including HDL) that does not involve executing it. In the past at Sizewell B, and more recently at Hinkley Point C, this has involved formal verification of software using the MALPAS Compliance Analyser for Class 1 systems with a claim of 10^{-4} pfd/pfa, and informal verification via symbolic execution using the MALPAS Semantic Analyser for Class 1 systems with a claim of 10^{-3} pfd/pfa. The static analysis family of techniques also incorporates techniques such as source-code comparison, which has previously been applied to systems with claims of 10^{-4} pfd/pfa (for example at Sizewell B), and attempts to provide confidence that the compiled program is a faithful representation of the underlying source code.

Note that this sub-section will describe these techniques in the context of the MALPAS tool, this is because it is the tool that has been used in precedent. However, the evaluations of each technique do not imply anything is amiss with the tool and would be the same for any tool used in a similar way.

At Sizewell B, formal verification (compliance analysis) was used to demonstrate that the software source code complied with both its software requirements specification and its software design specification (Ward, 1993). The analyst manually converted the details in

[41] This is typically a four-category system denoting an impact to nuclear safety (Category 1), documentation error (Category 2), non-functional or non-critical defect in the code such as comments (Category 3), and no anomaly (Category 4); for example see (Ward 1993).

these specification documents, which are written in a natural language specification, into a formal mathematical notation. The source code for each function was then modelled using an intermediate language that could, albeit with considerable additional human input, automatically demonstrate that the modelled source code matched its modelled specification[42].

This appears to be a rigorous process, though it suffers from only performing an analysis on individual components, rather than on the system as a whole. Minimal confidence can be developed on the safety of the overall system based on an in-depth analysis of one part of it. For example, such an approach cannot consider interface errors between system components, such as those caused by timing or communications faults[43]. Recognition that components of a system can individually function as intended, but nonetheless in combination still result in hazardous behaviours at the system level, is a key insight in modern safety engineering, and has motivated more recent approaches such as System-Theoretic Process Analysis (STPA).

Given that it offers little-to-no insight into system-level behaviours, the apparent objective of individual-component analysis is to establish confidence that the development process achieved its intended aims for that component, additional to that derived from the "production excellence" process itself. It can therefore be concluded that there is no direct safety benefit from the application of this technique, since there is only an intent to retrospectively demonstrate the achievement of a safety-related property, not to influence the development of the system itself.

Furthermore, behavioural formal verification, i.e. formal verification of the software functionality, used in this way can only generate confidence insofar as it concerns implementation against the software requirement and software specifications; no validation is performed of the correctness of the requirements and specifications, nor the correctness of earlier development lifecycle stages. However, it is not clear that confidence-building in even this narrow aspect is meaningfully achieved by behavioural formal verification of the source code either for reasons we shall now examine.

As noted, to perform behavioural formal verification on software not developed with formal methods, the analyst must convert a natural language specification into a mathematical formalism. However, natural language specifications are ambiguous, and do not explicitly outline developer assumptions that underpin them. Translation is dependent on analyst skill and judgement, rather than the ideal of unambiguous formal equivalence (Ward, 1993). There are therefore always going to be multiple, perhaps many potential translations from the natural language to formal definition.

It is therefore likely that either the analyst will require clarifications, or more likely that the formal verification itself will produce a finding based on a mismatch of the specification to the source code. In both cases, the only source of information is the developer of the system.

Clarifications will allow developer assumptions to be codified in the independent-verification mathematics, and their response to the findings in sentencing will be to highlight errors in the assumptions that the analyst has made in modelling the intent of the original specifications. The analyst will then take the reasonable course of action to adjust the

[42] The Sizewell B MALPAS analysis took 4.5 years and had a nominal team size of 15 people, rising to more than 80 in the later stages (Ward, 1993).

[43] Bruns and Anderson (1995) cite Littlewood and Strigini (1993), noting that "*faults which resist discovery by testing are typically the result of unforeseen interaction between components*". A similar argument can be applied to component-wise formal verification.

mathematical formalism based on this new information, perhaps highlighting that this suggests that the documentation should be made less ambiguous or to contain additional information.

At this point, consider what has occurred: the analyst's re-interpretation exercise is no longer independent of the developer in substance. Any mistakes that the developer has made in interpreting their specifications, or any erroneous assumptions made during the development now have a high likelihood of being encoded in the formal model. Indeed, the developer is always in a privileged position in these circumstances because they built the system and so know it better than anyone else. This gives an unavoidable authority bias during interactions, which the licensee is in no better a position to counteract. Since the only basis for the analysis is the natural language specification, the interpretation of the developer with respect to how the source code has been implemented (i.e. the aggregation of the programmer, reviewer, tester, etc., all of whom have converged on the same interpretation of the specification by the end of system validation) will therefore generally dominate.

The result is that this type of behavioural formal verification is likely to be based on a mathematical model that has converged on the developer's intent. The verification activity will therefore tend to confirm that the software behaves in the way the developers believe it is meant to behave[44], rather than what the software specification necessarily originally intended. This would tend to lead the analysis to focus on corrective measures to the documentation to aid analyst clarity, rather than to the software itself.

This is borne out empirically from the analysis at Sizewell B, which identified (as of 55% completion) approximately 2,000 findings (Ward 1993), of which:

1. No Category 1 errors — that is, no errors that impacted on safety.
2. 40% of the errors were Category 2, relating to the need to clarify the documented specifications.
3. 8% of the errors were Category 3 errors which proposed improvements to code defensiveness but were not critical defects.
4. 52% were classified as Category 4 (not requiring any action).

At the completion of the analysis campaign, there were still zero Category 1 findings, and only 6% of the total findings were Category 3, with the rest either Category 2 or 4 (Wichmann et al. 1995). The findings were therefore overwhelmingly dominated by categorisations requiring no change to the software, with the majority of findings at worst requiring clarification of the specification so that it reflected the implementation. The authors' experience performing and overseeing the similar work elsewhere is that the proportions of finding types are similar and for similar reasons.

Based on these figures, one of the following must be true: that the software was free from critical defects (in which case this suggests that the "production excellence" and its associated non-formal test and analysis activities were actually sufficient to assure safety and the formal verification offered limited additional insight into how the system would behave); that the *post hoc* analysis was incapable of finding significant defects in the software; or the *post hoc* analysis process is dominated by the authority a competent developer has over their code and their understanding of it.

[44] This may be particularly error-prone if the original software developers are no longer available, and so the personnel supporting the analysis in their stead are themselves having to infer the intended functionality. There is a risk here of the substitutes using what is written in the source code as the source of truth, or a deferral to the "authority" of the original developers.

It is effectively impossible to know which scenario above is most valid. In any case, there does not appear to be a meaningful safety benefit arising from the activity, with the risk that there is a false confidence generated from the presumption that the analysis is somehow meaningfully independent of the development.

This result appears unavoidable regardless of what stages of the lifecycle this type of after-the-fact formal verification is applied to[45]. If the analyst attempts to start modelling at the system level, they will still require extensive developer input to make the transition to the lower levels because for any system there is more than one way of designing and implementing a given set of requirements, and the analyst would need to codify all the assumptions the developer has made in doing so. Behavioural formal verification is therefore not a beneficial activity for safety, nor for a meaningful improvement in confidence when performed after the fact[46].

Semantic analysis suffers from the same issues as behavioural formal verification, but to a greater degree because the specifications are never formalised. Instead, the method converts sequential logic in software source code to a mathematical representation that the analyst then manually compares to the natural language specification. This again places reliance on the interpretation of the specification, where the developer will have a greater authority over the intent of the natural language specification.

There can be limited confidence that this is a genuinely independent approach offering significant improvement in confidence. It is worth noting that the proportions of findings presented earlier in this section incorporated the outputs from both compliance analysis and semantic analysis, so the conclusions are similar. We therefore conclude that it is not a beneficial activity for safety nor for a meaningful improvement in confidence.

A counterpoint to these conclusions could be that any additional verification activity that has the potential to identify defects is definitionally going to improve confidence, *particularly* if it finds no errors. To this, there are two replies: the first is that a modern, safety-standards compliant approach already performs a lot of verification, and this verification will almost certainly find errors in the initial implementation that are then resolved and verification repeated until the final exercise finds no defects of significance. Performing additional verification on already-verified code will suffer from diminishing returns[47]. This leads to the second point, which is that there has to be a point at which the time, effort, and cost of additional verification tasks has to be weighed against the safety benefit achieved, in line with the principle of reducing risk so far as is reasonably practicable (SFAIRP).

The time and costs associated with formal verification (compliance analysis) and semantic analysis can be estimated using the productivity ranges provided in the *MALPAS Advanced Software Analysis and Verification — Introductory Guide* (Atkins 2008) and some estimates of analyst cost and the number of lines of code (LOC) in a modern reactor protection system. For our estimates, we assume:

[45] i.e. whether *post hoc* analysis is performed on the artefacts of the completed system requirements, or on the artefacts of the completed coding phase, for example. The actual timing of the *post hoc* analysis is largely irrelevant.

[46] There is acceptance that formal methods used within the development process offer many benefits, but this is not widespread practice, nor mandated by standards at present.

[47] Software reliability growth modelling usually becomes less convincing at high reliability claims for this reason — it becomes increasingly difficult to demonstrate that the observance of no test failures shows high reliability due to the low expectation of failures at that reliability level.

1. A system with 1 million LOC. The Sizewell B PPS had approximately 100,000 LOC, and our experience is that an order-of-magnitude increase in the intervening 30 or so years is a reasonable estimate[48].
2. For compliance analysis, a productivity of 30 LOC/day. This is the midpoint of the range given by (Atkins 2008) of 10 to 50 LOC/day, which reasonably accounts for initially low but gradually improving productivity over the course of an analysis.
3. For semantic analysis, a productivity of 60 LOC/day. This is the midpoint of the range given by (Atkins 2008) of 20 to 100 LOC/day.
4. An analyst cost averaging £800 per day, based on our experience of market rates between £600 and £1000 per day.
5. 260 8-hour days per person-year, which is typical in planning to account for holidays, weekends, etc.
6. A 25% cost uplift to account for management of the work, verification activities, assurance activities, etc.

For compliance analysis, this produces a per-system estimate of ~33,300 person-days, or 128 person-years for a cost of ~£33.3m. For semantic analysis, this produces a per-system estimate of ~16,600 person-days, or 64 person-years for a cost of ~£16.7m. Note that these figures are still optimistic, because it also does not include the cost of developer time to support technical queries and sentencing meetings, which is typically charged to the licensee in addition to the development and supply costs for the C&I system. It also does not include the costs associated with the feasibility studies, tool developments or training associated with these activities.

In this example of a nuclear reactor protection system, these costs generally exceed the cost of developing and manufacturing the system itself[49]. Given the lack of safety benefit and the lack of meaningful improvement in confidence, we conclude that these activities are disproportionate to the associated time, efforts, and costs.

The presence of *post hoc* formal and semantic analyses in UK nuclear precedent is based on the technical landscape at the time of Sizewell B. Standards such as IEC 880:1986 (later to become BS EN 60880) did exist and were used as the basis for development of the Sizewell B PPS, but the functional safety standards relied upon in modern developments did not emerge until later[50].

However, the technical community was (and to some extent still is) convinced of the need for formal methods to assure high-integrity systems, to the extent that the original publication around that time of INT.DEF.STAN.00-55 (the Ministry of Defence software standard) *mandated* the use of formal methods for defence software projects (Watts 1989). In this environment, the *post hoc* behavioural formal verification of the Sizewell B PPS makes some sense since it was regarded as good practice to have a formally verified system.

Even this, however, would be to misrepresent the technical opinion at the time which was that the PPS should have been *developed* using formal methods. It is unclear that the

[48] This change in scale itself causes us to question whether earlier precedent is applicable or appropriate.

[49] In the case of smart devices, which typically cost on the order of hundreds or thousands of pounds, ICBM programmes alone typically cost several multiples more than this.

[50] There is evidence of national standards preceding the first edition of IEC 61508, which was published in 1998. For example, IEC 61508 was heavily influenced by the 1989 German standard DIN V 19250. However, this was a national standard and largely used only in Germany and Central Europe and there is no evidence of its direct influence on the development and licensing activities for Sizewell B. In any event, it is widely accepted that IEC 61508 was the first complete functional safety standard, and this came several years after Sizewell B entered service.

approach of applying them after-the-fact was ever satisfactory to practitioners of the time. The modern technical landscape is also different, and formal methods find use for integrity-checking, but are still not commonly applied for formal behavioural verification when developing software, particularly for nuclear systems. Having mentioned it in the previous paragraph, it is notable that after initial publication, subsequent issues of DEF.STAN.00-055 successively weakened and then removed the mandate for formal methods. This reflects the fact that the expectation, prevalent in the late 1980s and early 1990s, that behavioural formal verification would become a standard engineering practice has not yet materialised.

Another type of static analysis is integrity-checking, which looks for vulnerabilities in the code that may reduce the systematic reliability of the system. The arguments against behavioural formal verification or semantic analysis do not apply to integrity-checking, where software is examined for compliance to a set of general properties or rules (e.g. checking for accesses to arrays out of bounds). This is because there is no specification to interpret and the analysis is therefore more objective and independent of the developer. For example, a result that mathematically demonstrates the possibility of array access out of bounds cannot be defended based on a misinterpreted specification.

There is also widespread tooling available for this activity at this level of formalism, suggesting inherent feasibility. Such tooling, such as Polyspace Code Prover, does not require an analyst to specify formal mathematical properties, which are instead built directly into the tools. These tools are also generally able, to some extent, to consider the entire code base, rather than individual functions, when checking integrity properties, which can reduce the number of false positives requiring sentencing. As a result, the skills required for the work are likely to be more widely available and the attendant time, costs and effort are likely to be proportionate to the benefit obtained.

However, this also means that they are in widespread use *within* safety system development lifecycles, because modern standards require or recommend them and because they afford a relatively cheap means of reducing risk of remedial work later in the development process, or after the system has entered service. If the work has already been done by the developer, and already independently assessed for adequacy (including whether the tools were configured correctly for the task) then it seems questionable whether benefits would arise from a further analysis of a similar nature. We will return to this point later in the paper.

The other form of static analysis that appears in UK nuclear precedent for systems with claims of 10^{-4} pfd are activities to confirm that the source-code compilation tools have not negatively impacted on the design intent expressed in the software source code. A technique called source-code comparison (Pavey and Winsborrow 1993) was devised and executed by Nuclear Electric (later EDF Energy) for the Sizewell B PPS. EDF Energy has adapted this method and is attempting to employ it at Hinkley Point C (AREVA NP and EDF 2012). In a sense, this is another form of integrity-checking, albeit a specialised one.

This method involved decompiling the object code, processing both it and the original source code and constructing MALPAS models that would allow a demonstration of semantic equivalence. This technique is proprietary and owned by EDF Energy and, crucially, must be reconstructed for each compiler and each processor type (which may have different instruction sets); compiler/processor-specific modelling is required in the equivalence-checking tool (Pavey and Winsborrow 1993). The technique at Sizewell B was achievable because of the limited optimisations performed by the compiler, and the project coding standard constraints that avoided language constructs most likely to cause errors in the compiled code.

This allowed a clear source-object correspondence to be established at Sizewell B, which is key to making the technique tractable. Modern systems typically make use of optimisations, including whole-program optimisations, to meet performance requirements and other constraints (such as software size) which are likely to make automation of a complete formal demonstration of equivalence impossible, with consequent impacts on tractability. These complexities are being explored in the academic field of translation validation, but there is limited evidence of any industrial use and is largely an area of active research.

Of other industries examined, none has adopted such a formal approach to demonstrate equivalence between source and compiled image. In aviation, for DAL A software assurance under RTCA/DO-178C, there is a requirement to identify whether any additional code has been inserted by the compiler that is not directly traceable to the source code and to subject such code to additional verification (see Clause 6.4.4.2b of RTCA/DO-178C). However, this is linked to dynamic analysis activities, so the correspondence is established through a combination of test to establish correspondence from source code to executable object code and further test and/or analysis to examine any additional code that has been inserted and to determine its impact on the output. Extensive industrial tooling exists for this type of activity, but its formalism does not extend to that of source-code comparison.

The challenge of relying on precedent here, unlike for other static analysis methods explored in this subsection, is not so much the value of the technique if it can be performed[51]. The challenge is that it cannot be achieved for many modern systems to the same level of rigour as in preceding systems. A more extreme, but increasingly pertinent, example of this would be a system developed with FPGAs. FPGAs are developed using HDL, which is usually converted by a series of tools into a representation that reflects the configuration of the physical elements within an FPGA called the place-and-route netlist. This is then converted into a proprietary format called a bitstream using the FPGA vendor's tool, and this bitstream is what is used to configure the circuit.

However, the proprietary nature of the bitstream means that a technique like source-code comparison cannot be applied unless one has privileged access to the FPGA chip vendor's underlying intellectual property, which is not generally forthcoming[52]. This leaves an apparent gap in any justification because we are unable to "do what we have done before", and it has caused significant difficulties in deploying this useful technology within the UK nuclear industry, likely resulting in a detriment to safety improvements. The error here is not that the conversion process should necessarily be ignored, but that by focusing on prescribed techniques as the principal means of demonstration, we can lose sight of what questions we are trying to answer to satisfy ourselves that the system is safe. The focus on following UK nuclear precedent also diverts qualification away from newer approaches such as those described above for avionics software, or the use of formally-verified compilers such as CompCert (2026).

Indeed, for all the static analysis techniques, this is an issue that continues to arise: why are these techniques being executed? What questions are they trying to answer? This topic will be addressed in Section 3.

[51] Note however that exercising this technique still costs on the order of millions of pounds once analysis tool development is taken into account.

[52] One tool, PV-Bit by Graf Research, claims to be based on such privileged access but the nature of the commercial arrangements involved makes it challenging for a licensee to qualify the tool for this use since the outputs cannot be systematically verified against the inputs.

2.5 Independent Dynamic Analysis

It is unclear when independent dynamic analysis became established as an ICBM (for example in Table 2 of NS-TAST-GD-046 (ONR 2023)), and there appears to be limited precedent for its actual execution in this form. Other techniques have their origins in the licensing of Sizewell B, but independently performed dynamic analysis was not among the five activities that comprised the independent assessment (Ward 1993). For the Protection System at Hinkley Point C, for example, the relevant part of the ICBM programme consists of an independent review of the dynamic analysis undertaken by the developer, rather than an independent re-execution of the work (AREVA NP and EDF 2012). The appearance of independent dynamic analysis among the techniques associated with UK nuclear precedent therefore does not seem to arise from its repeated application to systems on previous projects. This makes it particularly important to ask what safety concern the activity is intended to address before treating it as an expected part of a qualification programme. In the authors' experience, however, independent dynamic analysis is applied to off-the-shelf COTS products, such as smart devices as part of their ICBM programmes.

Dynamic analysis is the technique usually employed by the developers of safety-critical or high-integrity systems, because it is required in relevant standards. It consists of executing requirements-based tests across units of code, or aggregations of such units, with the aim of demonstrating that the software complies with its design requirements. Separately, standards typically also require that some structural coverage metric is assessed to ensure that there is no code that is not traced to requirements and that other features such as conditions within the software are completely exercised.

As with static analysis, there is then an open question of what a completely independent execution of dynamic analysis is trying to demonstrate. Such an activity would involve one of:

1. Re-executing the analysis on a different tool;
2. Re-executing the analysis using different people;
3. Targeting different requirements; or
4. Some combination of the above.

If it is true that using a diverse set of tools and people could identify more issues, then this suggests that there is something wrong with the tools used by the development team (e.g., an inappropriate tool or incorrect configuration of it), or that there are process issues in its execution. We have already seen that modern safety standards require tool qualification (which should identify capabilities and limitations of the tool); and independent assessment should confirm the competence of personnel and the correctness of the tool qualification. If configuration and use of the tool is a specific concern, then it is more efficient to check that the configuration/usage was correct than to completely re-execute a verification programme from scratch. Independence is also of limited benefit, since a compliant development process would already require appropriate independence of the development and verification teams. Independent execution also relies heavily on developer interpretation of the requirements when findings are made, so this suffers similarly to the authority bias effects described for post-development behavioural formal verification.

In addition, modern test environments (and test harnesses) can be large, requiring significant implementation effort by the analyst, and cost. This can be tackled by independent re-implementation by the analyst or through re-use of developer artefacts (harnesses, etc.), the latter of which may diminish independence claims.

If the intent is to target different requirements, then it is unclear in general what these should be. Precedent for smart devices in the UK nuclear industry suggests that dynamic analysis may be targeted at robustness properties, or other areas of particular interest. However, even if these are not already addressed by standards that promote fault injection, stress testing and related techniques, it may be that simply asking the developer to perform more tests is a more straightforward route to achieving this. The key here is the independence and diversity of thought involved in devising the tests, not the way they are executed.

There is therefore no obvious benefit to an independent repetition of dynamic analysis in terms of improvements in safety or in increasing confidence, particularly if in addition to an adequate independent assessment of the original analysis. Since the costs will be of a similar order-of-magnitude to the developer's work themselves, as well as the need to develop a test environment, generate independent understanding of the code structure and licensing tools, it is apparent that this would be a substantial undertaking for the negligible value in outcome.

Independent dynamic analysis, like static analysis, focuses on software implementation. This focus gives rise to a number of other objections and detriments to safety, which are discussed in the next sub-section.

2.6 Perils of *Post Hoc* Source Code Analysis

Post-development source code analyses of developed software do not address the key risks associated with software, which lie in the specification and design of the overall system and how this is translated into software requirements, design and integration, rather than the implementation of the software itself.

Figure 8 shows the target of the source code-level ICBMs from precedent in context with the overall lifecycle of a C&I system. As stated previously, this highlights that only the implemented software is examined, but none of the preceding steps that dictate what that implementation is intended to achieve.

Any assurance derived from these activities, by focusing only on a demonstration that an implementation conforms to its specified behaviour that has already been made by the developer through other means, therefore provides limited benefit in understanding risk or demonstrating that it has been managed. This is because these activities provide limited insight into the behaviour of the system when the underpinning assumptions (such as system specification correctness and completeness, or interface guarantees) may be violated.

Incorporating post-development source code-level verification activities can also inhibit the longer-term achievement of safety. In the life of a C&I system, modifications to the software may be desirable, for example, to implement safety improvements to the system based on operational experience; to allow for wider plant modifications that have a safety benefit; or to patch the software to address identified security vulnerabilities, or changes in the threat landscape. However, requiring specific additional activities can introduce a bar to modification so high that such safety improvements will not be made because of the associated costs of re-executing analysis and the difficulty and costs associated with maintaining a pool of independent experts able to undertake them.

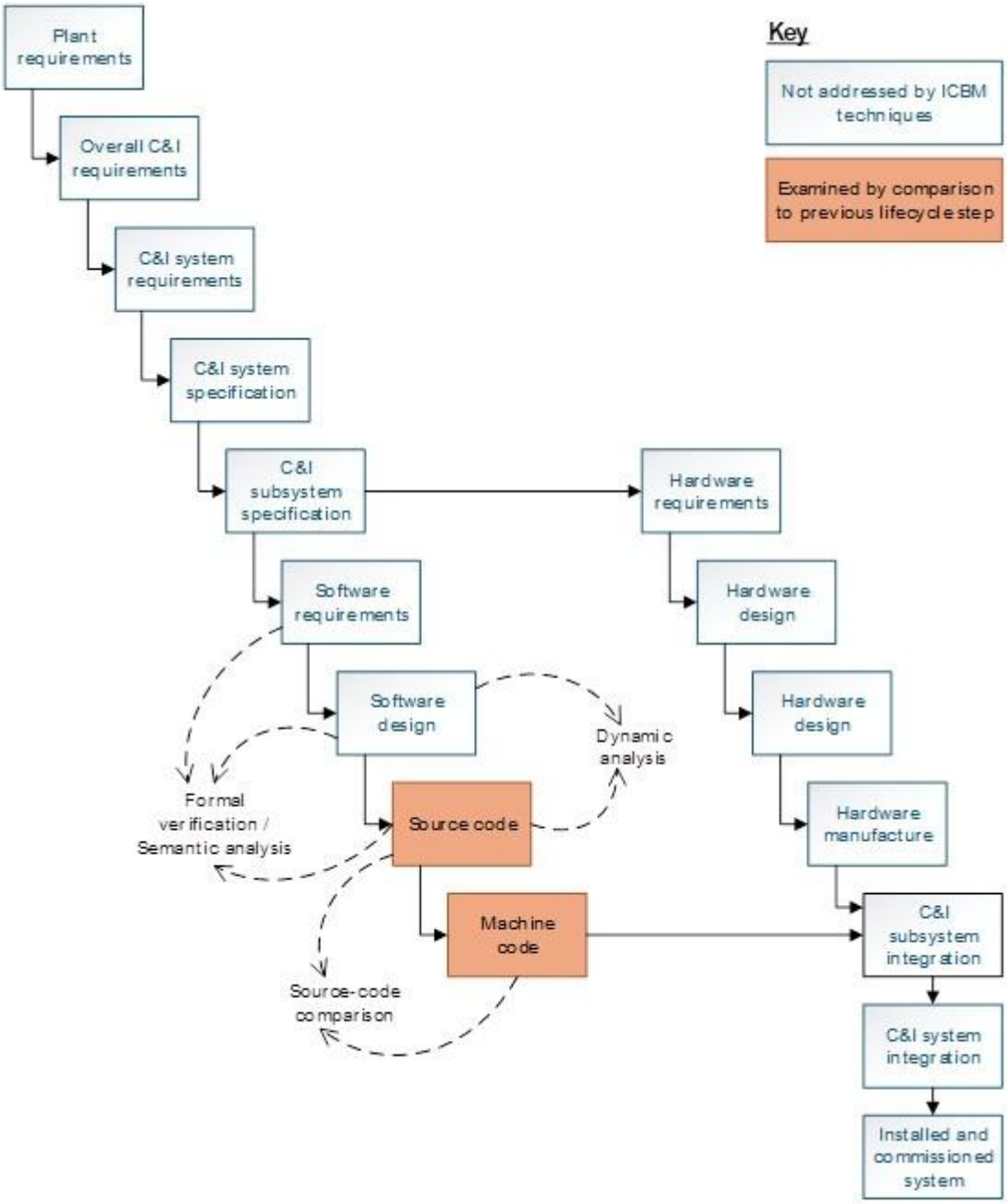


Figure 8 ~ C&I System Lifecycle Showing Focus of ICBM Source-Code Techniques

The effect of this is to add substantial costs into the consideration of whether an in-service improvement is regarded as reasonably practicable. This challenges whether an improvement contributes to reducing risk SFAIRP. Where improvements would be beneficial, but are essentially optional, the additional *post hoc* analysis costs will tend to encourage an operator not to implement them on the grounds of excessive additional cost and effort. This decision will be driven as much by perception of complexity as by a genuine evaluation of costs, but the effect will be that any such incremental improvements to plant safety are unlikely to be made.

The ICBM approach also tightly couples the demonstration of safety to specific tools and techniques. Over the life of a plant or system, advances in technology or methods are hard to adopt because it requires fundamental change to the entire basis of substantiation. The

licensee must maintain the skilled workforce and tooling required to be able to maintain the system (tool obsolescence being an inevitable problem over the lifetime of a nuclear facility), including executing what are often specialist techniques, thus incurring additional costs and barriers to modification through the multi-decade life of the facility. This is a challenge for the development tools and substantiation tools, which, particularly in the case of ICBMs for high-integrity systems, are more likely to be bespoke, further increasing the burden.

It would instead be preferable to adopt a goals-based approach, so that new techniques and methods could be adopted and used to drive later substantiation. Contemporaneously with Sizewell B's licensing, it was noted that the nature of the MALPAS analysis meant that it was process-based (Bruns and Anderson 1995). For example, the analysis is performed uniformly across the code rather than placing additional scrutiny on critical areas, or driving the analysis based on system requirements.

However, to adopt a goals-based approach, one must have some understanding of what the goals are, which is a topic we will revisit in Section 3.

2.7 Statistical Testing

Statistical testing is the only system-level activity that appears in ONR guidance for ICBMs (ONR 2020) (ONR 2023), which purports it to be a means of demonstrating the numerical reliability of the system after it has been validated[53]. This is an unusual inclusion in the first instance, because, despite its presentation as good practice within the regulatory guidance, there is no precedent for its use in licensing protection systems and yet there is strong emphasis on it being needed for Class 1 systems. The PPS at Sizewell B was not licensed based on statistical testing (May et al. 1995) (Hunns and Wainwright 1992)[54]. The proposed statistical testing at Hinkley Point C has not yet been performed and has reported encountering challenges with practicability, leading to divergence from pure statistical testing (Walker 2024). The problems with statistical testing are outlined in greater detail in (Rendell and Pollard 2026) but are summarised here.

Rendell and Pollard (2026) argue that the approach proposed by NS-TAST-GD-046 (ONR 2023) is insufficient in test volume to demonstrate the safety claims placed on multi-function protection systems. NS-TAST-GD-046 proposes ~4,600 tests on the system in order to demonstrate evidence of 10^{-3} pfd for the system, yet in reality the claims for reliability are on the lower-level safety functions in the plant engineering schedule that are allocated to the system. These functions' individual reliability claims cannot be aggregated into a single reliability figure for the system. Instead, Rendell and Pollard (2026) argue that the correct calculation should be ~4,600 tests of *each* of these functions. To suggest otherwise is to offer developers a perverse incentive to collect as much functionality as possible into a single but complex system to minimise the volume of testing required. For example, if a developer has two systems with claims of 10^{-3} pfd each, then based on NS-TAST-GD-046, each

[53] The discussion in this subsection is primarily focussed on the use of statistical testing as described in, for example, BS EN 61508-7:2010, Annex D, performed on the integrated, validated system, for the purposes of providing evidence of the system's reliability in performing its safety functions. This is the form of statistical testing utilised by ICBM programmes. This discussion does not discuss statistical testing as done during the development lifecycle for the purposes of subjecting software or units to high-volume tests, potentially in simulation. The "core conditions" of statistical testing for producing a reliability claim, as utilised in UK nuclear precedent, preclude the use of simulation due to the need to run tests on the same hardware and software as the operational system — research into use of simulation for reliability claims of this type is incomplete, e.g. May (2017). This sub-section does not comment on large-scale (potentially simulated) random testing that may not require an operational profile.

[54] A statistically informed testing campaign was performed after licensing, but available papers imply that it was effectively only performed to ~40% statistical confidence, with this testing only comprising suites of 5,000 tests for the 10^{-4} pfd reliability target.

requires ~4,600 tests for a total of 9,200 tests to demonstrate the reliability of its functions. If the developer combined the systems into one, they could limit themselves to 4,600 tests, and so save considerable time, cost and effort to apparently claim the same level of reliability for all the combined functions of both systems with half the volume of testing[55]. This is intuitively wrong and yet it is the logical conclusion of targeting a "system reliability" that has no meaning, compared to the actual safety claims, which are the reliabilities of each function. Given that, for example, reactor protection systems have multiple safety functions (tens to hundreds), these necessary test-volumes are likely to be impracticable, given that each test is likely to be on the order of minutes to tens of minutes, and use of simulation is unproven for this specific purpose (Rendell and Pollard 2026).

As described in Rendell and Pollard (2026), even if the test regime were executed, the inability to rigidly meet the assumptions required by statistical testing to achieve a reliability measure mean it is likely unknowable what level of confidence in the reliability claim is achieved. Further, deliberately diverging from pure statistical testing methods, e.g. to aid practicability, has further unknowable effects on the numerical outcome. A technique with an unclear outcome cannot meaningfully contribute to a genuine improvement in confidence, and so it is unclear what is achieved by a statistical test of this type.

Similar to the discussion on post hoc analysis of source code earlier, the ICBM-type approach of requiring statistical testing as part of software qualification (ONR 2011) potentially adds an additional disproportionate hurdle to future software modifications. If a modification requires a partial or full run of a statistical testing campaign, then expertise is needed to, for example:

1. Determine what level of testing is needed — a full statistical testing campaign, a reduced-size campaign, i.e. lower statistical confidence, a reduced-scope campaign, e.g. only testing certain fault scenarios? This will require specialist statistical testing expertise and impact analysis of the system modification.
2. Assess and potentially update the operational profile to account for any plant changes in the interim period.
3. Update the oracle to reflect any changes to the system under test.
4. Modify the test harness to accommodate signal changes.

Available research indicates that an answer to point #1 is not straightforward (Korhonen et al. 1997), but in some models may require a larger test suite than the original size target to be run after discovery and fix of a defect, and be dependent on when in the suite the failure occurred (Littlewood and Wright 1997). That a single line of code being modified may impact the whole system is challenging, especially when statistical testing is intended to be a black-box technique.

The test equipment and software may become obsolete faster than the system being tested, meaning that some of the above may even need to be rebuilt from scratch (this might be particularly pertinent given the earlier discussion about the disincentive for making modifications unless exceptionally necessary). In any case, the amount of effort needed is likely to be significant, thus adding further disincentive to make a modification, noting that to achieve full statistical rigour, even a very minor modification may warrant a full test campaign.

[55] If the two hypothetical functions are coupled, such that they would both be demanded by every test case in either of these hypothetical scenarios, then it may be justifiable to cover them both in the same system with only 4,600 tests. However, this is a limited special case and, for example, it is highly unlikely that any realistic test cases exist that would demand every function in a reactor protection system in the same test.

Precisely estimating the time, effort and cost associated with a statistical test campaign is difficult without a specific reference system because there is a dependency on the duration of testing, which is unknown at this stage of the development and depends on the reliability being claimed. However, there are significant elements of effort and cost that can be estimated to a degree because they arise regardless of the claimed reliability. F or example, for a reactor protection system:

1. At least one division of the system must be purchased for testing, in addition to the ones that will be deployed on the plant. The costs for this can be estimated to be of the order of £10m.
2. A test harness must be procured and/or developed. For a multi-functional system, this is estimated to be effort on the order of 10 person-years, with equipment costs ranging from the hundreds of thousands to millions of pounds.
3. Software for the test case generation, oracle and output comparator must be developed, which is estimated to involve development effort on the order of 5 person-years.
4. The event design itself for the operational profiles is estimated to be on the order of tens of person-years.

Additional to this are the costs and effort associated with the execution of the testing itself, with both factors rising with overall test duration because testing cannot be wholly unattended. There will be additional hardware costs associated with the testing, including replacement parts for the test harness, and effort associated with maintaining calibration. There are also effort and costs associated with the supplier of the system, with whom substantial engagement is required to integrate the system into the test harness[56]. In conjunction with the effort and costs established above it is sufficient to observe that the overall effort associated with statistical testing of a single system will be of the order of tens of person-years, and costs in the order of tens of millions of pounds.

The key aspect is that this expense would go to answer a question about the safety of the system that we are not asking. We want to know how reliable each of our safety functions are, but this is not what this expensive technique is able to practicably tell us.

2.8 Conclusions

This section has examined the precedent for applying verification techniques to software after the completion of development, and the use of statistical testing. In the manner performed in an ICBM programme, these are generally found to at best provide a false confidence based on what they purport to demonstrate and do not provide any meaningful benefit to safety, nor to establishing confidence in the achievement of safety at a plant level. The time, effort and cost of these activities as applied to a modern C&I system has also been examined and found to be disproportionate to any negligible improvement in safety or confidence in safety that is achieved. As a result, these techniques are judged to not be useful precedents to adopt in the qualification of modern C&I systems and may actually be detrimental to safety because of their impact of the preservation of qualification through the life of the system, and because the rigid desire for precedent prevents modernisation of existing facilities and new designs.

[56] This interaction with the supplier should also be handled carefully, to avoid "polluting" the independence in a manner similar to that which was described for static analysis.

In each case, the examination has turned to the question of what it is we are seeking to demonstrate by applying each technique. The answer we have landed on in general is that we are not answering anything beyond "d*oes this piece of software behave in accordance with the evidence we already have*?". However, this is a common problem with the ICBM approach based on precedent. It has been, uniquely within UK nuclear regulation, an in-effect prescriptive regime, which contradicts the goal-based approach upon which substantiation is meant to be based. As a result, every licensee or applicant for reactor design assessment has formulated a different question to which the 'required' ICBM techniques are the answer. It is more appropriate to start with the goal and determine the evidence on that basis. This is explored further in the next section.

3 What Question Must Be Answered?

In Section 2, we examined the techniques that have generally been applied to systems based on the precedents set at Sizewell B in the early 1990s, and concluded that they offered at best negligible benefit to safety or justifiable improvement in confidence in the safety of the system. The general approach to this form of independent re-verification after-the-fact was also critiqued because it acts as an inhibitor to safety improvements through the life of the plant.

A reasonable challenge might be that we have not considered alternative techniques and that if some further examination of techniques was undertaken, replacement techniques could be identified that would be more appropriate. However, the more fundamental question is why additional techniques are required at all — what is the principled basis upon which to decide to do more work than the original development and suitably-independent verification required by modern safety standards?

A view is taken in other jurisdictions and in other high-hazard industries that a development process according to a modern, safety-oriented standard, graded according to the safety significance of the system, is sufficient to ensure that the system is adequately dependable within its application. There is an argument that can be made here, where the system and its software have been developed in a lifecycle that begins with a plant-level hazard analysis (the plant design basis in nuclear jargon) and proceeds through a structured lifecycle of safety function definition, system allocation, development, validation, installation, and commissioning. In this case, all decisions and all of the implementation can be traced directly to reducing plant risk and all the evidence generated by the development lifecycle will ultimately trace to this purpose.

This argument may fall down in practice, especially in modern systems where much of the implementation (hardware, software, and HDL) is pre-developed. This could be pre-developed in the form of a platform (like a programmable logic controller or some other computerised platform), or in the form of a commercial off-the-shelf product providing some functionality for the system, e.g. a sensor transmitter. In these cases, the development lifecycle is focused on implementing capabilities that anticipate the needs of future users by following lifecycles that are intended to control or avoid latent systematic faults. Although these capabilities may be essential for the correct operation of the system and may have been developed well, they are somewhat disconnected from the plant-level hazards that the system is trying to prevent, or the plant-level hazards that spurious operation of the system may itself induce. We have in this case therefore arguably lost traceability to plant-level hazards, in that requirements for various system components are no longer directly driven

by the hazard analysis process. Furthermore, there should be concern about the interaction between the various pre-existing software elements as well as between these elements and any application-specific development.

We propose that the goal with CBSIS in general is to demonstrate that:

> *The risks associated with software and/or programmable logic used in implementing a safety function have been reduced so far as is reasonably practicable and to a sufficient level to support safe system operation.*

Satisfying this goal requires an understanding of how the intended and unintended behaviours of the CBSIS could affect plant safety, how consequential behaviours are prevented or controlled, and whether the available evidence is adequate for the safety significance of those behaviours.

This is not a goal addressed by the ICBM approaches in UK nuclear precedent. These look only at software implementation correctness with respect to the software specification, or require some degree of system-level testing that does not focus on safety functions and their implementation directly. They do not examine risk properly, and they do not examine behaviours that may emerge in systems combining software and programmable logic of different origins. It is in this respect that they are extremely limited.

This goal could potentially be addressed by the independent assessment of the development process, as that forms a judgement on the level of safety achieved. However, it is not obvious that the structure of development evidence, particularly where derived from a supplier audit in the case of COTS products necessarily wholly underpins that demonstration on its own.

As part of ongoing study and regulatory engagements in this area, the authors have instead proposed that the plant-level hazard analysis is linked to a software architecture failure analysis (performed using, for example a software interface failure modes and effects analysis for simple software architectures) via successive levels of failure analysis on the system, subsystems and components. This would permit the identification of controls within the overall system that prevent the propagation of postulated software failures, and those software failure modes that can propagate to the plant level with a consequential effect.

Armed with this analysis, each claimed propagation control can be closely examined to see if its specification and verification confirm that the failure that we are trying to control will be prevented. Any unmitigated postulated software failure modes can draw the assessor's attention to the existing verification evidence to explore whether these failure modes have been adequately addressed by the existing development work, using techniques and measures that are appropriate given the safety significance.

In this way, the evidence from the development lifecycle is re-purposed to a demonstration that the risks from the software to the plant have been adequately controlled. If there are gaps in this demonstration, it may highlight the need for extensions to the existing evidence (such as increasing the number of unit tests) or completely new analysis. This work remains in progress, and we hope to report further on it once we have trialled it on a design. However, it is not entirely novel, being an adaptation of the software architectural safety analysis described in Annex E of BS ISO 26262-6, augmented to allow it to be applied after-the-fact.

Nonetheless, given its lack of maturity, the approach outlined above is not presented as a universal replacement qualification method, which in any case would be counter-intuitive in a paper where we seek to identify problems with a prescriptive approach to qualification.

Its purpose here is to demonstrate that alternatives to routine independent re-verification exist, and that these alternatives can begin with a consideration of system behaviour rather than with a predetermined set of techniques.

Care is needed not to replace one form of prescription with another. If our approach were adopted simply because it had become the new precedent, without identifying the question it was intended to answer, the same underlying problem would remain to some extent. The appropriate analysis or verification activity will depend on the system, its application, the available development evidence and the uncertainties material to the plant safety case.

Earlier in this section, we proposed a possible goal, but we do not claim that this is necessarily the best or most complete expression of it. It may require refinement to distinguish between demonstrating that a system is adequately safe and demonstrating that further risk reduction would not be reasonably practicable. Nevertheless, expressing a goal of this kind changes the basis on which assurance activities are selected. It allows a licensee to begin with plant risk, identify the claims and uncertainties associated with the CBSIS, examine the available evidence and undertake further work where that evidence does not adequately support the safety case.

By contrast, the UK approach to date has used precedent for very specific confidence-building measures without clearly articulating the residual safety concern that those measures are expected to address after standards-based development and independent safety assessment have been completed[57]. Licensees are consequently encouraged to begin with historical techniques and then explain how they integrate into the overall safety argument (ONR 2011), rather than beginning with a safety claim or material uncertainty and identifying a suitable means of addressing it.

Regulatory guidance would therefore be more useful if it expressed the objective of any additional software qualification activity. Such an objective should identify what remains to be demonstrated, if anything, beyond production excellence and independent functional safety assessment as defined in standards, while allowing the licensee to propose evidence appropriate to the system, its application and its safety significance. This would preserve independent regulatory challenge without turning the methods used on one project into assumed requirements for the next.

The tentative approach described in this section may not be the best means of satisfying such an objective. More fundamentally, the goal proposed by the authors may not itself be the right one. This uncertainty does not justify reverting to precedent. Rather, it demonstrates the need to articulate and debate the regulatory goal before selecting the techniques by which it is to be satisfied.

Before asking what should replace ICBMs from precedent, it is therefore necessary to establish the unresolved safety concern that additional assurance is intended to address, and the goal against which any proposed activity will be judged. If these cannot be expressed sufficiently clearly to distinguish relevant evidence from irrelevant activity, there is no principled basis for requiring either the historical techniques or replacements for them.

[57] Or, as noted in this paper, some of the precedent is not even that – such as the "need" for statistical testing of reactor protection systems despite it never being done previously for licensing, and the "need" for independent execution of dynamic analysis, despite this technique historically being performed as a development-lifecycle activity.

4 Conclusions

This paper has examined independent static analysis (covering its various forms), independent dynamic analysis, and statistical testing, because these techniques have been repeatedly included in UK nuclear software qualification as activities performed after the development process has concluded; in many cases based on the precedent established during the licensing of Sizewell B. Although precedent can provide useful experience, repeated application has allowed these techniques to effectively become expectations without sufficient reconsideration of the technological context in which they arose or the safety purpose they are intended to serve. Where modern standards-based development, verification, validation, and independent functional safety assessment have already been performed, their routine use as additional independent confidence-building measures provides little additional safety benefit, or justified improvement in confidence relative to their considerable cost, effort, and through-life consequences.

This is not an argument against independent assessment, nor against undertaking additional analysis and testing where these are needed. It is an argument that additional work should address an identified safety claim, material uncertainty, or weakness in the available evidence. Independence of judgement should not be equated with wholesale repetition of development activities.

It is also not an argument that every project should deliberately start afresh and derive its approaches from first principles. Good practice should be recognised, and adopted where appropriate, and the experience of earlier projects should be used to improve the quality and safety of future ones. The argument is that "good" precedent for safety systems should have a material impact on the levels of safety achieved in the context of the system under consideration, primarily via the development process; blindly following precedents on the grounds that it is what has always been done is antithetical to the goal of improving safety overall. Our conclusion is that the precedent for the ICBMs established at Sizewell B is not appropriate in the modern technical landscape.

The possible goal- and failure-based approaches outlined in this paper are intended to illustrate that other ways of reasoning about software risk are available, not to prescribe a replacement set of techniques. The more important need is for the objective of any qualification beyond international standards to be stated clearly. Only then can licensees select proportionate evidence appropriate to the system, its application, and its safety significance; and regulators can then judge whether that evidence is adequate.

Precedent can provide useful experience, but it should not determine the answer before the safety question has been asked. Before requiring either established ICBMs or replacements for them, it is necessary to identify the unresolved safety concern and the goal that assurance additional to that performed as part of a modern safety development lifecycle is intended to satisfy. Without that foundation, there is no principled basis for determining what additional assurance activity might be appropriate, judging its adequacy or deciding whether it is needed at all.

Disclaimers

Opinions expressed herein are those of the authors, not necessarily those of their past and present employers or clients.

References

AREVA NP, and EDF. (2012). *PCSR – Sub-chapter 7.7 — I&C tools, development process and substantiation*. Available from https://www.edfenergy.com/sites/default/files/sub-chapter_7.7_-_iandc_tools_development_process_and_substantiation.pdf. Accessed 14th July 2026.

Atkins. (2008). *MALPAS - Advanced Software Analysis and Verification — Introductory Guide*. Available from http://web.archive.org/web/20240301202919/http://malpas-global.com/wp-content/uploads/download/MALPAS%20introductory%20guide.pdf. Accessed 13th July 2026.

Bruns G, and Anderson S. (1995). *Gaining Assurance with Formal Methods*. In: *Applications of Formal Method*s (pp. 33-54). Upper Saddle River, USA: Prentice Hall.

BS EN 60880. (2009). *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions*. BS EN 60880:2009. British Standards Institution, London.

BS EN 61508-4. (2010). *Functional safety of electrical/electronic/ programmable electronic safety related systems. Part 4: Definitions and abbreviations*. BS EN 61508-4:2010. British Standards Institution, London.

BS EN 61508-7. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems Part 7: Overview of techniques and measures*. BS EN 61508-7:2010. British Standards Institution, London.

BS EN 61513. (2013). *Nuclear power plants – Instrumentation and control important to safety – General requirements for systems*. BS EN 61513:2013. British Standards Institution, London.

BS EN IEC 62138. (2019). *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category B or C functions*. BS EN IEC 62138:2019. British Standards Institution, London.

BS EN IEC 61226. (2021). *Nuclear power plants – Instrumentation, control and electrical power systems important to safety – Categorization of functions and classification of systems.* BS EN IEC 61226:2021. British Standards Institution, London.

BS EN IEC 62138. (2019). *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category B or C functions*. BS EN IEC 62138:2019. British Standards Institution, London.

BS ISO 26262-6. (2018). *Road vehicles — Functional safety. Part 6: Product development at the software level*. BS ISO 26262-6:2018. British Standards Institution, London.

CompCert. (2026). Website: https://compcert.org/. Accessed 27th August 2026.

Hunns D. M, and Wainwright N. (1992). *Software-based protection for Sizewell B: The regulator's perspective*. In: International Conference on Electrical and Control Aspects of the Sizewell B PWR, (IEE Number no.361), Institution of Electrical Engineers, British Nuclear Energy Society, Institute of Energy, Institution of Mechanical Engineers, and Convention of National Societies of Electrical Engineers of Western Europe (EUREL), London.

Korhonen J, Pulkkinen U, and Haapanen P. (1997). *Statistical reliability assessment of software-based systems*. Finnish Centre for Radiation and Nuclear Safety. Available from https://inis.iaea.org/records/0sccc-2x936/files/28028382.pdf. Accessed 26th September 2026.

Littlewood B, and Strigini L. (1993). *Validation of ultrahigh dependability for software-based systems*. Communications of the ACM 36(11):69-80. Available from https://openaccess.city.ac.uk/id/eprint/1251/1/CACMnov93.pdf. Accessed 16th September 2026.

Littlewood B, and Wright D. (1997). *Some conservative stopping rules for the operational testing of safety-critical software*. IEEE Transactions on Software Engineering 23(11):673-683.

May J. H. R. (2017). *Theoretical feasibility of statistical assurance of programmable systems based on simulation tests*. In: 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, Czech Republic.

May J. H. R, Hughes G, and Lunn A. D. (1995). *Reliability estimation from appropriate testing of plant protection software*. Software Engineering Journal, November 1995:206-218.

ONR. (2011). *Generic Design Assessment – New Civil Reactor Build: Step 4 Control and Instrumentation Assessment of the EDF and AREVA*. Office for Nuclear Regulation. Office for Nuclear Regulation, Assessment Report ONR-GDA-AR-11-022. Available from https://www.onr.org.uk/media/tz0lphdm/ukepr-ci-onr-gda-ar-11-022-r-rev-0.pdf. Accessed 26th September 2026.

ONR. (2020). *Safety Assessment Principles*. Office for Nuclear Regulation. Available from https://www.onr.org.uk/media/pobf24xm/saps2014.pdf. Accessed 26th September 2026.

ONR. (2023). *Computer Based Safety Systems*. Office for Nuclear Regulation. Technical Assessment Guide NS-TAST-GD-046. Available from https://www.onr.org.uk/media/3def2aat/ns-tast-gd-046.docx. Accessed 16th September 2026.

Pavey D. J, and Winsborrow L. A. (1993). *Demonstrating Equivalence of Source Code and PROM Contents*. The Computer Journal, 36(7), 654-667.

Rendell M, and Pollard F. (2026). *Statistical Testing: Qualitative Arguments for a Quantitative Outcome?* Safety Critical Systems eJournal, Vol. 5, Iss. 1. Available from https://scsc.uk/journal/index.php/scsj/article/view/64/53. Accessed 11th October 2026.

RTCA/DO-178C. (2012). *Software Considerations in Airborne Systems and Equipment Certification*. RTCA Inc., Washington DC.

Walker L. (2024). *Case studies in operational profile based statistical testing for the quantitative reliability estimation of software based commercial off the shelf (COTS) equipment*. Technical Meeting on Commercial Equipment in Nuclear Applications and Non-Nuclear Standards for Safety Instrumentation and Control Systems, 1st—3rd July 2024. International Atomic Energy Agency, Vienna.

Ward N. J. (1993). *The Rigorous Retrospective Static Analysis of the Sizewell 'B' Primary Protection System Software*. SAFECOMP '93, (pp. 171-181).

Watts S. (1989). *Software row dogs nuclear power plans*. New Scientist, April 1st 1989. Available from https://www.newscientist.com/article/mg12216580-200-software-row-dogs-nuclear-power-plans/. Accessed 14th July 2026.

Wichmann B, Canning A, Clutterbuck D, Winsborrow L, Ward N, and Marsh D. (1995). *Industrial perspective on static analysis*. Software Engineering Journal, 10(2), 69-75.

ISSN 2754-1118 (Online) — ISSN 2753-6599 (Print)

Statistical Testing: Qualitative Arguments for a Quantitative Outcome?

Martin Rendell and Fred Pollard

Rolls-Royce SMR, United Kingdom

Abstract

Statistical testing is a technique accepted by functional safety standard BS EN 61508-2:2010 for higher integrity software-based systems, either as an alternative to black-box testing and field experience or as a technique to support code verification. It has historically been seen as relevant good practice in the UK nuclear industry as a post-validation activity. Its primary usage has been to provide evidence to support an upper bound for the numerical reliability claim on a programmable system. Using past published examples of statistical testing and the authors' own experience of implementing statistical testing systems, this paper evaluates the practicability of using this approach for complex systems like reactor protection systems, covering each of the core conditions underpinning the execution of the technique (accurate test operating environment, test-case independence, determination of pass/fail and zero test failures). We examine divergence from these assumptions, and how this may undermine the quantitative claims of the technique. Additionally, we discuss the validity of the test outcome for a complex system, and in particular whether statistical testing's implicit model of a software-based system reflects what we know about software-involved faults due to its focus on system reliability, and not on reliability of safety functions. We conclude that while concepts and approaches used to implement statistical testing can provide useful qualitative evidence, statistical testing itself should not be considered a practicable method for obtaining high-confidence quantitative reliability estimates for complex systems like reactor protection systems, and as a result the benefits of other techniques should be considered.

1 Introduction

1.1 Purpose

This paper reviews statistical testing as a means of quantitatively supporting a reliability claim at a target statistical confidence level for a complex programmable system such as a reactor protection system[58], through literature review and the experiences of the authors. This review evaluates:

[58] For the purposes of this paper, we consider a complex system to be a system that implements or is involved in the implementation of many distinct safety functions. We use the definition of a safety function provided by BS EN 61508-4:2010: "*function to be implemented by an electrical/electronic/programmable electronic safety-related system or other risk reduction measures, that is intended to achieve or maintain a safe state for the equipment under control, in respect of a specific hazardous event*".

© Rolls-Royce SMR 2026.

- Whether it is reasonably practicable to achieve all the core conditions of statistical testing to an adequate level of rigour, and
- Whether it is justifiable to claim the targeted quantitative results for a complex system like a reactor protection system.

This paper explores several aspects of statistical testing that have previously been identified to be problematic by existing literature (e.g. Littlewood and Strigini (1993), Butler and Finelli (1993), Ladkin and Littlewood (2016), and Daniels and Tudor (2022)) and expands upon them based on experience and further research, and complements this with new contributions — in particular:

- Whether statistical testing provides confidence in safety-function reliability, which is typically what is of prime interest to plant design and the safety case;
- Practitioner observations of oracle and harness design, and the ramifications of inadequately conceived inter-test state-reset design; and
- Observations on the use and practicability of statistical testing for complex modern safety systems.

This paper refers to "software", but most discussion is equally applicable to hardware description language (HDL)-programmed devices such as field-programmable gate arrays.

1.2 Overview of Statistical Testing

The input space and internal state of a software-based control and instrumentation (C&I) system is likely large, such that exhaustive testing is intractable for all but the simplest of systems.

Statistical testing of programmable systems is considered by BS EN 61508:2010 Parts 2 and 3 (referred to as "*probabilistic testing*" in Part 3) as a means of supporting a reliability claim on a software-based system in a specific configuration for a particular application, and has been promoted in the UK nuclear industry since the 1990s. BS EN 61508-2:2010 grades the technique's recommendation level and technique effectiveness according to Safety Integrity Level. Proponents of statistical testing argue that it is a useful technique to support the qualification of a software-based system, on the basis that there are no other agreed methods of quantitatively estimating the reliability of such a system in advance of its operation, e.g. Kuball *et alia* (2004), other than through analysis of prior-use data if the system already exists, as described in BS EN 61508-7:2010.

A statistical testing suite subjects the test object to a series of demands, statistically representative of the expected operational profile that the real system will experience in its target application. In the case of a reactor protection system this demand could be a reactor plant fault that requires the protection system to take a series of actions to protect the plant (such as tripping the reactor and subsequently executing other safety functions to bring the plant back to a safe state).

Given N passed tests and zero failed tests (where a "pass" is when the test object has performed the appropriate safety function(s) within defined accuracy and timing tolerances), statistical testing theory allows for a claim of a probability of failure on demand (pfd) of θ

to statistical confidence α. This relationship is given by Equation (1), with typical numbers provided in Table 12 overleaf[59].

$$N = \frac{\ln(1-\alpha)}{\ln(1-\theta)} \quad (1)$$

Table 12 ~ Statistical Test Suite Size

α	N	
	$\theta = 10^{-x}$	$\theta = 10^{-4}$
50%	$0.69/\theta$	6,931
90%	$2.3/\theta$	23,026
95%	$3.0/\theta$	29,957
99%	$4.6/\theta$	46,052

Note that Equation (1) is derived from a frequentist approach, which deals with aleatory uncertainty on the condition of model assumptions being true. Some discussion in this paper uses Bayesian statistics instead, where epistemic uncertainties in the test arrangements can be factored in. Regardless, the discussions in this paper are ultimately applicable to both, though in some cases in different ways.

Statistical testing has been used in the UK nuclear industry as a means of quantitatively demonstrating confidence in the system's reliability. Statistical testing is recommended by NS-TAST-GD-046, the UK Office for Nuclear Regulation's (ONR's) guidance to its own inspectors to demonstrate software-based systems reliability (ONR 2023).

This paper distinguishes between statistical testing used to support a numerical reliability claim and operational-profile testing used as a qualitative test activity.

1.3 Structure of This Paper

This paper provides commentary on statistical testing across the following main sections:

- Section 2 summarises a review of literature on uses of statistical testing in industry.
- Section 3 looks at the challenges involved in implementing a statistical testing campaign in such a way that it supports the core conditions of statistical testing.
- Section 4 examines how statistical testing relates to the plant safety case, and whether the technique's relationship with system integrity and safety-function reliability is appropriate.
- Section 5 considers whether aspects of statistical-testing methodology are worth applying in non-statistical contexts, drawing on published case studies of successes from statistical-testing campaigns.
- Section 6 provides concluding remarks.
- Section 7 provides brief thoughts on future work as a result of this paper's conclusions regarding alternative approaches and techniques.

[59] Equation (1) is the classical frequentist single-sided bound for N independently and identically distributed Bernoulli trials. A Bayesian formulation also exists, which in practical terms provides very similar numerical results in terms of the relationship between confidence, reliability and number of tests.

2 Statistical Testing in Industry

In the UK nuclear industry, statistical testing has been applied and completed for different systems and components. This includes the following examples:

- Gough and Kuball (2014) describe execution of statistical testing by EDF Energy on a subset of functionality on the replacement Dungeness B Data Processing and Control System (DPCS). This comprised 460 statistical tests on the premise that it would demonstrate 99% confidence that the alarm functionality would meet its 10^{-2} pfd reliability claim, plus 26 "padding" tests to provide additional coverage of under-tested scenarios.
- Kuball *et alia* (2004) describe a research project running a form of statistical testing on a programmable logic system, which revealed a non-safety-significant defect that other test techniques in the same study had not detected.
- EDF Energy performed statistical testing on several smart devices and programmable logic systems, such as a motor protection relay, a pressure relief valve controller, a system comprising two devices providing spine break detection, and several conductivity monitors (Kuball 2024) (Walker 2024). The first three were performed as part of activities to support a claim of 10^{-2} pfd; the conductivity monitor testing was for a claim of 10^{-3} pfd (after 2oo4 voting[60]).

No complex programmable protection system has been subjected to statistical testing as part of nuclear-site licensing activities in the UK. There are two programmable protection systems to consider in relation to this claim: the Sizewell B Primary Protection System (PPS) and the Protection System (PS) at Hinkley Point C.

In the case of the Sizewell B PPS, the nuclear site licence was granted on condition of a successful campaign of dynamic testing, not statistical testing (Hunns and Wainwright 1992) (May et al. 1995). While the for-licensing suite size of 50,000 tests correlates with Table 12 (i.e. 99% statistical confidence in 10^{-4} pfd), the Nuclear Installations Inspectorate (NII, the contemporary nuclear regulator) determined that "*the test results could not be used in this way because the tests were not conducted according to the necessary rules of statistical experiments*" (May et al. 1995). Note that the operational profile used for the tests was restricted to infrequent plant faults, as these were the faults where the laddic-based secondary protection system was not claimed (Tooley and Stokes 1992). After licensing, the owning utility company "volunteered" to perform an additional 5,000-test statistically representative test suite, claiming a resultant ~99% confidence in a 10^{-3} pfd claim (Littlewood and Wright 1995) (Littlewood and Wright 1997). Note that it is not clear from available information how this claim is supported by the test-suite construction, which was designed to show the PPS's reliability for fault scenarios for which functions were required to act with 10^{-4} pfd, i.e. no secondary protection. Furthermore, evidence was not found that all prior NII objections preventing statistical claims were resolved in the statistical experiments. In particular, in the description by Tooley and Stokes (1992) the eleven test scenarios in the initial campaign were focussed on a subset of functions, implying that the operational profile was incomplete. Tooley and Stokes (1992) argued (in the context of the non-statistical testing that they describe) that this was acceptable as the majority of other functions were tested by these scenarios anyway, but this is irrelevant for a later statistical testing campaign, where there is a need for coverage of the true operational profile as part

[60] Two out of four

of the statistical core conditions[61]. May *et alia* (1995) do not indicate that the statistical testing experiments would include any more scenarios than the original dynamic-test campaign.

The Generic Design Acceptance (GDA) submission for Hinkley Point C (HPC) committed to a statistical testing campaign of 50,000 tests to support the 10^{-4} pfd claim on the PS (ONR 2012). However, this campaign has not yet been executed due to the development lifecycle stage of the plant[62]. Walker (2024) indicates a shift from a pure statistical testing campaign to a more hybrid approach with redistribution of test cases within the test campaign. Other prospective nuclear new-build licensees in the UK, such as Horizon Nuclear Power's UK Advanced Boiling Water Reactor (ONR 2017) and Westinghouse's AP-1000 (ONR 2011a), also proposed statistical testing of their respective protection systems in their GDA submissions, but these new-build projects were shuttered before work could progress beyond desktop-based feasibility studies.

Outside of the UK nuclear industry, there is less published evidence for the implementation of statistical testing. During research undertaken for this study, limited examples were found in the public domain; the most well-described was statistical testing of the trip functionality within the loop-operating control system of an Advanced Test Reactor experimental loop at Idaho National Laboratory (US NRC 2017). In that paper, the interpretation of the test results provided an illustration of the epistemic uncertainty in statistical-testing outcomes[63]. "*Statistically valid, trajectory-based random testing*" was described as necessary for the Canadian CANDU shutdown system software (Ichiyen 1998), though further detail is not available. "*Reliability qualification*" is described in a CANDU software engineering standard (Ontario Power Generation Inc. 2013), including reference to key requirements in statistical testing. A guideline document (Strigini and Littlewood 1997) for application of statistical testing indicates that the European Space Agency was interested in statistical testing in the 1990s, but the extent of its application is not known to these authors.

There is mention of "*reliability demonstration tests*" including for software-based systems in the non-nuclear standard IEEE 1633-2016, though this comes from a commercial-product testing perspective that balances customer and manufacturer risk and looks to identify defects likely to be revealed in operation and/or to calculate a mean-time-before-failure (rather than pfd) measure. IEEE 352-2016 describes "*acceptance sampling*", though based on the sample figures used it does not seem to consider it for numerical targets more onerous than 5×10^{-3}; the descriptive text refers multiple times to "*quality control*" rather than particular safety-context intent. Relatedly, the US Nuclear Regulatory Commission (NRC) regulatory structure "*relies on the use of a qualitative approach for determining software reliability*" (US NRC 2022). Many other standards across numerous industries do not attempt to quantify the contribution of software to the overall reliability of a system. For example, in general functional safety (e.g. IEC 61508) it is generally accepted that quantified reliability relates to random hardware failures and is distinct from systematic safety integrity, which "*cannot usually be quantified*". Other industrial standards such as aerospace (e.g. SAE ARP4754A, RTCA/DO-178C), rail (e.g. EN 50129, EN 50716), and automotive

[61] Note if the contents of Section 4, where we discuss whether we are testing systems or functions, is considered here then the situation is worse — the "collaterally tested" functions are tested using only a subset of the operational profile scenarios relevant to them.

[62] As of 2024, the predicted cost is in the millions of pounds, and the project is still several years away from completing to support fuel-load (Walker 2024).

[63] A test suite of 10,000 tests was performed to provide evidence to support a 10^{-4} pfd claim on the software's trip functionality, where 5th (5×10^{-6} pfd) and 95th (3×10^{-4} pfd) percentile estimates for the pfd were calculated. These figures are based on a uniform prior, zero observed failures across the 10,000 tests, giving a posterior distribution of Beta(1, 10,001).

(e.g. ISO 26262) similarly separate out quantified reliability from systematic failures from elements such as software, controlling the latter via process controls. This is in contrast to the UK nuclear approach, which arose in the late 1980s and early 1990s when most such standards did not exist, and which has generally required quantification of safety function reliability in support of demonstrating sufficient reduction of risk.

There are no other recent statistical-testing projects (in the sense discussed in this paper) identified by this literature review, though an International Atomic Energy Agency (IAEA) paper suggests that the technique is "*often*" used. IAEA (1999) lists statistical testing as an "*optional*" technique. A regulatory common-position paper (ONR 2024) notes that statistical testing can be used either as a confidence-check during validation (i.e. as per ONR (2023)) or as an activity during the development lifecycle's verification (akin to Musa (1993)).

All examples of statistical testing identified in this review are believed to have been performed in an open-loop approach[64]. No evidence was found for closed-loop statistical testing of large software-based systems during this review; for example, Sizewell B PPS (May et al. 1995), Dungeness B DPCS (Gough and Kuball 2014), and Hinkley Point C PS (ONR 2011b) are all described or implied in open-loop terms, i.e. pre-defined input transients for each test case[65]. The comparison between open- and closed-loop is beyond the scope of this paper, but warrants consideration when building a statistical test harness[66].

3 Challenges of Implementing Statistical Testing

3.1 Core Conditions

Existing literature places the following *core conditions* on the implementation of a statistical testing campaign:

1. Each test case shall be selected from the (known or predicted) operational profile of the real system, such that the number of test cases based on each operational-profile scenario is proportional to the frequency of that scenario in the operational profile.
2. The test object's behaviour shall only be dependent on the stimuli received in relation to the current test case; i.e. previous tests shall not affect the results of the current test case.
3. The test object shall use the same hardware and software version/configuration as the real system.
4. There shall be a means of checking whether the test object has provided the correct response to each test case.

[64] Open-loop testing: the test object's behaviours have no effect on test-case stimuli presented to it, usually because the test-case data is generated in advance of test execution.

[65] Closed-loop testing: the test object's outputs are fed back into a live simulation of its environment, such that the following time steps may be influenced by the test object's behaviours. This approach can present additional challenges for test-harness integration, but can provide more powerful evidence (see Section 6).

[66] Various studies (for example Leveson (2004), UK HSE (2003), EPRI (2008), and Prokop (2024)), have noted that, typically in dangerous behaviours of a system, every component in the system operated exactly as intended. A study of space systems failures since Ariane 5 observed that standard V&V processes are generally effective at avoiding errors in translation from requirements to code that cause failures, whereas errors in data and in particular specifications were the cause of failures studied (ASCA 2007).

5. In order to estimate statistical confidence α and probability of failure on demand θ, there shall be no observed failure by the test object to correctly respond to a demand.

These core conditions contribute to the justification that the statistical assumption of independent and identically distributed (i.i.d.) demand-sampling from the real operational profile is achieved.

Note that core condition 3 (same software/hardware version) is effectively derived from core condition 1 (accurate operational profile) because the software's environment includes its system hardware and the behaviour of each part of software. They are treated separately in this review as they can be encapsulated neatly as separate subtopics, though often in literature the "same software/hardware version" condition is not enumerated explicitly, e.g. see US NRC (2026).

Adequately achieving all five core conditions is especially challenging for large, bespoke systems such as a reactor protection system and for systems with high reliability targets. BS EN 61508-7:2010 explicitly notes that "*in practice, it is very difficult to demonstrate ultra-high levels of reliability using these techniques*". IAEA (1999) states that:

> *...the production of the test data and the evaluation of the correct results become an enormous and demanding task. The effort to establish the part of the input space to be tested and the distribution of test data is also very large.*

The following subsections consider each of these core conditions and the challenges associated with implementing them. These subsections highlight some of the ways in which it is difficult or impossible to quantitatively support the core conditions, which undermines the ability to make quantifiably confident claims of the pfd. An inability to adequately meet all these core conditions makes it difficult to justify the i.i.d. Bernoulli demand models underpinning Equation (1).

Figure 9, overleaf, shows the key constituent parts of a test system, all of which need to be adequately designed, built and used in such a way as to fulfil the core conditions. A weakness in any of them may result in an unquantifiable effect on the outcome, if that weakness is even identified by the test engineer at all[67]. Inadequacies may affect the statistical conclusions' veracity (e.g. a defective oracle does not detect a fault) or validity (e.g. the test operational profile omits key parts of the real operational profile).

While the following terms can be used interchangeably to some extent, this paper uses the following definitions:

Initiating Event: For a test suite for demand-based functions, this is the event that marks the beginning of the demand on the system's functions. For a reactor protection system, this might be a pipe break in the reactor primary circuit.

Scenario: A definition of the progression of the plant properties following an initiating event. This can be used as the basis for test cases.

Test case: An instance of a scenario, defining a series of inputs to the test object.

[67] The "simulated subsystems" box in the Figure refers to where parts of the C&I system are not directly tested, for example redundant divisions of a reactor protection system.

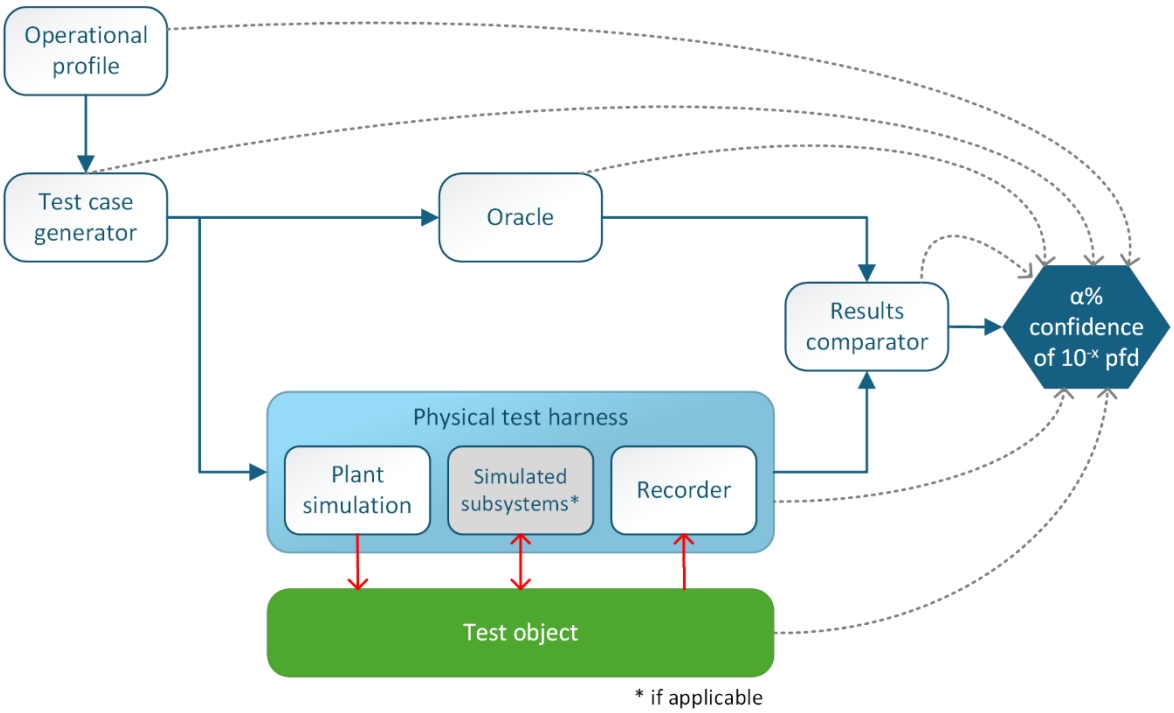


Figure 9 ~ All Statistical-Test Harness Components Contribute to the Veracity and Validity of the Statistical-Testing Outcome

3.2 Implementation Challenge 1: Building the Test Operational Profile

Establishing an accurate test operational profile is a significant undertaking[68]. For a reactor protection system, there is an inherent need to model a large proportion of the plant. There may be many different plant fault scenarios[69]. However, unlike in typical transient analysis, it would not be valid for a statistical testing campaign to only model the "bounding" plant faults, because there is no definable relationship between plant fault severity and likelihood of failure of a protection-system software function. The operational profile defines demands on the test object, but the demands may be affected by other plant systems (e.g. secondary protection systems, control and prevention systems), each of which may actuate or not actuate (either by design or due to a fault). This adds significant complexity. To achieve a well-detailed operational profile for a reactor protection system will require years of analysis effort, involving sensitivity analysis and determination of what can and cannot be modelled with the simulation tools[70].

Can a probabilistic safety assessment's conservative best-estimate (50% confidence) plant initiating-event frequencies, which are used to generate the operational profile distribution, adequately support a statistical-testing outcome of α= 99%? Bishop and Povyakalo (2017) consider that the frequency of each scenario in the modelled operational profile may be inaccurate, and that this may affect the value of α. Bishop and Povyakalo demonstrate that

[68] This literature review revealed relatively emotive language being used to describe this task, such as "*a task to reckon with*" (Horgan and Mathur 1996) and "e*normous and demanding task*" (IAEA 1999).

[69] Walker (2024) states that Hinkley Point C's PS statistical testing covers 150 separate initiating events across all plant states, covering ~70 unique design-basis initiating faults, across a wide range of initiating-event frequencies.

[70] In fact, depending on the capabilities of the simulation tools, there are some fault scenarios that may not be possible to model in a way that is useful for the test operational profile; this would represent a gap which will need justification.

extra tests can be added to the suite as "padding tests" to reach the original target α, but this has the undesirable effect of increasing the overall test-suite size.

Even if the relative frequencies of initiating events are all representative, the modelled operational profile can never truly be complete[71]. However, it seems reasonable to suspect that the areas of the system's design that are more likely to contain flaws are the areas where a possible operational demand was not foreseen. This might be in terms of a plant or systems fault, or more likely a "random" operator or maintenance error or action on the plant unforeseen by the system design. Connected devices, e.g. sensors, may not behave exactly as simulated. Fault studies may consider the effect to be bounded by plant faults (e.g. an operator erroneously opening a valve may be equivalent to a pipe break) but can we be sure there is no difference in effect on the test object's stimuli? Ageing, upgrades, and replacements in the plant will further perturb the true operational profile in ways that cannot be predicted before plant commissioning; arguably, the best we can claim is that the statistical-testing operational profile is accurate for the moment of plant switch-on only. Strigini and Littlewood (1997) acknowledge ageing effects, but do not provide a solution, even within the scope of their paper which is for smaller and less complex environments than a nuclear power plant. If the same system is to be installed in a different environment (e.g. a reactor protection system in a different plant that is not perfectly identical) then consideration is needed about whether the test operational profile is still valid.

Miranda and Bertolino (2017) sum up the challenge's significance succinctly, with reference to Lyu (1996), when they say,

> *... operational profile-based testing is grounded on the notion that not all faults have the same importance: depending on how it will be exercised by the users, a program can show quite different levels of reliability.*

Note that while Musa (1993) suggests that operational-profile strict accuracy is not necessary to gain improvements, this is in relation to operational-profile testing during system development for the purposes of detecting defects early, not for estimating high-integrity system reliability. This distinction is made clear by Strigini and Littlewood (1997), who note that:

> *... errors in the operational profile can in theory cause the estimation to be radically wrong (taking any value between 0 and 1).* […] *There is insufficient experience for a tester to know in advance all the approximations that can be allowed without causing excessive error.*

For a reactor protection system, it will be impossible to validate the test operational profile by comparing to the system once it is in operation, because its demand frequency is (hopefully) so low that there will be a minimal body of evidence, even using data from every reactor in the world (IRSN 2017). The more likely outcome of attempted validation would be to find that a real fault scenario diverges from what was modelled in the test scenario, due to any of the reasons discussed in this subsection. For non-statistical testing, divergence would just be a limitation of the simulation, but for statistical testing it undermines the validity of the statistical assumptions.

As highlighted by Littlewood and Strigini (1993), reliability is a measure of a product working in a particular environment. We might be measuring the reliability of the system as a test object in the test laboratory, but there is a lot of work to show this environment allows the reliability result to be translated to reality.

[71] As noted by IRSN (2017), there are too many different input sequences to know which are likely, possible or not possible. BS EN 60880:2009 notes that the difficulty in defining the operational profile accurately increases with number of inputs.

3.3 Implementation Challenge 2: Ensuring Test-Case Non-Interaction

There are two main options for ensuring non-interacting test cases: 1) reset the test object (e.g. power cycle and return to a known initialised state), or 2) "flush" any internal state through carefully selected series of inputs over a defined period. Both approaches usually require a settling period to ensure the internal state has reached a stable, static and predictable state after the reset/flush.

When reinitialising the test object through either means, the re-initialised state must be understood. This requires assumptions of the internal workings of the system, which is at odds with the "black-box" intended nature of statistical testing, especially for more complex systems[72]. Kuball and May (2007) suggest an approach for clearing the internal history, though it is not clear how easily scalable this is to larger systems with interconnected logic (it is likely for example that a reactor protection system will have minutes or hours of history to clear), and may be susceptible to undocumented or otherwise unexpected or opaque internal history, which is more likely for more complex systems.

Two potential options for confirming history length are to perform static code analysis or to instrument the executable during testing (May et al. 1995). However, the static analysis may not be complete until after statistical testing needs to start (particularly challenging if test infrastructure changes are needed, or if the analysis reveals very long reset times). Instrumenting the executable under test (potentially a significant undertaking to perform exhaustively due to complexity) would violate the "same version" core condition of statistical testing.

This reset challenge is a reason why ONR's (2023) guidance to inspectors recommends that systems are designed with statistical testing in mind; however, this can be challenging to achieve in practice, especially when building applications on pre-existing platforms, rather than purely bespoke systems. Strigini and Littlewood (1997) suggest methods for designing systems as such, though for a reactor protection system the limiting factor is likely to be the plant requirements (i.e. the application functionality needs to have timers ranging from seconds to hours). Ultimately, there is unlikely to be enough incentive for manufacturers to design their systems to support statistical testing directly, especially on platforms with significant existing history, and even with such functionality implemented, there would be a question of whether the statistical testing engineer can rely on this functionality or would have to confirm it independently.

With the *possible* exception of static analysis[73], none of these reset approaches will definitely reveal defects in the system where internal states and memories do not behave in the expected manner, unless these defects are directly revealed by any pre-test checks or the test case itself. It could be argued that if the test case does not reveal a defect, then that defect has no functional effect, but this conclusion is dependent on the validity of the test case and does not acknowledge that we are still only sampling a tiny fraction of input space. Defects due to long-term operation, e.g. counter overflow (EPRI 2010), or other background activities, e.g. exhausting memory (Coolen 2012), may not be detected during statistical testing, especially if the test object is physically reset between tests. Ladkin and Littlewood (2016) discuss masked dependencies that may not have been recognised; without extensive static analysis it may be difficult to prove that none of these affect reset and re-initialisation.

[72] This is one of the reasons statistical testing of non-simple systems usually ends up being "grey-box" testing in practice.

[73] "Possible" because calculation of memory length for a system comprising concurrent subsystems may require very detailed static analysis with a variety of techniques.

Such defects would undermine the assumption that the system's response to a demand is independent of the time (and behaviours) between reset and demand.

Test-case independence (or at least control of side effects) is usually required by other testing techniques as well (not just statistical testing). However, the crucial distinction is that statistical testing is specifically trying to demonstrate the numerical probability of failure on demand, and so is reliant on test independence to support the quantitative outcome.

3.4 Implementation Challenge 3: Test Object and Real System Are Identical

The constraint of the test object being identical to the real system (as identified in ONR (2023) and Ontario Power Generation (2013)) exists because the hardware/software versions and their configuration effectively contribute to the operational environment of the software (see core condition #1). It helps to narrow down the state space that needs to be explored, since a single difference in configuration file or software logic could change the movement through the system's state space in unpredictable ways.

This system configuration constraint means that statistical testing for reliability determination cannot commence until after final validation of the system[74]. For a bespoke system, particularly something like a reactor protection system, this is likely to place statistical testing on the critical path for plant commissioning and operation, because the software is likely to be finalised close to plant operation dates in order to account for other plant changes on which the system is dependent. Considering that a test suite may take months or years to execute[75], this places pressure on the statistical testing campaign to complete. Running multiple instances of the test setup in parallel (i.e. multiple test objects) would of course reduce this, but at significant expense due to the size of these highly specialised pieces of high-integrity equipment. Running simulated copies in parallel would undermine the same-configuration assumption.

Available literature (for example, Tooley and Stokes (1992), ONR (2012), ONR (2020) and ONR (2011a)) indicates that any statistical testing (or in the case of Sizewell B, dynamic testing) performed or planned to be performed on nuclear reactor protection systems focuses on a test object that is a copy of only a single physical division of the system (i.e. one of the several redundant copies of the same protection equipment and logic), with other redundant divisions simulated by the test harness. As noted in regulatory reports, this requires justification[76]. Without adequate justification, the test suite size will need to be multiplied by the number of redundant divisions (this may be challenging if divisions are not using an identical binary and hardware configuration, which is likely).

---

[74] See May *et alia* (1995) and Strigini and Littlewood(1997). Gough and Kuball (2014) argue for waiting for the final validated version from a practical perspective as well.

[75] Tooley and Stokes (1992) indicates that in the initial version of the Sizewell B dynamic testing, each test took on average 3 minutes (whereas Littlewood *et alia* (1998) cite 5 minutes per test), and Dolan (1994) states that the ~50,000-test Sizewell B campaign took 6 months. We posit that the average-time difference may be due to different iterations of the test suite (perhaps with changes to testing approach), and note that 5 minutes per test aligns with 6 months test duration, if running tests "24/7". In modern protection systems, with increased autonomy and complexity, any means of accelerating or truncating test cases are less feasible. Gough and Kuball (2014) note that 486 transients (for a system with a 10^{-2} pfd claim, plus padding tests) averaged 4.75 hours, taking a total of 96 days. For a 46,000 test suite with average test length (including reset time) of 30 minutes, the campaign would need to run continuously for nearly 1,000 days. In addition to the nominal time to run the tests, there is usually a need for additional padding time to account for pauses in testing, e.g. for test system maintenance, problem-solving of the test setup or investigating oracle discrepancies (which for Sizewell B took considerable effort (Arthur 1993)).

[76] In part because protection systems' divisions' application logic are not usually completely identical, e.g. ONR (2011a). Inter-division communications will need to be realistically modelled, and differences between the object code of the real division and the object code of the simulated divisions, and the resulting timing behaviours, may exist.

There appears to be no conclusive research into using emulation or simulation of the test object to accelerate statistical testing, and we are not aware of any research or examples of a statistical testing campaign being performed with emulated test objects that has been able to quantitatively support the statistical testing theory convincingly without making qualitative arguments that move us away from the core conditions. May (2017) developed a theoretical approach for running a large number of tests on an emulated object and then a small number of confirmatory tests on a real test object, to support a quantitative estimation of α and θ. However, May's approach has not yet been developed further.

It seems inarguable that running tests in simulation will provide some qualitative confidence at least, even if it cannot formally support a quantitative confidence due to the hard-to-quantify assumptions needed. Since it is possible to run many more tests in simulation than on a real system, does a focus on calculating α distract us from taking steps to achieve more general benefits?

Another option for tackling the programme/schedule problem is to perform statistical testing on a version of the system that precedes the final validated version. This violates the core condition of testing the same version as in operation, so the value of the testing is diminished. A successful pre-final version statistical testing campaign will give some confidence that there is a good base level of reliability, but a single defect fix or hardware/software change made for the final version could introduce (or unmask) a defect, thus changing the system's reliability "*in subtle but unknown ways*" (Strigini and Littlewood 1997). It may be possible to address this by running (on the final version) a second full campaign or a reduced-size campaign. The reduced-size campaign is likely to need to be very short to allow completion prior to operation, and a full re-test would have to be finished after the plant is operational unless the operation date is delayed, so there is a risk that we would only discover that the final protection system's reliability is less than expected once it is already in use[77]. A partial re-test would be challenging to justify quantitatively, since it would need to make further assumptions on the homogeneity of the software (or alternatively open the black box to work out the validity of assumptions, and thus tie our understanding of the software behaviour to its implementation characteristics rather than purely to its response to test trajectories). We assert that a partial re-test would be a qualitative demonstration only; but if a partial re-test is a qualitative demonstration, what was the point of the pre-final test having a quantitative goal? Could we have made better use of test time on the pre-final version if we were not chasing quantitative outcomes?

As discussed in Sub-section 3.6, the need to use the final version results in additional project schedule risks — the testing would be performed at the end of the project (after the development lifecycle has finished) so the onerous requirements of statistical testing in response to test failures would have an even more severe effect.

Note that the discussion around this implementation challenge has focussed on the software version of the real and test objects, though the same stipulation also should apply to the hardware. It is not inconceivable (and indeed not quantifiable with respect to the statistical-testing confidence level) that minor differences between the hardware in the test object and the hardware in the real system may affect the software behaviour. Further, the effect of differences in local electromagnetic interference may change inputs to the software. While

[77] This becomes both a safety risk (fuel is present in the core and we have now identified a defect) and a reputational risk (we have turned the system on before running all the tests). The reputational risk aspect relating to protection-system testing is one known all too well to Sizewell B, where its initial 50,000 test run showed ~50% discrepancies. These discrepancies were all demonstrated to be weaknesses of the test harness, but the numbers were obtained by the British press and sparked public concern, even leading to a motion in the UK Houses of Parliament (Hansard 1993).

potentially negligible, this is just another example of where we have to make some qualitative judgement on how representative our test setup is if we want to support the statistical-testing quantitative outcome — and, for examples such as those discussed in this subsection, there is no evident basis for quantifying the relationship between such differences and the claimed confidence level.

3.5 Implementation Challenge 4: Determining Pass/Fail (Detecting All Faults)

When statistical testing is carried out on a system that performs one or more safety functions, what should the acceptance criteria for a test be? In statistical testing, it is that the test object has performed the correct actions at the correct time, where "correct" is in reference to the system requirements specification for the system under test. Tolerances are allowed for the timing of the actions, because of real-world delays like propagation of signals through asynchronous components, sample rates, and so on. No tests shall have unexplained failures if the statistical confidence in failure rate is to be achieved.

Determining "correct" traditionally involves construction of an "oracle" tool by the statistical-testing team, which models the system requirements independently of the test object, much like in other forms of testing. This can be a significant undertaking especially for more complex systems.

While the oracle should not use development artefacts from any software/hardware created after the finalisation of the system requirements, in reality the degree of independence from the implemented system often ends up being weakened slightly so that the oracle can account for how the implemented system actually behaves (e.g. time response, algorithm choices). A purely black-box approach to building the oracle is often not possible when accuracy and precision need to be high (e.g. when testing safety-critical systems), and so diligence is required to ensure that common-mode failure is not established between the oracle and test object as a result of "peeking" at test-object design data — such failures may lead to false negatives.

On the other hand, an oracle that is low-fidelity, or overly stringent, can lead to many false positives (tests marked as fail that should have passed), which can significantly increase the manual workload to show that the test object is not to blame, adding effort at the tail end of a test campaign. For example, Sizewell B's statistical testing saw ~50% of its initial ~50,000 dynamic tests marked as "discrepant" by its oracle (Dolan 1994) — the majority of these were due to oracle shortcomings[78], and all were shown to be false positives (Arthur 1993). While oracles like this are now better understood, and so likely to have a better accuracy, they can pose a risk if inadequately conceived or constructed. Building an oracle is hard, as it requires implementation of the system requirements in an ideal model, while also accounting for behaviours of both the physical test harness and the system under test. If redundant divisions are simulated as part of the harness, this can add further complexity.

Ladkin and Littlewood (2016) look at this implementation challenge from a different perspective. If a test fails because of defect *A*, then the conditions required to activate defect *B* may not be reached in the test. As a result, we may fix defect *A*, but defect *B* would only be revealed on the second run (if at all, as it could be in an uncommonly exercised part of state space). Linking to implementation challenge #5, this may place severe pressure on the test schedule.

[78] Though many were due to other aspects of the test harness, e.g. the electrical simulation and monitoring.

Littlewood and Wright (2007) use a simplified model of a test campaign as an input to a Bayesian belief net, and find that the more test cases that are successfully executed with no failures, the greater the likelihood that the oracle is passing tests without justification (i.e. the oracle is failing to detect or masking genuine defects in the test object) and the greater the likelihood that the specification is wrong, and so the lower the confidence in your statistical testing result. The paper suggests this result is unintuitive, but it is not inconceivable that an oracle may be unintentionally built to be lenient on the test object due to its design being influenced by the implementation under test. In reality, to reduce false positives, trial campaigns might be run as part of the test system's acceptance testing — this requires rigour to ensure that defects in the test object are not falsely attributed to issues in the test infrastructure.

Knight and Leveson (1986) conclude that n-version programming does not guarantee absence of common failures. Analogously, consider that the test object and the oracle (including test harness) are two diverse systems implementing the same functions. In this case, common-cause failure is possible, and indeed if one of the diverse systems is less reliable than the other (i.e. the oracle, which is not high-integrity software), then its own reliability will dominate the overall probability of the two systems coincidentally failing to actuate a safety function. Coincidental failure of the oracle and the test object to actuate a safety function at the right time may result in the test passing instead of failing (false negative). In a system with many safety functions, this false negative may be missed by the test engineer due to the correct concurrent actuation of many other safety functions — i.e. the test engineer sees that there was some safety response, but does not see that all necessary safety responses occurred.

In summary, because the oracle is software-based, its own reliability is not quantifiable. If we cannot guarantee a perfect oracle, or quantify its reliability, how do we support a 99% confidence calculation in the reliability of the system under test?

3.6 Implementation Challenge 5: No Tests Shall Fail

Why is this a challenge? Surely this is the ultimate goal of any test suite? This subsection highlights how a test failure may challenge the project schedule, where such a failure would leave the accumulated evidence, and whether statistical testing is the optimal use of effort.

If a failure is detected during statistical testing, then, unlike in other forms of testing, it is challenging, or maybe impossible, to use impact analysis to understand how the defect fix affects the test outcome, i.e. the target confidence in reliability. As a result, after the fix a full suite re-run may be needed, or at least a significant expansion of the test-suite size. This is clearly a concern for the project schedule, because the testing should be performed on the final version and so is near the critical path. This creates friction with the safety justification — achieving the aim of demonstrating confidence will directly lead to operational delays. In addition, potentially unintuitive statistical justifications may be needed for any course of action other than full re-test, which is undesirable in a safety-related justification[79].

Any test technique that encounters a failure will likely require additional subsequent effort, but the distinguishing feature of statistical testing is that the design of the test suite is focussed on statistical distribution, not on, for example, functional or input-condition coverage. Therefore, if a test failure invalidates the statistical value of the suite, then the

[79] Responding to a statistical-testing failure is the subject of several papers (Littlewood and Wright 1995) (Littlewood and Wright 1997) (Korhonen et al. 1997) (Strigini and Littlewood 1997).

evidence generated by the suite until that point is likely to be less compelling than that which other techniques may have generated in the same time period. This is especially the case in relation to testing performed on pre-final versions (e.g. simulation testing of the application prior to hardware integration) — potentially far higher test throughput can be achieved at that point, i.e. at a time that is easier to fix any defects (and indeed the fixes are likely to be more thorough, as late-stage fixes often may be constrained so as to reduce late-stage impacts).

Kuball *et alia* (2004) state that their statistical testing campaign revealed failures that other forms of testing in their study did not, but concede that the consequence of this was that a reliability figure could not be claimed from the statistical testing until the defects were removed and the system successfully re-tested. This was despite the fact that the defects did not affect safety. Equation (1) is blind to defect severity.

The critical consideration is whether test time and effort should be spent performing statistical testing (especially considering all of the challenges that have been identified in this review paper) rather than other techniques that forgo a quantitative demonstration of a reliability target in favour of more targeted and justifiable testing.

The challenges associated with understanding test value when there has been a failure links to Section 4, where we consider what statistical testing tells us about the reliability of individual safety functions.

3.7 Other Implementation Challenges

The above subsections have covered several of the assumptions and constraints in Figure 9, but not all. It is likely that, as with many test systems, the test harness's injection of signals into the test object will be only a model of how real signals are injected onto the real system — for example, instead of a continuum of inputs, it is likely that signals will be injected in a regular series of discrete time steps, likely with each signal being injected synchronously. This reduces the representativeness of the stimuli and may mask temporal-based defects in the test object.

As already noted, test-suite execution length usually presents a challenge in statistical testing. In high-integrity validated systems, where statistical testing is typically used, faults are rarely detected, so there is little data available to indicate at what point in a test campaign this detection will occur: are faults first detected in the first few iterations of a particular scenario, or do most defects require such specific conditions that they are likely to need many tests to be found? If the former, this makes it harder to justify to programme managers why statistical testing should continue for 5,000 or 50,000 tests, when defects are likely to be found in the first few hundred tests[80].

The idea of uniformity of quality across the software under test (which we perceive to be an implicit assumption of statistical testing when performed at system- rather than function-level) may be challenged if, during the development lifecycle of the test system (including harness, test case generator, etc.), some trial tests are run (not necessarily with a statistical distribution) on the test object (possibly a pre-release version) using the test system. A test harness for a complex test object is challenging to build, and likely to have flaws in its interface with the test object, so this trial approach may be essential for a practicable test project. If a fault is detected in the test object during these trials, then the safe thing to do

[80] Indeed, to de-risk the programme, a prudent statistical test campaign would seek to run tests based on each scenario at the beginning to catch immediate failures, and only then start to do repeats (variations) of each scenario.

would be to fix it before the final version. But any such faults will make it even more important that the test operational profile is complete and representative, because now we have only improved the code quality in the area of code that our test profile will test. Ultimately, we are trialling our test system's correctness using the test object it is built to test. If we do a formal campaign and achieve 50,000 failure-free tests, is this because the test object is highly reliable, or is it because the trial tests drawn from the test operational profile have already detected all the faults that can be revealed by the test operational profile? This may qualitatively make sense (we have improved quality where we think it is likely to matter, as per Musa's (1993) approach) but it may make it harder to justify the validity of the quantitative calculation of α. It also places more importance on our test operational profile being complete compared to the real operational profile throughout the operational life.

In this Section, Bayesian inference techniques have already been touched on in a specific case (the discussion about Littlewood and Wright's (2007) result indicating that continuing to show test passes may call the oracle into question). A more recent paper (Salako and Zhao 2023) uses conservative Bayesian inference (CBI) to examine whether the i.i.d. criterion for sampled tests is necessary for a statistical testing result, and argues that it is not. However, while it proposes an approach that does not require i.i.d. sampling in a test suite, the high levels of confidence achievable if we did assume i.i.d. sampling are unobtainable in their conservative Bayesian approach. Salako and Zhao find that in fact unless the engineer has very good confidence $1 - \varphi_2$ that tests are not positively correlated (i.e. the engineer has confidence that the success of one test does not imply the success of the next; this correlation might indicate that for example the tests are not challenging enough, the oracle is not good enough, or that state is not reset between tests), then confidence will reach a peak during a campaign if all tests are passing, and then diminish with further successful tests. In the extreme case of low confidence in no correlation then running any tests immediately diminishes the confidence established from prior knowledge right from the start of the campaign (see Salako and Zhao's Figures 4e and 4f for $\varphi_2 = 0.3$; Figure 4e is reproduced overleaf as Figure 10, where *Beta(1,10,000)* represents a prior representing confidence from other sources (e.g. development quality, analysis, other test) of 10^{-4} pfd)[81].

On the face of it, Salako and Zhao's work seems to enhance the understanding of statistical testing in a pragmatic and conservative manner, providing a grounding for interpreting test results. However, we consider running a test campaign using this CBI approach:

1. The engineer designs a test system and tries to minimise φ_1 and φ_2 through good test-system design and implementation[82].
2. The engineer starts running the test suite, running 10,000 tests initially.
3. Zero test failures due to the test object are encountered (i.e. no false positives identified).

[81] Salako and Zhao (2023) Figure 4f uses a prior with lower prior confidence in the test object's pfd, showing that achievable confidence from this testing is even lower and drops off more sharply.

[82] φ_1 is the confidence that tests are negatively correlated.

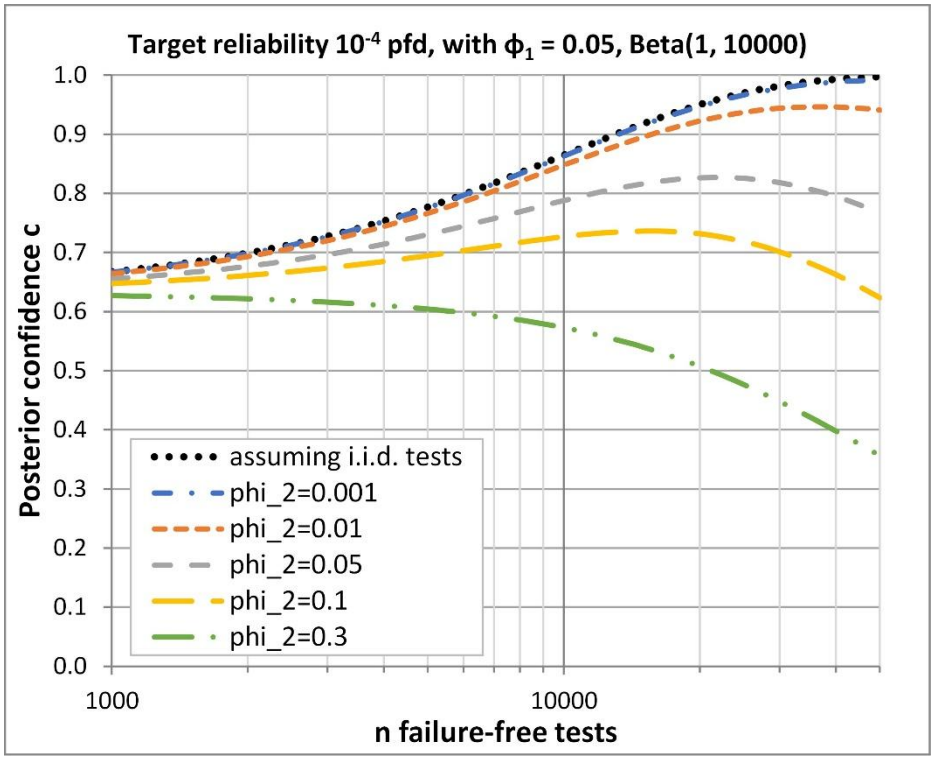


Figure 10 ~ As Confidence in Test Setup Diminishes (Higher Φ2), the Lower the Achievable Confidence in Test-Object pfd (recreated from Salako and Zhao (2023))

Using Figure 10, what confidence do we now have in the pfd? If we have high confidence that our test setup is good and tests are not being unjustifiably passed, then we should keep testing to increase confidence further. If, however, our confidence in our test setup is lower, any more tests may reduce our confidence in the pfd (that is, if our confidence has not already started reducing already).

With this CBI approach, we have gone from the traditional statistical testing approach of assuming our good test-setup design fully meets the core conditions and thus we use Equation (1), to a situation where we assume that our ability to quantitatively define our numerical confidence in properties of the outcome of the test campaign (i.e. test correlation) is strong. From all of the implementation challenges discussed already in our paper, it is unclear how the test engineer could determine whether their φ_2 was 0.001, 0.01, 0.1, 0.3 or any other value with enough certainty to be of use.

We argue that use of CBI does not solve the underlying challenge discussed throughout this paper. Even with the additional mathematics of CBI ,we are just using qualitative, subjective judgement to underpin quantitative parameters. The research by Salako and Zhao (and others) into CBI exposes mathematically (and graphically) the consequences of over-trusting a test setup, but does not provide a means of dealing with the test engineer's quantitative judgement being inaccurate. Are we trying to do what humans are notoriously bad at: estimating confidence and probabilities based on judgement alone?

One could adjust where the tipping point of the confidence curve is by adjusting the parameters φ_1 and φ_2, but this is what we tried to do at the start by doing good test design. After those 10,000 tests in our hypothetical test campaign, we are only now further questioning our test design because we have passed all the tests. If we do take time to improve the test design, and then continue testing, how do we quantify the confidence-building value of the first batch of tests?

To draw a positive note from the CBI work, one may argue that there is indeed a justification for performing smaller statistical-testing campaigns than is traditionally done — for the 10^{-4} pfd system in Salako and Zhao, the test engineer would be well-advised to stop after ~20,000 tests for peak confidence, rather than run 46,000 for a hypothetical 99% confidence. This would of course be dependent on being able to justify adequate numerical confidence in the test setup, which Section 3 has cast doubt upon.

3.8 Summary of Implementation Challenges

Section 3 has discussed many of the implementation challenges, both technical and programme-based, that face a project looking to use statistical testing to provide quantitative evidence of the reliability of a programmable system. None of the challenges are insurmountable individually, but tackling all of them is a significant endeavour and unlikely to be reasonably practicable to a level of rigour or certainty necessary to support quantification of α. Having to make compromises and judgements to overcome the challenges can lead to tricky qualitative arguments, and risks undermining the targeted quantitative outcome.

IAEA (2018) states that when dealing with high-reliability systems, the dominant uncertainty tends to come from epistemic rather than aleatory uncertainty, due to the high basic assumption of reliability of the system under assessment. This reflects in the described challenges of implementing statistical testing to meet the core conditions — we are limited in the accuracy of the test infrastructure we can produce when we are trying to demonstrate very low failure rates in the test object. Use of CBI (see Sub-section 3.6) illustrates IAEA's point quite clearly.

We conclude that in execution of statistical test campaigns it is easy to underestimate uncertainty, especially in complex systems like a reactor protection system. We see that oversights leading to overconfidence historically seen in other domains (such as in measurement of physical constants, as studied by Henrion and Fischhoff (1986)) may also be pervasive in software engineering.

4 System Complexity and the Safety Case

A picture of the types of systems and components that have been subjected to statistical testing emerges from a literature review. For the most part, the technique has been applied to simpler systems — devices of limited functionality (e.g. Kuball *et alia* (2004), Kuball (2024)), or low-demand/protection functionality subsets of larger programmable systems (e.g. Gough and Kuball (2014), US NRC (2017)). In the latter cases, the testing was limited to the functionality subset due to the challenges associated with statistical testing of high-demand/continuous systems.

When it comes to programmable systems, "complexity" is a disputed term: a system or component with only one primary function may be complex due to its support functions or simply its implementation, for example. Regardless, statistical testing theory is "blind" to the size or complexity of the test object by any metric — a smart device and a reactor

protection system will be subject to the same number of tests at a given confidence and reliability target[83].

A different perspective on complexity is to consider the size and variety of the operational profile. A simple trip amplifier, which changes its relay output according to a single analogue input's value compared to a set point (perhaps even including hysteresis), can be tested using a relatively small operational profile — there is very little dimensionality to the input space, and so within the statistical-testing framework there are few unique input scenarios around the point of safety-function demand[84]. For a reactor protection system, on the other hand, a whole library of distinct plant faults contributes to the operational profile, and each fault may have subtypes depending on the behaviour of dependent and independent plant variables.

Now, we consider what is claimed by the safety case.

It is convenient shorthand to attribute reliability (pfd) figures to a system, but the reality is that safety claims are made on the reliability of *safety functions*. These may be implemented in part or in whole by a system or component, but the reliability that the safety case seeks to demonstrate is the achieved reliability of the safety functions, not the components themselves.

Statistical testing within BS EN 61508-2:2010, for example, is used as a validation activity. Validation is used to show that the system meets "*in all respects the requirements for safety in terms of the required safety functions and safety integrity*", which means that the safety functions themselves shall be assessed, not the system specifically. Even where statistical testing is used as part of a system integration activity, this exercise is to demonstrate that the subsystems developed to implement the safety function "*interact correctly to perform their intended function*".

We select or build a system based on the determined reliability requirements of the functions allocated to it. The fault and protection schedule, and associated analysis, will focus on the likelihood of the *function* failing: the system is just part of the implementation of the function. Despite this, statistical testing always appears to be performed as if the system's reliability is homogeneous, and therefore claims to be a measure of the system reliability, not the *function* reliability[85]. This system-based approach perhaps originates from descriptions such as Musa (1993), where the objective was to use operational-profile testing to reduce the likelihood of the system encountering faults, rather than demonstrating the reliability of individual functions (that paper discusses how to best use test time to focus on the software functions most likely to be used in real life).

We will now consider a hypothetical system, with safety functionality delivered by three separate internal boards or processors. In our example, each safety function could plausibly be implemented by its own software on its own processor on its own internal board, all present within a common chassis of a system. This is an extreme version of a common arrangement in a reactor protection system, where the multiple safety functions assigned to the system are distributed over multiple discrete processing units. From the perspective of statistical testing this is a black box and the internal details of implementation do not matter.

[83] Sizewell B PPS contained around 100,000 lines of code (Littlewood et al. 1998), and Hinkley Point C's PS comprises even more lines (AtkinsRéalis 2020). By comparison, many smart devices (especially those developed for safety industries) are likely to be limited to thousands or perhaps a few tens of thousands of lines of code.

[84] Each test case must feature a demand on the safety function(s), but there are only so many distinct ways to approach a setpoint: perhaps by changing ramp rates, or introducing trajectory shapes in the build-up to the demand moment.

[85] Several studies suggest that defects are clustered rather than uniformly spread throughout the code. For example, see Hopkins and Hatton (2019), Boehm and Basili (2001) and Daniels and Tudor (2022).

We run a statistical testing suite for our hypothetical system, where boards #1 and #2 are tested in nearly every test case due to the frequency of the fault scenarios that demand those functions in the operational profile. Board #3's function is only called a small number of times, if at all, due to the statistical sampling. All tests pass, meaning we claim high confidence in the *system*'s reliability. For each safety function, the confidence in function reliability is very different, yet it is this reliability that we actually want to demonstrate. For the safety function implemented by board #3, we have almost zero statistical confidence in its reliability based on how much it was tested. For the safety functions implemented on boards #1 and #2, their confidence will also be slightly lower than the confidence in the overall system. The diminishing confidence per safety function becomes more challenging the more functions that the system implements (and potentially as more categories of scenarios are modelled).

In the fault and protection schedule, the fault's initiating event may have a low frequency, but the reliability of the safety function will always align with the system's overall pfd (e.g. 10^{-3} pfd). For a single-function smart device, there would be no problem, because evidence of the reliability of the safety function and system will be aligned. For a multi-function system like a protection system, to argue that a test suite featuring a handful of tests touching a particular function shows 99% confidence of 10^{-3} pfd *in that function* either: 1) assumes homogeneity of defects within the system (such that each fault scenario has equal chance of revealing a defect); or 2) double-counts the initiating event's own low likelihood. Neither of these arguments are sound: the first is not based on any conventional understanding of software defects, and the second fails to represent the numbers in the fault and protection schedule.

A hypothetical fault and protection schedule for a plant with a protection system with only two safety functions is shown in Table 13. The final column indicates what a $\alpha = 99\%$, $\theta = 10^{-3}$ pfd statistical testing campaign would tell us about the two functions. In reality, some initiating events may demand multiple safety functions, so the functions are tested in multiple scenarios; however most, if not all, functions will not be demanded by every initiating event.

Table 13 ~ Illustrative Fault and Protection Schedule

Initiating event	Initiating-event frequency / y^{-1}	First-line protection	Claimed first-line reliability / pfd	Number of statistical tests	Statistical confidence in first-line function
Fault 1	10^{-1}	Safety function A	10^{-3}	4,560	~99%
Fault 2	10^{-3}	Safety function B	10^{-3}	45	~4%

For context, if using a purely statistical sampling approach, Hinkley Point C PS statistical testing would use ~25,000 tests on the three most common fault scenarios, leaving 25,000 tests for the remaining ~147 scenarios, with the majority of those tests covering only the

next 21 most common scenarios (Walker 2024)[86]. To combat this, the project is reported to be looking to redistribute tests to increase coverage of lower-frequency scenarios without increasing the overall test-suite size (Walker 2024). If the targeted pfd is less onerous (e.g. 10^{-2} pfd) then coverage per function and scenario may be even thinner, reflected in the grading of technique effectiveness by safety integrity level in BS EN 61508-2:2010.

We have not identified any instances where the "enhanced model" binning method proposed by May *et alia* (1995), which is claimed to help address the issue of system complexity and coverage, has been used in a real statistical test campaign. By considering code coverage in the calculations, it makes the statistical-test result more conservative (i.e. more tests required to demonstrate the same pfd), so this will exacerbate the test-suite execution-time challenge. In terms of validity, it still seems to assume homogeneous defect distribution across the system, since the metrics are not sensitive to the aggregated code coverage, and that code coverage is an accurate proxy for state-space coverage. May *et alia* (1995) do not define what code-coverage metric would be used, but this may strongly affect the validity of the code-coverage to state-space-coverage relationship. Further, by binning based on code coverage, this further requires us to move into white-box territory, rather than the black-box territory supposedly targeted by statistical testing. May and Lunn (1995) describe the enhanced model in more detail, and acknowledge that more work is needed to account for code complexity (and hence distribution of sources of failure) in the enhanced model.

Finally, regarding the use of statistical testing as evidence to support the safety case, it is not clear how to reconcile the statistical testing approach with the fact that these systems can also be interacted with by reactor operators, a challenge discussed by Strigini and Littlewood (1997). The technique does not have a developed means of realistically modelling human interactions with the system[87], and thus any human-interface functionality is not considered as part of the modelled operational profile (socio-technical aspects are ignored). To some extent this is reasonable, because in a reactor fault the automatic protection functions are expected to negate the need for operator interaction; however, this assumes that the system is in a predictable state unfettered by human interaction at the start of and during a transient. For systems with demands where human interaction is likely, this becomes more significant. Any human interaction, whether concurrent or in advance, may ultimately affect the internal state of the system. To exclude human interactions in the estimation of reliability must implicitly assume that the system development lifecycle is equally good at implementing and verifying manual functions as it is at implementing and verifying automatic functions.

We conclude that it is not clear how to measure the value of the quantitative reliability estimate delivered by statistical testing when building a safety case for complex systems, and nor is it clear how to align the statistical findings with what the safety functions actually need. We conclude that to actually support the safety case's claims on the safety functions, it would be necessary to run a test suite *for each safety function*, each of the size given by Table 12. For a realistic protection system, this would likely be intractable due to the number of safety functions typically allocated to these system — both Sizewell B PPS and Hinkley

[86] Note that the material in Walker (2024) was presented at a publicly accessible IAEA conference, but is not in a readily accessible or peer-reviewed form. For this reason, these values should be taken as indicative/illustrative and should not be quoted without that qualification. Based on our experience of similar systems, we posit that these figures would follow similar patterns for the majority of statistical-testing campaigns.

[87] For example, Gough and Kuball (2014) describe how simplifications were needed to handle operator resetting alarms, which was acknowledged to not be fully realistic. That this relatively simple interaction was challenging highlights why a human-interaction model is often omitted.

Point C PS literature indicates that time-to-complete is a significant consideration, so even a doubling of test time would be unachievable[88].

5 What Can We Learn from the Statistical Testing Method?

Despite the concerns outlined so far in this paper regarding quantitative evidence, statistical testing practices can be useful in developing qualitative evidence of system quality. This seems inarguable, as statistical testing usually involves a campaign of tests that are more holistic than testing performed during the system development lifecycle.

Kuball *et alia* (2004) describe how their test campaign using statistical testing techniques found a defect that several other test techniques did not. This is positive, but we argue that this is not because of the mathematical statistical testing theory but rather due to the method of sampling, which was not a pure operational profile[89]. Lessons learned from how their test campaign was structured from a functional perspective (i.e. upon what types of scenarios they focussed) may be used to help build future campaigns based on operational profiles, even if the statistical testing theory, e.g. Equation (1), is not used to unilaterally dictate the size or distribution of the campaign. The work of Kuball *et alia* focussed tests on combinations of error conditions, rather than what statistical testing would usually do, which is to sample tests based purely on their frequency in the operational profile.

A different technique that may achieve similar results to Kuball *et alia* (2004) is combinatorial (t-way) testing (Kuhn 2011). Like the test campaign of Kuball *et alia*, it aims to determine combinations of conditions that may reveal defects. Techniques have been developed in the field of combinatorial testing to achieve an efficiently sized test suite that minimises over-coverage of combinations of conditions. Note that combinatorial testing, while providing a rigorous approach, does not provide a quantitative reliability measure.

As described by Kuball (2024), a statistical testing campaign for a spine-break detection system (claimed at 10^{-2} pfd) revealed discrepancies between the system's documentation and its behaviour with respect to its previously less-tested rate-of-change alarm. This demonstrated the benefit of detailed testing of a device's individual functions, and of not assuming that successful testing of function A implies equal quality in function B.

Kuball and May (2007) describe work in a statistical testing project to identify how to initialise the test object between tests to ensure non-interference between tests. This involved review of device documentation, and a series of tests to build an understanding of the length of the internal memory's history. This was a black-box approach, *inferring* the history length. For simpler devices, this inference may have a reasonable degree of confidence, but for larger systems the ease of building confidence will diminish. Again, the approach taken is not something that is exclusive to statistical testing, but it is a technique that could be useful for other forms of testing.

[88] While the actual increase in test time is inherently and strongly dependent on application context, we argue a doubling of suite size would be optimistic due to relation of functions (for example, a function protecting against high pressure in a vessel is unlikely to be demanded in the same test as a function protecting against low pressure in that vessel). That Tooley and Stokes (1992) reported that their set of scenarios only tested "*a majority*" of safety functions indicates that some functions are demanded in only specific circumstances.

[89] Their test suite was more challenging than a randomly sampled statistical testing suite because test cases were excluded if they had already been covered by earlier test phases. This was pragmatic, and meant that the statistical test suite could focus on the more onerous scenarios rather than just test the most common/less challenging scenarios. Exclusion of test cases from the statistical testing profile would require analysis to demonstrate exhaustiveness of existing testing.

We believe that, given the difficulties in providing enough quantitative confidence that assumptions are fulfilled and implementing the statistical-testing campaign to achieve this, the target for programmable-systems assurance should not be to determine a quantitative measure of reliability, but to build a qualitative confidence in the system and its implementation of its safety functionality. By avoiding a focus on the quantitative objective, this may open the door to more valuable testing that generates greater confidence-building value, perhaps even with less cost and effort (for example by altering the distribution and content of tests, or allowing simulation testing of emulated objects to contribute to the outcome).

6 Conclusion

We have argued that in the case of complex systems, such as reactor protection systems, it is not reasonably practicable to adequately achieve all the statistical testing core conditions to a sufficient level, so that the confidence value derived from the results would have supportable quantitative meaning[90]. Past attempts at using complex statistics to demonstrate that qualitative arguments do not invalidate the quantitative claims tend to be limited to hard-to-scale model cases and, even then, these attempts often rely on qualitative assumptions.

Our position does not appear to be unique, being reflected for example in the (non–system-specific) notes in nuclear C&I systems standard BS EN 61513 that "*there are no generally recognised means to quantify the reliability of software*" and in generic functional-safety standard BS EN 61508-3:2010 that "t*he probability of occurrence of systematic failures cannot in general be quantified*". A literature survey produced for the ONR (ATLAS Consortium 2023) assessed that "*no single method alone is considered sufficient to accurately estimate software reliability*".

We conclude that, ignoring challenges around test-case independence, the limit of what we can quantitatively claim from a traditionally executed statistical testing campaign of size N is the following:

> *The probability θ, to a statistical confidence α, that a system (identical to the test object in version, configuration and state of condition) with one safety function (or multiple decoupled safety functions that are demanded in every test) will, when subjected to test-harness stimulation (selected from the test distribution derived from the model of the real operational profile), respond in a manner that aligns with the predictions of the oracle's model of the system requirements.*

If statistical testing were performed in a closed loop (i.e. the outputs of the test object can influence subsequent simulation steps) and the pass/fail determination were based on whether the simulated plant is safe, then the final clause would be:

> *... will respond in a manner that results in the simulated plant reaching a safe state.*

Note that both open- and closed-loop testing conclusions above are still limited to the test environment — information about how they behave in the real environment can only be inferred based on understanding of the closeness of the two environments.

[90] A judgement specifically on the appropriateness of statistical testing for digital devices of limited functionality (as defined in BS IEC 62671:2013 and commonly referred to as smart devices) is beyond the scope of this paper.

Inadequacies in the fidelity of the operational profile, plant simulation, oracle, test harness, or representativeness of the test object are either unquantifiable or can only be approximated, so would only serve to reduce the confidence α by an unknowable amount when trying to translate this outcome to the real system. This is further complicated by considerations of system and plant ageing, maintenance (e.g. see Ladkin and Littlewood (2016)), modifications, and the effect of human interaction with the system and plant. As seen in the CBI approach in Littlewood and Wright (2007) and Salako and Zhao (2023), the confidence reduction associated with inaccurate assumptions can be stark and hard to quantitatively predict. In large multi-function systems, are we just over-fitting to an under-scoped inaccurate test operational profile, and in single-function devices are we just assuming that by running thousands of tests we would more likely find a defect in the device's safety function than a device defect being masked by an inadequate test oracle[91]?

We question why the target confidence α is seemingly always 95% or 99%[92], when, for UK nuclear at least, the reliability claim placed on the system is a 50% confidence "best-estimate" in the probabilistic safety analysis. The additional tests performed to take the statistical testing from 50% to 95% (or 99%) confidence provide diminishing levels of real value, and the dependence on the core conditions becomes increasingly strong. For a 10^{-4} pfd system, Table 12 showed that the difference between 50% and 99% confidence is 39,000 tests; however, many of these extra tests will be of the same few scenarios that would have already been tested quite extensively within the limits of the test operational profile's fidelity[93].

We also conclude that statistical testing is focussed on providing a measure of the implemented *system* reliability, yet the safety case (and indeed BS EN 61508) is primarily concerned with the reliability of each *safety function*. This distinction has not previously been discussed in statistical-testing literature to the best of our knowledge. Statistical testing therefore relies on flawed deductive reasoning to support the safety case, such as:

Premise 1 Statistical testing provides confidence α that the system reliability is at least as good as θ when subjected to its operational profile.

Premise 2 The system implements one or more safety functions.

Conclusion We have confidence α that each safety function implemented by the system has reliability at least as good as θ when subjected to the system's operational profile.

The first premise means that there the probability that the "next" simulated plant fault will reveal a system failure does not exceed θ. We consider Fault X: our operational profile tells us that Fault X will happen at a frequency of 10^{-x} y^{-1}. The safety case is interested in whether the protection system will act correctly when Fault X happens. But, if Fault X is infrequent, then its associated safety functions have received minimal testing, so we have minimal confidence on that safety function's reliability. The above premises can only be used to reach the conclusion if we assume that software-based systems only fail randomly, not systematically, or that the probability of a rarely used function failing *on demand* is the same

[91] This sentence swings both ways — the two questions could apply to both types of test object.

[92] Indeed, NS-TAST-GD-046 recommends this target.

[93] Taking the illustrative numbers in Walker (2024), a 50% confidence suite might test the most common scenario over 2,000 times. Do we get more confidence by increasing the suite to 99% confidence, which will include running another ~13,000 tests of that scenario? A single omission from the operational profile's description of that scenario, e.g. a human interaction with the system, or an unmodelled fault, or misconfiguration, could completely change the system behaviour. Further, as noted by Salako and Zhao (2023), more tests may reduce confidence if there is any doubt in the test setup.

as a frequently used function[94]. Otherwise, we suggest that the statistical-testing suite needs to be big enough that each safety function is demanded by *N* tests, rather than *N* tests for the system as a whole.

Musa (1993) provides figures extolling the commercial benefit of operational-profile testing in commercial software development, including as part of the development lifecycle. The numbers are compelling, yet it is hard to find similar evidence for statistical testing of *high-integrity* programmable systems, especially when they have already completed validation. From all of the literature studied, Kuball *et alia* (2004) declare three (non-safety-related) defects identified by statistical testing for a safety interlock system, and other studies noted by Kuball (2024) refer to discrepancies from user documentation, or lack of detail in documentation, leading to configuration oversights detected by the test campaigns in commercial off-the-shelf products. While Kuball *et alia* (2004) argue the secondary benefit of statistical testing finding the other defects more efficiently than the other techniques in their study, it is not clear that statistical testing would have been *the* most efficient way to achieve the result. For example, combinatorial testing covering arrays can be calculated to find the most efficient way of covering variable interactions and, alternatively, more efficient defect-finding might be achieved by the "*enhanced functional testing*" described in Kuball (2024), akin to "*expanded functional testing*" as described in BS EN 61508-7:2010.

We conclude that in practice, statistical testing should not be pursued as a technique for *quantitatively* demonstrating system reliability. If we remove the objective of a quantitative result, we lose several constraints, and so open the door for extracting more value out of testing, such as use of simulators/emulators to increase test throughput, use of instrumentation to measure coverage, and more efficient exploration of state space. Walker (2024) indicates that such movement towards qualitative test benefit is already being considered in large real-life projects.

We argue that this proposition for forgoing statistical testing does not necessarily require a direct replacement for the technique (i.e. another approach to quantitatively demonstrating reliability of a software-based system), because our assertion is that, for the majority of systems (especially many-function systems), quantitative demonstration of reliability is not in fact achieved with statistical testing today anyway, for the reasons outlined in this paper.

All is not lost, however, because without the constraints required for quantitative analysis, there may time and resources available to extract a benefit from performing other techniques such as operational-profile testing prior to C&I system validation to demonstrate behavioural properties. This would be for the purposes of finding and fixing defects rather than developing numerical perceived confidence in the numerical confidence in reliability. Such pragmatic techniques can serve to actually increase the quality of the system and work on minimising systematic defects, and then eventually to provide qualitative confidence that operational use will not reveal further defects, building on the work of Musa (1993).

7 Further Work

Some of the principles of test infrastructure and test suite design developed to support statistical testing over the years are useful, and there is some evidence to suggest they have benefits over other forms of testing. However, it is proposed that they should be used with

[94] We argue that it is not valid to assume that all functions are equally defective (or undefective) due to the concept of defect clustering.

the objective of targeting valuable and justifiable qualitative evidence of the reliability of safety functions, complementing other forms of quantitative evidence of the functional correctness of the system and its functions.

As well as considering the value that pre-validation operational-profile testing of software may offer, research into other techniques such as combinatorial testing (Kuhn 2011) (Diemert et al. 2023) or expanded functional testing (BS EN 61508 7:2010) may also supplement this work and inform testing approach.

This paper has focussed on larger, complex systems like reactor protection systems. Smaller, simpler systems or components, such as digital devices of limited functionality (smart devices), may not be as sensitive to some of the arguments made in this paper, and so may benefit more from a "purer" statistical-testing approach. Further work is needed to determine whether statistical testing has value for such devices, or whether operational-profile testing, combinatorial testing, or another option are more appropriate. It is noted that overall safety arguments for and against statistical testing of smart devices may have different structural considerations compared to larger systems and a holistic approach to the smart devices' safety justification would be needed. We ask what "reliability" for a smart device when integrated into a system means. How does one determine the pfd required for a smart device within a system — if the safety function reliability is claimed as 10^{-2} pfd, then what would be the smart device's "reliability" in its integrated environment — presumably better than 10^{-2} pfd unless the rest of the system is flawless? What benefit is there of determining the "reliability" of a smart device, if the safety function to which it contributes also requires the action of the wider system in which the smart device is integrated? Failure of the smart device to respond to a demand may not prevent the safety function being performed, and conversely, the safety function may still not be achieved by the system even if the smart device behaved according to its specification.

Disclaimers

Opinions expressed herein are those of the authors, not necessarily those of their past and present employers or clients. The authors note, however, that our past and present employers and clients may materially benefit if the conclusions of this paper are more widely accepted.

Acknowledgments

The authors would like to thank Dan Finnigan (Rolls-Royce SMR), Lee Walker (Rolls-Royce SMR), and Waleed Chaudhry (Holtec Britain) for their comments on an early draft of this article.

The authors would also like to express their gratitude to the anonymous peer reviewers who provided supportive, thoughtful, and constructive commentary on this article, improving its robustness, accuracy and value.

References

Arthur C. (1993). *Extra tests forced on Sizewell's safety software*. New Scientist, 6th November 1993. Available from https://www.newscientist.com/article/mg14018980-600-extra-tests-forced-on-sizewells-safety-software/. Accessed 20th December 2025.

ASCA. (2007). *Risk-informed safety assurance and probabilistic risk assessment of mission-critical software-intensive systems*. ASCA Inc., Report AR 07-01. Available from

https://ntrs.nasa.gov/api/citations/20100006923/downloads/20100006923.pdf. Accessed 16th September 2026.

AtkinsRéalis. (2020). *Atkins to verify the safety of reactor protection software at Hinkley Point C*. Trade Release of 23rd April 2020. https://www.atkinsrealis.com/en/media/trade-releases/2020/2020-04-23. Accessed 12th February 2026.

ATLAS Consortium. (2023). *Provision of technical support to the consideration of realistic software modelling in PSA*. ATLAS Consortium, 3rd March 2023. Available from https://www.onr.org.uk/publications/regulatory-reports/research/research-reports/onr-rrr-112. Accessed 13th February 2026.

Bishop P, and Povyakalo A. (2017). *Deriving a frequentist conservative confidence bound for probability of failure per demand for systems with different operational and test profiles*. Reliability Engineering & System Safety 158:246-253.

Boehm B, and Basili V. R. (2001). *Software Defect Reduction Top 10 List*. IEEE Computer, 34(1):135-137.

BS EN 60880. (2009). *Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions*. BS EN 60880:2009. British Standards Institution, London.

BS EN 61508-2. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems. Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems*. BS EN 61508-2:2010. British Standards Institution, London.

BS EN 61508-3. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems Part 3: Software requirements*. BS EN 61508-3:2010. British Standards Institution, London.

BS EN 61508-4. (2010). *Functional safety of electrical/electronic/ programmable electronic safety related systems. Part 4: Definitions and abbreviations*. BS EN 61508-4:2010. British Standards Institution, London.

BS EN 61508-7. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems Part 7: Overview of techniques and measures*. BS EN 61508-7:2010. British Standards Institution, London.

BS EN 61513. (2013). *Nuclear power plants – Instrumentation and control important to safety – General requirements for systems*. BS EN 61513:2013. British Standards Institution, London.

BS IEC 62671. (2013). *Nuclear power plants – Instrumentation and control important to safety – Selection and use of industrial digital devices of limited functionality*. BS IEC 62671:2013. British Standards Institution, London.

Butler R. W, and Finelli G. B. (1993). *The infeasibility of quantifying the reliability of life-critical real-time software*. IEEE Transactions on Software Engineering 19(1):3-12.

Coolen F. (2012). *On some statistical aspects of software testing and reliability*. Advances in Intelligent and Soft Computing 170:103-113.

Daniels D, and Tudor N. (2022). S*oftware Reliability and the Misuse of Statistics*. Safety-Critical Systems eJournal. Vol.1, Iss. 1. Available from https://scsc.uk/r1518. Accessed 16th September 2026.

Diemert S, Casey A, and Robertson J. (2023). *Challenging Autonomy with Combinatorial Testing*. In: *2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 214-222. Dublin, Ireland.

Dolan B. (1994). *Software testing at Sizewell*. The RISKS Digest, Volume 15 Issue 58, 23rd February 1994. Available from https://catless.ncl.ac.uk/Risks/15/58#subj5.1. Accessed 20th December 2025.

EN 50129. (2003). *Railway applications — Communication, signalling and processing systems — Safety related electronic systems for signalling.* EN 50129:2003, CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EN 50716. (2023). *Railway Applications — Requirements for software development*. EN 50716, 1st Edition, 2023. CENELEC (European Committee for Electrotechnical Standardization), Brussels.

EPRI. (2008). *Operating experience insights on common-cause failures in digital instrumentation and control systems*. Electric Power Research Institute, Inc. Available from https://www.epri.com/research/products/1016731. Accessed 1st July 2026.

EPRI. (2010). *Estimating failure rates in highly reliable digital systems*. Electric Power Research Institute, Inc. Available from https://www.epri.com/research/products/000000000001021077. Accessed 1st July 2026.

Gough H, and Kuball S. (2014). *Application of Statistical Testing to the Data Processing and Control System for the Dungeness B Nuclear Power Plant (Practical Experience Report).* In Proceedings of the 10th European Dependable Computing Conference, EDCC 2014. pp158-165.

Hansard. (1993). *House of Commons debate*. In: Commons Chamber Volume 233. November to December 1993.

Henrion M, and Fischhoff B. (1986). *Assessing Uncertainty in Physical Constants*. American Journal of Physics 54(9):791-798.

Hopkins T, and Hatton L. (2019). *Defect patterns and software metric correlations in a mature ubiquitous system*. arXiv 6th December 2019. Available from https://arxiv.org/abs/1912.04014. Accessed 16th February 2026.

Horgan J, and Mathur A. (1996). *Chapter 13: Software Testing and Reliability*. In: Lyu M. R. (Editor), *Handbook of Software Reliability Engineering*. IEEE Computer Society Press and McGraw-Hill Book Company, New York, USA. pp. 531-566.

Hunns D. M, and Wainwright N. (1992). *Software-based protection for Sizewell B: The regulator's perspective*. In: International Conference on Electrical and Control Aspects of the Sizewell B PWR, (IEE Number no.361), Institution of Electrical Engineers, British Nuclear Energy Society, Institute of Energy, Institution of Mechanical Engineers, and Convention of National Societies of Electrical Engineers of Western Europe (EUREL), London.

IAEA. (1999). *Verification and validation of software related to nuclear power plant instrumentation and control*. International Atomic Energy Agency, Technical Reports Series No. 384. https://www.iaea.org/publications/5718/verification-and-validation-of-software-related-to-nuclear-power-plant-instrumentation-and-control. Accessed 16th February 2026.

IAEA. (2018). *Dependability assessment of software for safety instrumentation and control systems at nuclear power plants*. International Atomic Energy Agency, Nuclear Energy Series No. NP-T-3.27. https://www.iaea.org/publications/12232/dependability-assessment-of-software-for-safety-instrumentation-and-control-systems-at-nuclear-power-plants. Accessed 11th February 2026.

Ichiyen N. M. (1998). *Software in Safety Applications*. International Symposium on Evolutionary Water Cooled Reactors: Strategic Issues, Technologies, and Economic Viability. Available from https://inis.iaea.org/records/3tfe8-def24/preview/30020332.pdf. Accessed 11th February 2026.

IEC61508. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems*. IEC 61508, in 7 parts, 2nd Edition, 2010. International Electrotechnical Commission, Geneva.

IEEE 352. (2016).: *IEEE Guide for General Principles of Reliability Analysis of Nuclear Power Generating Station Systems and Other Nuclear Facilities*. IEEE 352-2016. Institute of Electrical and Electronics Engineers, New York.

IEEE 1633. (2016). *IEEE Recommended Practice on Software Reliability*. IEEE 1633-2016. Institute of Electrical and Electronics Engineers, New York.

IRSN. (2017). *Principles relating to the digital instrumentation and control design approach*. Institut de Radioprotection et de Sûreté Nucléaire. Available from https://research-assessment.asnr.fr/sites/en/files/2023-12/IRSN_Safety-approaches_digital-instrumentation-and-control_2017.pdf. Accessed 1st July 2026.

ISO 26262. (2018). *Road vehicles — Functional safety*. ISO 26262:2018, in 12 parts, 2nd Edition, 2018. International Organization for Standards, Geneva.

Knight J, and Leveson N. (1986). A*n experimental evaluation of the assumption of independence in multiversion programming*. IEEE Transactions on Software Engineering SE-12(1).

Korhonen J, Pulkkinen U, and Haapanen P. (1997). *Statistical reliability assessment of software-based systems*. Finnish Centre for Radiation and Nuclear Safety, Report number STUK-YTO-TR--119.

Kuball S. (2024). *Black box testing for functional safety*. T6A Symposium 2024 — Guidance in Functional Safety Compliance. The 61508 Association. Available from https://61508.org/wp-content/uploads/2024/11/08A-T6A-Event-2024-A08-SKuball.pdf. Accessed 16th September 2026.

Kuball S, Hughes G, May J, Gallardo J, and John A. (2004). *The effectiveness of statistical testing when applied to logic systems*. Safety Science 42:369-383.

Kuball S, and May J. (2007). *A discussion of statistical testing on a safety-related application*. In: Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability. 2007;221(2):121-132.

Kuhn R. (2011). *Introduction to combinatorial testing*. National Institute of Standards and Technology, Presentation at Carnegie-Mellon University, 7th June 2011. Available from https://mse.s3d.cmu.edu/facstaff/faculty1/faculty-publications/miranda/kuhnintroductioncombinatorialtesting.pdf. Accessed 18th December 2025.

Ladkin P. B, and Littlewood B. (2016). *Practical statistical evaluation of critical software*. Paper presented at the 24th Safety-Critical Systems Symposium, Brighton, UK. February 2016. Available from https://rvsbi.de/publications/Papers/LadkinLittlewood-Practical-FinalV2.pdf. Accessed 16th February 2026.

Leveson N. (2004). *A new accident model for engineering safer systems*. Safety Science 42:237-270.

Littlewood B, Bainbridge I, and Bloomfield R. (1998). *The use of computers in safety-critical applications*. Health and Safety Commission, London, UK.

Littlewood B, and Strigini L. (1993). *Validation of ultrahigh dependability for software-based systems*. Communications of the ACM 36(11):69-80. Available from https://openaccess.city.ac.uk/id/eprint/1251/1/CACMnov93.pdf. Accessed 16th September 2026.

Littlewood B, and Wright D. (1995). *Stopping Rules for the Operational Testing of Safety-Critical Software*. Digest of Papers, Twenty-Fifth International Symposium on Fault-Tolerant Computing. Pasadena, USA.

Littlewood B, and Wright D. (1997). *Some conservative stopping rules for the operational testing of safety-critical software*. IEEE Transactions on Software Engineering 23(11):673-683.

Littlewood B, and Wright D. (2007). *The use of multilegged arguments to increase confidence in safety claims for software-based systems: A study based on a BBN analysis of an idealized example*. IEEE Transactions on Software Engineering 33(5):347-365.

Lyu M. R. (Ed.) (1996). *Handbook of software reliability engineering*. IEEE Computer Society Press and McGraw-Hill Book Company, New York, USA . Available from https://www.cse.cuhk.edu.hk/~lyu/book/reliability/. Accessed 1st July 2026.

May J. H. R. (2017). *Theoretical feasibility of statistical assurance of programmable systems based on simulation tests*. In: 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, Czech Republic.

May J. H. R, Hughes G, and Lunn A. D. (1995). *Reliability estimation from appropriate testing of plant protection software*. Software Engineering Journal, November 1995:206-218.

May J. H. R, and Lunn A. D. (1995). *A model of code sharing for estimating software failure on demand probabilities*. IEEE Transactions on Software Engineering 21(9):747-753.

Miranda B, and Bertolino A. (2017). *An assessment of operational coverage as both an adequacy and a selection criterion for operational profile based testing*. Software Quality Journal 26:1571-1594.

Musa J. (1993). *Operational profiles in software-reliability engineering*. IEEE Software 10(2):14-32.

ONR. (2011a). *Generic Design Assessment - New Reactor Build: Step 4 Control and Instrumentation Assessment of the Westinghouse AP1000® Reactor*. Office for Nuclear Regulation, Assessment Report ONR-GDA-AR-11-006.

ONR. (2011b). *UK EPR GDA Project – Resolution Plan for GI-UKEPR-CI-02*. Office for Nuclear Regulation, Resolution Plan GI-UKEPR-CI-02-RP. Available from

https://www.onr.org.uk/media/2mhmjqly/resolution-plan-gi-ukepr-ci-02.pdf. Accessed 16th September 2026.

ONR. (2012). *New nuclear reactors: Generic Design Assessment - Electricité de France SA and AREVA NP SAS UK EPR™ nuclear reactor - Summary of the GDA Issue close-out assessment of the Electricité de France SA and AREVA NP SAS UK EPR™ nuclear reactor*. Office for Nuclear Regulation, Assessment Report ONR-GDA-SR-12-001.

ONR. (2017). *New Reactor Division: Step 4 Assessment of Control and Instrumentation for the UK Advanced Boiling Water Reactor*. Office for Nuclear Regulation, Assessment Report ONR-NR-AR-17-017.

ONR. (2020). *Independent Confidence Building Measures for complex control and instrumentation systems*. Office for Nuclear Regulation, Regulatory Observation RO-UKHPR1000-0057.

ONR. (2023). *Computer Based Safety Systems*. Office for Nuclear Regulation. Technical Assessment Guide NS-TAST-GD-046. Available from https://www.onr.org.uk/media/3def2aat/ns-tast-gd-046.docx. Accessed 16th September 2026.

ONR. (2024). *Licensing of safety critical software for nuclear reactors – common position of international nuclear regulators and authorised technical support organisations*. Regulator Task Force on Safety Critical Software (TF SCS) Report. Available from https://www.onr.org.uk/publications/regulatory-reports/other-reports/licensing-of-safety-critical-software-for-nuclear-reactors. Accessed 25th February 2026.

Ontario Power Generation (2013). *Standard for Software Engineering of Safety Critical Software*. Ontario Power Generation Inc., CANDU (Computer Systems Engineering Centre of Excellence), CE-1001-STD Revision 3.

Prokop L. (2024). *Historical aerospace software errors categorized to influence fault tolerance*. In: IEEE Aerospace Conference, Big Sky, USA.

RTCA/DO-178C. (2012). *Software Considerations in Airborne Systems and Equipment Certification*. RTCA Inc., Washington DC.

SAE ARP4754A. (1996). *Recommended Practice, Guidelines for Development of Civil Aircraft and Systems*. ARP4754A. SAE International, Warrendale, PA. Available at https://doi.org/10.4271/ARP4754A. Accessed 13th July 2026.

Salako K, and Zhao X. (2023). *The Unnecessity of Assuming Statistically Independent Tests in Bayesian Software Reliability Assessments*. IEEE Transactions on Software Engineering 49(4):2829-2838.

Strigini L, and Littlewood B. (1997). G*uidelines for Statistical Testing*. ESA/ESTEC, PASCON/WO6-CCN2/TN12. Available from https://openaccess.city.ac.uk/id/eprint/254/. Accessed 1st July 2026.

Tooley P. A, and Stokes P. (1992). *Dynamic testing of the Primary Protection System*. In: International Conference on Electrical and Control Aspects of the Sizewell B PWR, (IEE Number no.361), Institution of Electrical Engineers, British Nuclear Energy Society, Institute of Energy, Institution of Mechanical Engineers, and Convention of National Societies of Electrical Engineers of Western Europe (EUREL), London.

UK HSE. (2003). *Out of control: Why control systems go wrong and how to prevent failure*. United Kingdom Health & Safety Executive. Health & Safety Guidance, HSG238. Available from https://www.hse.gov.uk/pubns/books/hsg238.htm. Accessed 1st July 2026.

US NRC. (2017). *Development of a statistical testing approach for quantifying safety-related digital system on demand failure probability*. United States Nuclear Regulatory Commission. Available from https://www.nrc.gov/docs/ML1715/ML17151A215.pdf. Accessed 1st July 2026.

US NRC. (2022). *US Nuclear Regulatory Commission's approach to software reliability including consideration of common-cause failures*. Technical Meeting on the Software Reliability of Digital Instrumentation and Control Systems for Nuclear Power Plant Safety, 13th—16th December 2022. International Atomic Energy Agency, Vienna.

US NRC. (2026). *(Availability of) An International Report on Safety Critical Software for Nuclear Reactors by the Regulator Task Force on Safety Critical Software (TF-SCS)*. United States Nuclear Regulatory Commission. International Agreement Report NUREG/IA-0463. Available from https://www.govinfo.gov/app/details/GOVPUB-Y3_N88-PURL-gpo67557. Accessed 12th August 2026.

Walker L. (2024). *Case studies in operational profile based statistical testing for the quantitative reliability estimation of software based commercial off the shelf (COTS) equipment*. Technical Meeting on Commercial Equipment in Nuclear Applications and Non-Nuclear Standards for Safety Instrumentation and Control Systems, 1st—3rd July 2024. International Atomic Energy Agency, Vienna.

About the Safety-Critical Systems eJournal

Purpose and Scope

This is the Journal of the Safety-Critical Systems Club CIC (SCSC), ISSN 2754-1118 (Online), ISSN 2753-6599 (Print). Its mission is to publish high-quality, peer-reviewed articles on the subject of systems safety.

When we talk of systems, we mean not only the platforms, but also the people and their procedures that make up the whole. Systems Safety addresses those systems, their components, and the services they are used to provide. This is not a narrow view of system safety, our scope is wide and also includes safety-related topics such as resilience, security, public health and environmental impact.

Background

When the Safety-Critical Systems Club (SCSC) was set up over thirty years ago, its objectives were to raise awareness of safety issues and to facilitate safety technology transfer. To achieve these objectives, the club organised events, such as Seminars and an annual Symposium, and also published a newsletter, Safety Systems, three times a year.

The Newsletter, in addition to news, opinion, correspondence, book reviews, and the like, also carries articles discussing current and emerging practices and standards. The length of such articles is limited to about two and a half thousand words, which does not allow an in-depth treatment. It was therefore decided to add a third string to our bow and supplement the events and newsletter with a journal containing longer papers. That journal is published here, as the Safety-Critical Systems eJournal, and comprises at least two issues a year.

Content Sources

Sources include the outputs of SCSC working groups; solicited technical articles and topic reviews; submitted articles on new analysis techniques, discussion of standards, and industrial practice; and guidelines and lessons learned. If you wish to contribute, please see, "Information for Authors".

Types of paper include, but are not limited to:

Technical Articles: Written by practitioners and describing practical safety engineering and assurance techniques, and their industrial applications.

Integration Studies: Written by practitioners reporting upon successful (or otherwise) synergies achieved in practice with other assurance domains, e.g. systems engineering, reliability/availability/maintainability engineering, resilience, human factors, security, and environment.

Position Papers: Written by, or on behalf of, Regulators, Standardisation Organisations, or other official bodies, setting out their position on a topic, e.g. the interpretation of a particular standard or regulation.

Review Articles: Papers highlighting recent developments and trends in some aspect of safety-critical systems or of their use in a particular industrial sector.

Historical Articles: Papers describing the development of safety assurance in an industrial sector; how we got to where we are today.

Perspectives: The authors' personal opinions on a subject, e.g. whether to use statistical methods in particular scenarios.

Reports: The lessons learned from incidents or the outcomes of trials with a description of scenarios, or methods, and a discussion of the results obtained.

Tutorials: Material prepared in support of tutorials to be presented at Safety-critical Systems Symposia. (These are not peer-reviewed.)

Working Group Outputs: Written by, or on behalf of, Safety-Critical Systems Club Working Groups to include discussions, underpinning theory, or guidelines.

Copyright and Disclaimer

The author(s) of each paper shall retain copyright in their work but give the Safety-Critical Systems Club permission to publish in both on-line and printed formats. While the authors and the publishers have used reasonable endeavours to ensure that the information and guidance given in this work is correct, all parties must rely on their own skill and judgement when making use of this work and obtain professional or specialist advice before taking, or refraining from, any action on the basis of the content of this work.

Neither the authors nor the publishers make any representations or warranties of any kind, express or implied, about the completeness, accuracy, reliability, suitability or availability with respect to such information and guidance for any purpose, and they will not be liable for any loss or damage including without limitation, indirect or consequential loss or damage, or any loss or damage whatsoever (including as a result of negligence) arising out of, or in connection with, the use of this work. The views and opinions expressed in this publication are those of the authors and do not necessarily reflect those of their employers, the Safety-Critical Systems Club, or other organisations.

Letters to the Editor

The editorial to the first issue of this journal said, "*You may find some of this material controversial, or you may think that it does not go far enough. Subsequent issues of this journal will have provision for readers' letters to the Editor responding to individual papers.*" Such a letter should be no more than 1000 words in length (not counting title, attribution, or references). That would take up no more than two pages of the journal. Note that a letter should ideally address a single concern with few, if any, external references.