The Safety of Systems

Related titles:

Towards System Safety
Proceedings of the Seventh Safety-critical Systems Symposium, Huntingdon, UK, 1999
Redmill and Anderson (Eds)
1-85233-064-3

Lessons in System Safety
Proceedings of the Eighth Safety-critical Systems Symposium, Southampton, UK, 2000
Redmill and Anderson (Eds)
1-85233-249-2

Aspects of Safety Management
Proceedings of the Ninth Safety-critical Systems Symposium, Bristol, UK, 2001
Redmill and Anderson (Eds)
1-85233-411-8

Components of System Safety
Proceedings of the Tenth Safety-critical Systems Symposium, Southampton, UK, 2002
Redmill and Anderson (Eds)
1-85233-561-0

Current Issues in Safety-critical Systems
Proceedings of the Eleventh Safety-critical Systems Symposium, Bristol, UK, 2003
Redmill and Anderson (Eds)
1-85233-696-X

Practical Elements of Safety
Proceedings of the Twelfth Safety-critical Systems Symposium, Birmingham, UK, 2004
Redmill and Anderson (Eds)
1-85233-800-8

Constituents of Modern System-safety Thinking
Proceedings of the Thirteenth Safety-critical Systems Symposium, Southampton, UK, 2005
Redmill and Anderson (Eds)
1-85233-952-7

Developments in Risk-based Approaches to Safety
Proceedings of the Fourteenth Safety-critical Systems Symposium, Bristol, UK, 2006
Redmill and Anderson (Eds)
1-84628-333-7

Felix Redmill and Tom Anderson (Eds)

The Safety of Systems

Proceedings of the Fifteenth Safety-critical Systems Symposium, Bristol, UK, 13-15 February 2007

Safety-Critical
Systems Club

BAE SYSTEMS

Springer

Felix Redmill
Redmill Consultancy, 22 Onslow Gardens, London, N10 3JU

Tom Anderson
Centre for Software Reliability, University of Newcastle,
Newcastle upon Tyne, NE1 7RU

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN-10: 1-84628-805-3 Printed on acid-free paper
ISBN-13: 978-1-84628-805-0

The use of registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant laws and regulations and therefore free for general use.

The publisher makes no representation, express or implied, with regard to the accuracy of the information contained in this book and cannot accept any legal responsibility or liability for any errors or omissions that may be made.

9 8 7 6 5 4 3 2 1

Springer Science+Business Media
springer.com

PREFACE

Since 1993 the Safety-Critical Systems Club has hosted the Safety-critical Systems Symposium (SSS) each February. Every year the programme has reflected what is then of particular interest to the safety community – in ways of working, in analysis techniques, in technology, in standards, and in research work that is on the point of being useful to practitioners. This book presents the papers delivered at the landmark fifteenth SSS.

A subject that is mostly neglected by safety practitioners but which, if studied more carefully, could lead to both technical and economic efficiencies, as well as more effective protection, is the relationship between safety and security. The management of both is based on risk analysis and there are indications that the analyses could effectively be combined. The Symposium has offered papers on this subject in the past, and this year there are three.

Continuing the trend of recent years, there are five papers on the development of safety cases, which are intended to demonstrate, or claim, the achievement of safety – in defined applications and under given circumstances. Some of the papers report on experiences in the field, but others venture to propose new ways in which safety cases may be used or extended.

Other areas of the safety domain whose importance is increasingly being recognised are safety management and safety assessment, and both are represented by three-paper sessions. One of the assessment papers is on human reliability assessment, a topic with which engineers are advised to familiarise themselves to a greater extent than hitherto. Indeed, a useful starting point is this paper, for it offers some historic background as well as a review of the techniques in the field.

The final section at the Symposium, and of this book, is on the use of 'formal' methods in achieving and demonstrating safety. Such methods have in the past been deemed to be expensive and only worth using in extreme circumstances, but many now claim that their proper use achieves such advantages that they should be employed as a matter of course. The papers here are on specification and the C language.

Whereas this book represents the content of the Symposium, it can only hint at the effort that goes into the event's organisation. We thank the papers' authors and their organisations for writing and presenting their papers and, thus, for contributing to a substantial Proceedings. We also thank Joan Atkinson for her continued indefatigable efforts in arranging the event's logistics, upon which everything else depends. And we look forward to SSS '08, planning for which has already commenced.

FR & TA
November 2006

THE SAFETY-CRITICAL SYSTEMS CLUB
organiser of the
Safety-critical Systems Symposium

What is the Club?

The Safety-Critical Systems Club exists to raise awareness of safety issues in the field of safety-critical systems and to facilitate the transfer of safety technology from wherever it exists. It is an independent, non-profit organisation that co-operates with all bodies involved with safety-critical systems.

History

The Club was inaugurated in 1991 under the sponsorship of the UK's Department of Trade and Industry (DTI) and the Engineering and Physical Sciences Research Council (EPSRC). Its secretariat is at the Centre for Software Reliability (CSR) in the University of Newcastle upon Tyne, and its Co-ordinator is Felix Redmill of Redmill Consultancy.

Since 1994 the Club has been self-sufficient, but it retains the active support of the DTI and EPSRC, as well as that of the Health and Safety Executive, the Institution of Engineering and Technology, and the British Computer Society. All of these bodies are represented on the Club's Steering Group.

The Club's activities

The Club achieves its goals of awareness-raising and technology transfer by focusing on current and emerging practices in safety engineering, software engineering, and standards that relate to safety in processes and products. Its activities include:

- Running the annual Safety-critical Systems Symposium each February (the first was in 1993), with Proceedings published by Springer-Verlag;
- Organising a number of 1- and 2-day seminars each year;
- Providing tutorials on relevant subjects;
- Publishing a newsletter, *Safety Systems*, three times annually (since 1991), in January, May and September.

Education and communication

The Club brings together technical and managerial personnel within all sectors of the safety-critical community. Its events provide education and training in principles and techniques, and it facilitates the dissemination of lessons within and between industry sectors. It promotes an interdisciplinary approach to safety engineering and management and provides a forum for experienced practitioners to meet each other and for the exposure of newcomers to the safety-critical systems industry.

Focus of research

The Club facilitates communication among researchers, the transfer of technology from researchers to users, feedback from users, and the communication of experience between users. It provides a meeting point for industry and academia, a forum for the presentation of the results of relevant projects, and a means of learning and keeping up-to-date in the field.

The Club thus helps to achieve more effective research, a more rapid and effective transfer and use of technology, the identification of best practice, the definition of requirements for education and training, and the dissemination of information. Importantly, it does this within a 'club' atmosphere rather than a commercial environment.

Membership

Members pay a reduced fee (well below the commercial level) for events and receive the newsletter and other mailed information. Without sponsorship, the Club depends on members' subscriptions, which can be paid at the first meeting attended.

To join, please contact Mrs Joan Atkinson at: Centre for Software Reliability, University of Newcastle upon Tyne, NE1 7RU; Telephone: 0191 221 2222; Fax: 0191 222 7995; Email: csr@newcastle.ac.uk

CONTENTS LIST

Interdependence of Safety and Security

Demonstrating Safety

Safety Management

Trends in Safety Case Development

Lessons in Safety Assessment

High Integrity from Specification to Code

Interdependence of Safety and Security

Achieving Safety through Security Management

John Ridgway
Serco Integrated Transport
Stockton-on-Tees, England

Abstract

Whilst the achievement of safety objectives may not be possible purely through the administration of an effective Information Security Management System (ISMS), your job as safety manager will be significantly eased if such a system is in place. This paper seeks to illustrate the point by drawing a comparison between two of the prominent standards within the two disciplines of security and safety management.

1 Introduction

If you have ever found yourself on a safety-related project and in the unfortunate position of having to provide the safety justification for using MicroSoft Windows, then, like me, you may have been tempted to seek favourable testimony on the internet. Yes, I know that was naïve of me, but in desperation there lies blind optimism. And so, with a disarming eagerness, I typed in 'Windows safety' into my search engine and sat back to survey the measured deliberations of the safety engineering community; doubtless some would be encouraging me, whilst others would be pretty much against the whole idea. The plan was simple – construct the argument based upon the supporting comments and pretend one hadn't read the dire warnings.

Well, the only thing wrong with my plan was that, out of a quadrillion hits offered up to me, not a single one was prepared to say anything more reassuring than 'death and prosecution awaits all who try!' This wasn't the ringing endorsement I was looking for. Furthermore, even I couldn't help noticing that, whenever a detailed demolition was provided, it invariably focused on the security weaknesses of MicroSoft's redoubtable portfolio. Indeed, I strongly suspected that if I had typed 'Windows security' I might have been presented with exactly the same quadrillion hits. As far as my PC is concerned, it seems 'security' and 'safety' are synonyms.

Now let me tell you from the outset, I don't believe this for a minute. I have experienced the dubious pleasure of putting together my employer's Information Security Management System (ISMS) based upon the BS 7799 security standard, and I have also had occasion to delve into the cornucopia of delights known as IEC 61508. As far as I could determine, these two standards, each magnificent in their own way, were definitely addressing different subjects; I was sure of that.

However, my experiences have also left me with another, perhaps more insightful, conviction – whilst security may be insufficient to achieve safety, you can still go a long way towards meeting your safety objectives if you abide by a suitable security management regime. So convinced have I become of this, that I have agreed to put flesh to the idea by critiquing the above standards and performing what is known in management-speak as a gap analysis. I could have chosen other standards and frameworks such as DEF STAN 00-56 and ITSEC as the basis of my argument but the words of my old physics teacher came to mind: 'When describing your methods, Ridgway, you would be better off if you stuck to those you have actually used'. Good advice, which I sometimes follow to this very day.

In fact, I should point out that, such is the fast and furious world of standards development, it is possible that the versions I used for my critique may have been superseded by the time you read this. For the record, therefore, the security standard I used is BS ISO/IEC 27001-2:2005[1], which has as its partner in crime BS ISO/IEC 17999:2005[2]. For the safety management standard, I have used BS EN 61508:2002.

2 General Parallels

2.1 Risk Management and the Nature of the Risks

2.1.1 The Underpinning Principles

The first thing that needs to be said is that the commonality between security management and safety management is underpinned by the fact that they are both forms of risk management. And, as far as risk management is concerned, there is very little that is new under the sun. The basic idea is very simple:

- Establish a corporate policy.
- Set yourself appropriate objectives.
- Determine what risks are currently present.
- Do something to reduce the risks to acceptable levels (i.e. levels that are commensurate with meeting your objectives).
- Check the effects of your actions.
- Constantly review the situation.

Everything you are likely to encounter in the risk management literature will be a variation on the above theme. If your organization already has in place a culture of risk management, such as that required by IEC 27001, then you hav꜀ a good foundation for dealing with safety risk when introducing new systems. The question is, just how far can you go in harmonizing the risk management processes designed for the two matters in hand: security and safety?

[1] This has supbrseded BS 7799 Part 2. It is the set of requirements.

[2] This has now superseded BS 7799 Part 1. It is the set of accompanying guidelines.

2.1.2 The Nature of the Risks

Differences in approach are, to a certain extent, driven by differences in the nature of the risk, as implied by the terminology used by the two disciplines. IEC 27001 defines information security as 'preservation of confidentiality, integrity and availability of information'. IEC 61508 defines[3] safety as 'freedom from unacceptable risk', where risk is defined as 'a combination of the probability of occurrence of harm and the severity of that harm'. In turn, harm is defined as 'physical injury or damage to the health of people either directly or indirectly as a result of damage to property or the environment.' Already, you should appreciate that information security management may bear upon safety management, since it includes security controls that protect against damage to information processing systems; damage which may in turn lead to physical harm[4]. It is significant that integrity is specifically mentioned in IEC 27001's definition of security, given the prominence accorded by IEC 61508 to functional Safety Integrity Levels (ref. IEC 61508-1, clause 7.6.2.9). IEC 27001 does not go so far as to define levels of integrity, but the parallel is clear; very often the measures required to ensure informational security are identical to those required to achieve functional safety integrity, especially when dealing with software.

2.1.3 Risk Assessment

The different definitions also lead to differences in the way that risk is assessed. In safety management, the risk level can be determined simply by defining potential levels of physical harm (often based upon the numbers of people killed or seriously injured) and then combining these with the probability of the associated hazardous event (possibly including a measure of the exposure to the hazard). For information security management, the risk assessment often involves the determination of three components: the asset value, the vulnerability resulting from existing gaps in protection, and the level of environmental threat, i.e. the likelihood that the vulnerabilities will be exploited. There are parallels between these two assessments but, to fully appreciate this, one has to understand how asset value may be calculated.

In practice, asset value is usually more to do with mission criticality than the asset's replacement cost. And whilst IEC 27001 may be concerned more with the impact upon your business processes than your duty of care, there can be no doubt that failure to exercise this duty has a business impact. It may seem cynical, but to view safety risk in terms of organizational security one simply has to think of people in terms of their organizational importance[5]. Indeed, attributing a monetary value to a loss of life is often the bottom line used to balance risks and to determine whether

[3] IEC 61508 definitions are provided in Part 4 of that standard.

[4] For example, a security breach that leads to the unavailability of a safety function has obvious safety importance.

[5] If you harm your employees you may loose their services or they may sue you. Members of the public will be equally eager to resort to litigation.

reasonably practicable measures have been implemented. I should also point out that I am not the first person to suggest to you the relevance of the concept of asset value when thinking about safety (Maguire, 2006).

In conclusion, the definitions used by the security and safety management fraternities may seem very different, but underneath they are basically using the same formula:

$$\text{Risk} = \text{Likelihood} * \text{Impact}$$

And when thinking about the impact, two of the three components of the definition of information security, i.e. integrity and availability[6], can be seen as having a direct safety importance.

2.2 When is Enough, Enough?

2.2.1 ALARP and all that

I mentioned above the importance of having a risk management culture in place, and I suggest now that this is more fundamental to safety management than having a so-called safety culture. In particular, having a risk management culture will mean that you are used to the idea of determining the nature of jeopardy and dealing with it with measures that are commensurate with your organization's appetite for risk.

Of course, this appetite must be properly aligned with your legal obligations (the good ship Corporate Governance and all who sail in her). In safety management, this has resulted in the concept of As Low As Reasonably Practicable (ALARP), which you will also find associated with health and safety management in general. ALARP is covered in some detail in IEC 61508 Part 5 but is not mentioned at all in IEC 27001. Nevertheless, the security standard does speak of introducing security controls to reduce risk to acceptable levels and it requires that management explicitly accept the residual risks.

On the other hand, IEC 27001 embodies the concept of Continual Improvement[7] and the Plan Do Check Act (PDCA) cycle[8], neither of which are explicit in IEC 61508. To understand the potential relevance of this to safety management, it is important to remember where the idea of Continual Improvement comes from and why it is relevant to an organization.

[6] I have to admit that I have difficulty seeing how matters of confidentiality can feature, directly or otherwise, in a safety assessment.

[7] This used to be known as Continuous Improvement until some wag pointed out that 'continuous' implies an unbroken continuity and that isn't what the gurus and pundits had meant. It has now been corrected, but it is worth noting that it is the same gurus and pundits that say that error prevention is better than error correction!

[8] In this respect, IEC 27001 is aligned with ISO 9001 and ISO 14001.

2.2.2 Is Continual Improvement Relevant?

When dealing with quality management, it is obvious that there is a corporate survival benefit in continually improving the quality of the product for a given outlay; otherwise your competitors may gain a commercial advantage. There is no reason to believe that such commercial competition will ever come to an end, and so the need for indefinite improvement may be assumed. In security management, there is a similar form of selection pressure that results in evolutionary improvement; however, in this instance, you are in competition with those who seek to exploit your security vulnerabilities. Such ingenious ne'er-do-wells are unlikely to agree to a truce, and so indefinite improvements in security controls may be necessary simply to maintain current levels of security.

But in safety management where is the pressure for indefinite improvement supposed to be coming from? Unless you are working with a self-imposed ambition to drive safety risk down to arbitrarily low values, there is no reason to apply further effort once the Broadly Acceptable risk level has been achieved.[9] The PDCA cycle can be used as the basic process for introducing the controls that drive down risk, but it does not have an inbuilt cut-off. One is supposed to use the PDCA cycle indefinitely; therefore, management would have to remove further risk reduction from their PDCA agenda once the Broadly Acceptable Risk level has been reached. From that point onwards, the PDCA cycle is only relevant to safety management to the extent that increased risk efficiency may be sought (i.e. can you achieve the same risk levels more cost-effectively?). There is also the question of continual maintenance of the system in order to ensure that the required safety objectives continue to be achieved throughout the life of the system. Even though the PDCA cycle does not feature explicitly in IEC 61508, there is no reason why it cannot be employed as part of the operation, maintenance, modification and retrofit activities of the overall safety lifecycle (ref. IEC 61508-1, clauses 7.7, 7.15 and 7.16).

2.3 Required Documentation

The documentation sets called for by the two standards are a reflection of the lifecycles that the documents are intended to support. In the case of IEC 61508, the documents are used to support a development lifecycle, and so many of them are plans or outputs of development activities (Annex A of IEC 61508-1 provides an example documentation structure). In the case of IEC 27001, the documents are used to record the organization's policies and then demonstrate how the policies are implemented in terms of selected security control measures. Further documentation is produced as output of the PDCA cycle (e.g. reviews, audit reports, corrective actions, etc.). This set of documents then forms the basis upon which third party certification is sought. Specifically, the documents called for by IEC 27001 are as follows:

- Security policy and objectives manual.
- Statement of Scope for the system.

[9] As per ALARP.

- Risk assessment procedure.
- Risk assessments and asset register.[10]
- Descriptions of security controls.
- Procedures for maintaining the security controls and reviewing/maintaining the management system.
- Statement of Applicability (describing the adoption of controls advocated by IEC 27001, Annex A).
- Security records (e.g. vetting reports, logs, incident reports, etc.).

An examination of the above list should convince the reader that there are a number of parallels to be made with the documentation advocated by IEC 61508. In particular, by documenting the arrangements called for by clause 6 of IEC 61508-1, the ubiquitous Safety Management Plan will cover much of the safety equivalent of the above. In addition, insofar as design documents identify the information assets and information processing assets, the design specifications produced under the IEC 61508 framework will serve to identify many of the assets that will require registration. Even IEC 27001's Statement of Applicability has an obvious counterpart in the cataloguing of the adoption of measures and techniques advocated for a particular SIL.

The import of the above is that much of the documentation that will have been produced for IEC 27001 purposes will be of relevance to IEC 61508, and vice versa. I'm not trying to say that your ISMS documentation is all you will need. After all, I've already conceded that achieving security is necessary, but not sufficient, to the achievement of safety, and this is bound to reflect in the sufficiency of any documents produced. However, there is much scope for re-use and adaptation. You would definitely not be starting with a blank sheet of paper.

2.4 Organizational Issues

By now, if you're still with me, you will have come to expect that the commonality of the underlying risk management processes is likely to result in common themes when it comes to looking at organizational arrangements. And if you harbour such expectation, then I don't intend disappointing you. Evidence the following list of managerial responsibilities. The list is taken from IEC 27001, but I defy you not to be able to apply it directly to IEC 61508:

- Establish policy.
- Ensure objectives and plans are established.

[10] Strictly speaking, IEC 27001 does not mandate an assets register, but it is one of the controls advocated in Annex A [ibid]. Personally, I fail to see how an effective ISMS can work without one. See section 3.3 for a further discussion of the importance of asset management.

- Establish roles and responsibilities.
- Communicate to the staff the importance of meeting objectives, conforming to policy and their responsibilities under law.
- Provide sufficient resources (ensuring adequate training, awareness and competence).
- Decide criteria for acceptance of risk.
- Ensure internal audits are carried out.
- Conduct management reviews.

The differences, of course, lie in the detail. For example, the training required will obviously depend upon whether it is security or safety awareness that is at stake. Nevertheless, the overall framework applies equally to both disciplines. Any management that has set out its stall to obtain certification to IEC 27001 (or ISO 9001 for that matter) will have in place the basic wherewithal to tackle the organizational challenges presented by IEC 61508. You can characterize this by talking about having a safety management culture, but more fundamental than that is the need to simply have a management culture. By this I mean that your management doesn't question the importance of clear leadership and all that goes with it[11]. In practice, I suspect that most organizations fail in their safety objectives, not because they fail in establishing a safety culture, but because they probably don't have the wherewithal to establish a culture of any chosen form. If this sounds a little harsh, I should point out that forming a desired culture is no mean feat and takes a lot more than the endless proclamation of corporate rhetoric.

3 SPECIFIC CONTROLS

In addition to specifying a general framework for the establishment, implementation and review of security policy, IEC 27001 also advocates a number of specific control objectives and controls that are deemed important in establishing an effective ISMS[12]. These controls are not intended to be exhaustive, nor are they mandatory; the Statement of Applicability is used to document and justify exemptions. In very broad terms, Annex A of IEC 27001 is analogous to the measures and techniques advocated by IEC 61508 (ref. IEC 61508-7). Therefore, to complete my critique of the benefits of sound security management in the pursuit of safety objectives, I shall run through each of the objectives and controls of IEC 27001, Annex A and offer my comments. For your information, the subtitles below include the precise Annex A reference for each subject area.

[11] Of course I am talking about the desired reality, rather than the managerial self-delusion that is so sadly commonplace.

[12] These are specified in Annex A of IEC 27001 and further described in BS ISO/IEC 17799:2005.

3.1 Security Policy (A.5)

IEC 61508 requires that the organization's management should state its safety policy (ref. IEC 61508-1, clause 6.2.1). Insofar as the achievement of security objectives may be a necessary element of the strategy for meeting safety objectives, the security policy established for IEC 27001 may be called up by the safety policy.

3.2 Organization of Information Security (A.6)

As discussed above, implementing the organizational framework for IEC 27001 will go a long way towards providing the managerial regime within which IEC 61508 objectives may be pursued. Any management system in which responsibilities are clearly defined and management are committed to providing adequately trained resources will provide the foundation required for IEC 61508. Furthermore, there are a few specific controls advocated by Annex A of IEC 27001 that seem to be equally good advice for those wishing to establish a sound organizational structure for the purposes of supporting safety objectives.

The first of these is the establishment of a management authorization process for the introduction of new systems. IEC 27001 stops short of mandating the production of 'Security Cases' (by analogy to the Safety Case) but, there again, IEC 61508 doesn't specifically call for Safety Cases either[13].

The second security control I have in mind is the establishment of suitable contacts with relevant bodies such as emergency services and local authorities (this is particularly relevant when considering Business Continuity, more of which later). I presume this would be taken as read by the safety manager.

Finally, IEC 27001 emphasises the benefits to be gained by joining special interest groups but, then again, I am obviously preaching to the converted on that one!

Interestingly, whilst IEC 27001 calls for independent review of security policy, it doesn't go as far as IEC 61508 in its advice as to just how independent such review should be (ref. IEC 61508-1, clause 8.2.14). This is probably because IEC 27001 has no concept of Security Integrity Level analogous to the SILs of IEC 61508.[14]

[13] IEC 27001 is a management system standard and, as such, is not to be used as the basis for a security evaluation of a particular computer system. In contrast, standards such as the IT Security Evaluation Criteria (ITSEC) or the more recent Common Criteria (CC), as specified in international standard ISO/IEC 15408, describe frameworks for system evaluations. Such frameworks allow computer system users to specify their security requirements, for developers to make claims about the security attributes of their products, and for evaluators to determine if products actually meet their claims. The documentation of such claims, together with the evaluation thereof, constitutes a Security Case by any other name.

[14] Once again, the reason for this is that IEC 27001 is a management system standard and is not to be used as a standard upon which a particular security evaluation may be made. Compare CC and its prescription of seven security evaluation levels, EAL 1 to EAL 7.

IEC 27001 is somewhat concerned with the risks associated with allowing access to an organization's information assets and information processing assets. For example, special arrangements are advised when setting up contracts with maintenance contractors or allowing access or contact by members of the public. These are, to a certain extent, driven by confidentiality concerns, and these are not normally a safety issue. However, system integrity may also be at stake. For example, it should be remembered that a system that is working in a public environment usually has to be that much more robust for security purposes and the same may be said for safety management, since any act [of vandalism] that leads to loss of system integrity may have a safety importance.

Of course, when it comes to safety management, the issues surrounding the use of maintenance contractors take on greater significance. The recent Potters Bar and Hatfield railway accidents provide good examples, but such problems are not limited to the railways sector. It wasn't so long ago (1997) that the maintenance contractor for the Queen's Flight broke the cardinal rule of servicing more than two engines, at a time, on a four-engine jet. Thankfully, the pilot discovered the problem in good time, leaving himself the luxury of one working engine with which to land. It is not for me to suggest that economics had anything to do with the procedural lapse leading to this particular incident[15], but such considerations cannot be ignored when commercial subcontracts are involved.

3.3 Asset Management (A.7)

Whilst effective asset management is central to the purposes of sound information security management, it does not appear to play such a central role in IEC 61508. However, in my opinion, this is one of the weaker areas of that standard. It is undoubtedly the case that maintaining an inventory of equipment in a safe state presupposes that the location and status of all items is known. Furthermore, IEC 27001 emphasises the need to define asset ownership and establish the rules for acceptable use. Both of these strike me as being useful and relevant details when ensuring that assets are maintained in a safe state.

It is conceivable, therefore, that an asset register compiled for the purposes of security management would be suitable for safety management purposes (and vice versa), or at least it would provide a natural starting point. For example, from the safety management perspective, it would be useful to add data that records the current state of the asset concerned and when it was last maintained. It is worth noting that the creation of such a database was one of the key recommendations made following the Hatfield rail crash of 17th October 2000 (ORR, 2006). In that incident, Railtrack's failure to possess a global picture of the current state of the rail network was cited as one of a number of contributory factors. The problem was also exacerbated by the fact that the maintenance work had been outsourced and there were significant deficiencies in the manner in which the subcontractors were being managed. Both problems were rectified as a result of Network Rail's decision to bring all rail maintenance back in-house.

[15] The Parliamentary Select Committee were, however, less coy on the subject.

3.4 Human Resources Security (A.8)

Anything that can be done to ensure human resources security will also benefit human resources safety.[16] Consequently, many of the controls that are listed in IEC 27001, Annex A are also of interest to the safety manager. Consider the following list:

- Ensure that everyone works to a documented job description that calls upon applicable role definitions.
- Screen individuals as part of the recruitment process.
- Ensure that adherence to your organization's policies forms part of the terms of employment.
- Supervise staff to ensure compliance with policy.
- Train staff to ensure awareness of policies.
- Treat violation of policy as a disciplinary matter.
- Ensure access control policies are brought up to date when staff leave the organization.

In consideration of the above list, the only thing that changes between matters of security and matters of safety is the policy concerned; the controls implied are equally applicable in their general form.

3.5 Physical and Environmental Security (A.9)

Clearly, physical and environmental security has, amongst other things, the purpose of protecting assets against damage or any interference that may compromise functionality. There can be no question, therefore, that implementing the physical security controls advocated by IEC 27001 may contribute to meeting an organization's safety objectives. Specifically, the following are recommended by IEC 27001 and have obvious safety relevance:

- Ensure that there is a secure perimeter to any premises within which the system operates.
- Provide physical entry controls for secure areas.
- Physically secure offices, rooms and facilities where necessary.
- Protect against environmental threats such as fire, flood, earthquake or any other horrid thing you can think of.

[16] See, for example, IEC 61508-2, Table B.4.

- Provide suitable safeguards against environmental influences such as humidity, temperature, electromagnetic interference, etc. (cf. IEC 61508-2, Table A.17).
- Ensure power supply integrity (cf. IEC 61508-2, Table A.9).
- Ensure the integrity of cabling (cf. IEC 61508-2, Table A.13).
- Make sure that all equipment is correctly maintained (cf. IEC 61508-7, clause B.4.1).

3.6 Communications and Operations Management (A.10)

This is quite a broad subject area within IEC 27001, including, as it does, controls that are several and varied. And it is perhaps in this area, more than most, that the parallel between security and safety objectives is at its most pronounced. Indeed, *coincidence* would be a better term to use than *parallel.* Of those controls that are listed in IEC 27001, I offer the following as being of particular interest to the safety manager (corresponding IEC 61508 references are supplied where appropriate):

- Document all operational procedures (cf. IEC 61508-1, clause 7.7).
- Ensure effective change management (cf. IEC 61508-1, clause 7.16 and IEC 61508-7, clause C.5.24).
- Segregate operational duties to avoid unauthorized, unintentional or even malicious misuse of equipment.
- Keep development and operational environments separate.
- Establish and monitor service level agreements for servicing and maintenance contractors (cf. IEC 61508-1, clause 6.2.5).
- Take account of the business/mission criticality of systems when negotiating service level agreements.
- Ensure system capacity planning (cf. IEC 61508-7, clauses C.5.20, C.5.21).
- Implement procedures for acceptance of new systems and subsequent modifications (cf. IEC 61508-1, clause 7.16).
- Establish controls aimed against protection from malicious code.
- Perform regular and sufficient system back-ups.
- Implement network controls and establish network service agreements.
- Implement media handling procedures.
- Establish policies and procedures to ensure the integrity of systems when connected to external systems.
- Operate a regime of audit and monitoring of system use to obtain an early indication of corruption or attempted unauthorised access.

- Maintain system operation and fault logs and perform appropriate analysis at suitable intervals.

I think it's hard to argue with any of the above. Although the list comes from IEC 27001, it could also pass as a set of subjects covered by either a safety management plan or an operation and maintenance plan.

3.7 Access Control (A.11)

The safety importance of establishing effective and appropriate access control is acknowledged by IEC 61508 (ref. IEC 61608-7, clause B.4.4). Failure to ensure that only suitably authorised and trained individuals have access to a system has clear implications for system integrity. IEC 27001 offers several controls aimed at achieving the security objective of preventing inappropriate access. Whilst some of these are primarily aimed at maintaining confidentiality, the majority also have the effect of protecting system integrity and are, therefore, germane to safety management. Hence, I suggest that the following IEC 27001 security controls and control objectives should be added to your safety management shopping list:

- Establish a clear, detailed and documented access control policy.
- Create a formal process for user registration and deregistration.
- Restrict and control the allocation of user privileges, e.g. define the set of additional options available to a system administrator.
- Implement an effective password management regime.
- Regularly review user access rights.
- Establish a policy for use of network services.
- Use appropriate authentication methods to control access by remote users.
- Consider use of automatic equipment identification to authenticate network connections.
- Strictly control access to diagnostic ports.
- Segregate groups of information services, users and information systems on networks.
- Control access to shared networks.
- Exercise effective network routing control.
- Strictly control access to operating systems (e.g. secure log-in, password management, automatic log-out upon timeout, etc.).
- Tightly control the use of system utility programs.

The general rule is simple. If a system allows inappropriate access, then that system's integrity is under threat, and that clearly has a potential safety importance.

3.8 Information Systems Acquisition, Development and Maintenance (A.12)

Given that this is the whole subject matter of IEC 61508, one would expect there to be some interesting parallels within this section of Annex A of IEC 27001. For example, one should not be surprised to hear that IEC 27001 calls for all security requirements to be properly specified prior to system acquisition or development. I don't really have any great insights to offer here[17] other than to point out that, by specifying the security and safety requirements in the same document, one may be in a better position to appreciate the extent to which they are mutually supportive. Thankfully, neither standard is prescriptive when it comes to the required document structure (see section 2.3) and so one is at liberty to unify documents in this manner.

IEC 27001 lists a small number of controls aimed at ensuring correct processing within software applications. These certainly don't go anywhere near the level of advice provided by IEC 61508, but I list them here anyway for your consideration:

- Applications should validate data upon receipt.
- Validation checks should be incorporated into applications to detect any corruption of information through processing errors or deliberate acts (this is reminiscent of the failure assertion programming covered in clause C.3.3 of IEC 61508-7).
- In communications applications, appropriate controls should be identified and implemented in order to ensure message authenticity and to protect message integrity (for example, cf. IEC 61508-7, clause C.3.2).
- Output from applications should be validated to ensure that the processing of stored information is correct and appropriate to the circumstances.

In addition to the above, IEC 27001 advocates the use of cryptographic controls. Whilst one may be tempted to dismiss these as being purely concerned with maintaining confidentiality (and hence of limited interest to the safety manager), the truth is that they may also be used to protect authenticity and integrity, which should be enough to kindle interest in the safety camp.

Furthermore, IEC 27001 also advocates a small number of controls that may be exercised during system development in order to promote the resulting system integrity. Once again, the implementation of these controls comes nowhere near to matching the measures and techniques advocated by IEC 61508. Nevertheless, their safety relevance is clear, and so it would be remiss of me to overlook them here:

- Implement sound Change Control procedures (nothing new here).

[17] Note, however, that this subject was ably covered in a paper presented at the 13th Safety-Critical Systems Club Symposium (Lautieri, Cooper, Jackson, 2005).

- Technically review applications following operating system changes (this is an example of impact analysis, as advocated by IEC 61508-1, clause 7.16.2.3).
- Restrict changes that are made to acquired software packages.
- Control the installation of software on operational systems.
- Protect and control test data.[18]
- Restrict access to program source code.
- Supervise and monitor outsourced software development.

All good advice, but frankly, I'm not sure I would go anywhere near a safety-related system whose development was outsourced, unless the outsourcing was specifically to take advantage of the safety expertise of the organization concerned.

Finally, there is another item of IEC 27001 advice in this area that strikes me as particularly germane to the safety case (especially when justifying the use of Software of Uncertain Pedigree (SOUP)). It is the need to obtain, and act upon, timely information relating to system vulnerabilities. In particular, this means that the IT department should keep abreast of the latest viruses, worms, etc. that are doing the rounds, and act quickly to introduce the appropriate safeguards. Such vulnerabilities are likely to apply to specified items of SOUP.

3.9 Information Security Incident Management (A.13)

IEC 27001 includes a set of guidelines aimed at ensuring that security events and weaknesses associated with information systems are communicated in a manner that allows timely preventive and corrective action to be taken. The parallel with safety incident reporting is clear enough, but the question is whether a single reporting mechanism could be made to apply to both security and safety, i.e. if you already have security incident management procedures in place, do you have the basis for meeting the safety incident management requirements of safety standards such as IEC 61508?

Well, in light of the controls deemed appropriate by IEC 27001, and given IEC 61508's lack of prescription on this subject (ref. IEC 61508-1, clause 6.2.1, paragraph i)), it is difficult to argue that a unified system couldn't be made to work. To meet IEC 27001, your security incident management procedures should:

- Report events through appropriate management channels as quickly as possible.
- Require that all parties concerned note and report any observed or suspected weaknesses in either the system or associated services.
- Ensure a quick, effective and orderly response to reported incidents.

[18] Data management is another of those areas where I think IEC 61508 could be improved.

- Monitor and analyse the types, volumes and costs of incidents.
- Collect any evidence required to proceed with any legal action deemed necessary as a result of an incident.

The above list is provided in IEC 27001 in regard to security incidents, but by simply avoiding specific reference to 'security' I believe I have offered a list of controls that applies equally to safety incident management. Therefore, I think it should not be beyond the whit of the average manager to construct a set of procedures that can be applied generically; and an existing security incident management system should provide an ideal starting point. If you are an IEC 27001 certificated organization with security incident management procedures that do not offer such a foundation, then you have been seriously short-sighted.

Incidentally, the presupposition behind the collation of evidence is that your organization will be taking legal action against parties who have breached corporate policy. This may be the case for either security or safety incidents, though I must admit that your interest in collecting safety evidence may very well be motivated more by the need to protect against litigation, rather than to instigate it.

3.10 Business Continuity Management (A.14)

Business Continuity is a reference to those arrangements that are put in place by your organization to counteract interruptions to business activities, to protect critical business processes from the effects of major failures of business systems or (heaven forbid) disasters, and to ensure their timely resumption. In the case of IEC 27001, the specific concern is the impact on information processing systems. For IEC 61508, the concern would be the impact on safety systems and the presumption would be that system availability is a safety issue. However, I don't think that the motives for requiring business continuity have a great deal to do with the tenets to be followed. Consequently, I suggest that the implementation of the following controls (taken from IEC 27001) is every bit as relevant to the meeting of safety objectives as it is to the meeting of security objectives:

- Identify the events that can cause interruptions to business processes and assess the probability and impact of such interruptions.
- Develop and implement plans to maintain or restore business critical operations (in the required timescales) following interruption to, or failure of, key business systems.
- Establish and maintain a single framework for business continuity plans that ensures a consistent approach and identifies priorities for testing and maintenance of the arrangements.
- Test and update regularly the business continuity plans to ensure that they are up-to-date and effective.

You will note that the term 'business process' is being used here in its most general sense and may include the functioning of a safety-related system. Safety engineers sometimes prefer terms such as 'mission critical' to 'business critical' but we are talking about the same thing. Suffice it to say, many of the practical arrangements that may be found in the average business continuity plan (e.g. provision of redundant systems, auxiliary power units, system back-up and recovery procedures, etc.) look awfully familiar to anyone who has spent any time at all wading through the measures and techniques presented in IEC 61508's many and magnificent tables.

3.11 Compliance (A.15)

IEC 27001 includes a number of controls and control objectives that may help you avoid breaches of any legal, statutory, regulatory or contractual obligations. Whilst the subject matter is specifically security-related (e.g. intellectual property rights, data protection and the privacy of personal information, prevention of misuse of information processing facilities, etc.), many of the general principles apply universally. Indeed, any company that has got its act together with respect to corporate governance should already have in place the managerial framework for ensuring legal, statutory, regulatory and contractual compliance; no matter what the subject. In particular, the protection of organizational records, whilst listed in IEC 27001 as a security concern, has clear ramifications when safety management is concerned. An organization that does not routinely identify its important records, and protect them accordingly, will find it all the more difficult to support the regime required for the effective compilation of the average safety case. As with risk management, this is one of those examples where the existing managerial culture may make the safety manager's job all the more easy, or drive him[19] to drink.

And whilst we are talking about professional angst, spare a thought for the person who has to audit all of this. Contrary to popular opinion (and I speak from personal experience here), the auditor's job is not a second-best alternative for those who missed their way as a traffic warden. On the contrary, reporting upon the non-compliance of colleagues can be a wretchedly unfulfilling experience; not to say career limiting if the miscreant has friends in high places. For this reason alone, it is commonplace to give all auditing duties to the same hapless individual, since it is rare to find two people in the one organization who possess the required combination of professional commitment and personal disregard. It is perhaps fortunate, therefore, that the skills and insights an auditor requires to navigate the choppy waters of security misdemeanour are not a million miles from those that may avail the safety auditor. Put another way, if you are looking for someone to perform your safety management audits, you could do a lot worse than pick someone with a security/quality management background; and I'm sure I could offer very competitive rates.

[19] I am not being sexist here. Dipsomania is not restricted to the male gender, any more than is the lack of good judgment leading to a person choosing safety management as their career.

4 General Conclusions

I hope that the forgoing comparison of IEC 27001 and IEC 61508, scant though it may be, has been sufficient to persuade you that the differences between the general principles of security management and safety management are not as significant as the similarities. I hope I haven't left myself open to accusations of the 'one size fits all' approach, since the devil is most certainly in the detail. Nevertheless, anyone who has any degree of exposure to the two disciplines cannot help but be struck by the number of times that the same topics keep cropping up. The language used may at times give the impression that the subjects are importantly distinct, but this is often an illusion and if you were to scratch the surface you will find that both camps are really talking about the same thing: good old-fashioned risk management. Furthermore, this paper has identified several specific controls that have been proposed for security management but are equally pertinent to the meeting of safety objectives. The paper concludes that the pursuit of security objectives is often necessary, though rarely sufficient, for the attainment of safety objects.

The favoured buzz phrase of the image conscious manager is 'integrated solution'. Often this term is used vacuously and portends nothing. On this occasion, however, I put it to you that the term has a genuine importance and integration will lead to a cost-effective and consistent approach to corporate security and safety management. This paper has identified a number of candidate areas for integration but has not sought to indicate how such integration may be achieved. It has, however, shown how the implementation of just one of the available security standards (IEC 27001) can assist in compliance with one of the prominent safety standards (IEC 61508). After focussing on only two standards, one should be wary about drawing general conclusions. Nevertheless, the results of this comparison are strongly suggestive that a harmonised approach should be readily achievable.

References

IT Security Evaluation Criteria (ITSEC) Version 1.2 (1991). HMSO.

BS EN/IEC 61508 (2002). *Functional safety of electrical/electronic/programmable electronic safety-related systems.* Commission Electronique Internationale.

ISO/IEC 27001 (2005). *Information technology – Security techniques – Information security management systems – Requirements.* ISO/IEC.

BS ISO/IEC 17799 (2005). *Information technology – Security techniques – Code of practice for information security management.* Commission Electronique Internationale.

ISO/IEC 15408 (2005). *Information technology -- Security techniques -- Evaluation criteria for IT security.* Commission Electronique Internationale.

Lautieri, Cooper, Jackson (2005). *SafSec: Commonalities Between Safety and Security Assurance.* In: Redmill F, Anderson T (eds): Proceedings of the Thirteenth Safety-Critical Systems Symposium, Bristol, UK, 8-10 February 2005

Maguire R (2006). *So how do you make a full ALARP justification? Introducing the Accident Tetrahedron as a guide for Approaching Completeness.* In: Redmill F, Anderson T (eds): Proceedings of the Fourteenth Safety-Critical Systems Symposium, Bristol, UK, 7-9 February 2006

ORR (2006). *Train Derailment at Hatfield – A Final Report by the Independent Investigation Board.* http://www.rail-reg.gov.uk/upload/pdf/297.pdf.

Towards a Unified Approach to Safety and Security in Automotive Systems

Peter H Jesty
Peter Jesty Consulting Ltd, Warwick Lodge, Towton,
Tadcaster, LS24 9PB, UK

David D Ward
MIRA Ltd
Nuneaton, CV10 0TU, UK

Abstract

At the time when IEC 61508 was being created, analogous work was also being done to harmonise security evaluation criteria. Although there was no cross-fertilisation between these two activities, the MISRA project did use the ITSEC evaluation criteria as the basis for its recommendations on the requirements for software at varying levels of integrity. This paper points out the advantages of this approach for safety engineers, and explains how it overcomes some of the difficulties that people now have in applying IEC 61508. It also shows how the approach can be used for other attributes such as electromagnetic compatibility.

1 Introduction

Whilst there has been a considerable amount of work done in the fields of Safety Engineering and Information Technology Security, and at similar times, there has been little cross-fertilisation, even though many of the problems being addressed are similar.

The field of Safety Engineering saw the creation of a number of Standards, in particular (IEC61508 1998-2000). This is based on various earlier national standards and guidelines, and its fundamental philosophy is that safety hazards must be identified, their respective risks must be assessed, and that the resulting Safety-Related System (SRS) must be developed to a level of assurance that is commensurate with that risk. There are currently four Safety Integrity Levels (SILs), and the standard contains various sets of tables of techniques and measures that should be used to meet a given SIL.

At around the same time as IEC 61508 was being created, the European Commission sponsored the harmonisation of security evaluation criteria (ITSEC 1991), based on various existing and separate national criteria from France, Germany, The Netherlands and the United Kingdom. Similar work had also taken place elsewhere, and these results were then pooled to create a single set of Information Technology (IT) security criteria, which became (ISO/IEC15408 2005). The fundamental philosophy is centred on the concept of Protection Profiles and Security Targets. A Protection Profile is an implementation-independent set of security requirements for a category of Target of Evaluations (TOEs) that meet specific consumer needs, and a Security Target is a set of security requirements and specifications to be used as the basis for evaluation of an identified TOE. The Security Target includes a set of assurance security measures, each of which is specified as one of eight Evaluation Assurance Levels (EALs)[1]. ISO/IEC 15408 contains a hierarchically ordered set of assurance procedures to achieve a given EAL.

These two different approaches have continued in parallel with little or no cross-fertilisation, with the exception of the original MISRA Guidelines (MISRA 1994), which will be discussed below.

1.1 Automotive Applications

Whilst all vehicles now have a number of SRSs, at the time of writing there are few vehicle systems in production that have security issues, but this will change soon.

At the present time security issues related to vehicle systems fall into two categories:

- Security of software (code and data) against "hacking". Many manufacturers now use "end of line" programming, where the software and appropriate configuration data are loaded into flash memory at the end of the vehicle production line. This also provides the possibility to reprogram systems in the field, for example a fix for a warranty issue or to add a new feature. Similarly some statutory data such as the vehicle odometer (mileage) reading is now stored electronically. Such software and data need to be protected against unauthorized access – either from nefarious motivations (e.g. falsifying the recorded mileage of a vehicle) or from tampering. In the latter case there is an industry that has grown up around enhancing the performance of vehicles by "chipping", that is, reprogramming vehicle systems by changing the software – typically in the engine management system. Such replacement software has not been subject to the rigour of the design and validation processes used for the OEM software.
- Whole-level vehicle security (against theft) aspects. In this case there are legislative and insurance requirements for the vehicle to be fitted with systems (including, but not limited to, electrical/electronic/programmable electronic systems) that provide the vehicle with a defined level of resistance against unauthorized entry and use.

In the future, the growth in communications and the introduction of telematic or "intelligent transportation systems" will mean that the majority of vehicles have some form of outside network connectivity. Indeed it is being proposed in some

[1] ITSEC has seven assurance levels.

quarters to use Internet Protocol based mechanisms, which could result in the scenario that every vehicle is connected to the Internet and has an IP address. This will evidently raise the stakes in terms of the security functions that are required within vehicles and their communications systems to prevent all of the security issues typically associated with computer networks from affecting the safety of road transport. This would include, but not be limited to, preventing actions such as the introduction of malware, spoofing of addresses or messages, and interfering with vehicle functions.

Clearly if a Tier 1 supplier[2] were to produce both safety-related and security-related vehicle systems, it would not wish to have two different development processes in order to receive third-party assessment for all of them. This problem would be exacerbated greatly if a particular system was both safety-related ***and*** had security issues.

2. Description of the Two Approaches

2.1 Safety-Related Systems and IEC 61508

The process of creating an electrical, electronic or programmable electronic SRS that is recommended by IEC 61508 is centred on the Safety Lifecycle (see Figure 1). The first few phases of the Safety Lifecycle require an analysis to be undertaken of how the SRS might perform in its intended environment. The objective is to identify any safety hazards that may result from the normal operation of the SRS, or from failures of one of more parts of that system. The risk associated with each hazard is analysed and safety requirements are chosen to mitigate each risk to an acceptable level.

In addition to the Safety Functional Requirements, a principal product of the Hazard and Risk Analysis process is the Safety Integrity Requirements, including the SIL, required of the SRS. The SIL is specified in terms of targets for either the average probability of the failure of the SRS to perform its safety function on demand, or the probability of a dangerous failure per hour (see Table 1). The remainder of the Standard consists primarily of instructions, usually in the form of tables of processes, on how to achieve a given SIL.

It is at this point that the Standard becomes imperative without always explaining the reasoning behind its requirements. This can become a real problem when the SRS in question does not fit neatly into the overarching system model assumed by the writers of that Standard. The demonstration of a failure rate for electrical or electronic hardware can often be done in advance of the operational phase. However, when the SRS includes software this is no longer possible, despite the demands of some assessors to do so. Indeed IEC 61508 provides no evidence as to why product integrity should be inferred from the application of the processes

[2] A supplier with prime design responsibility for key sub-systems or components of the end product.

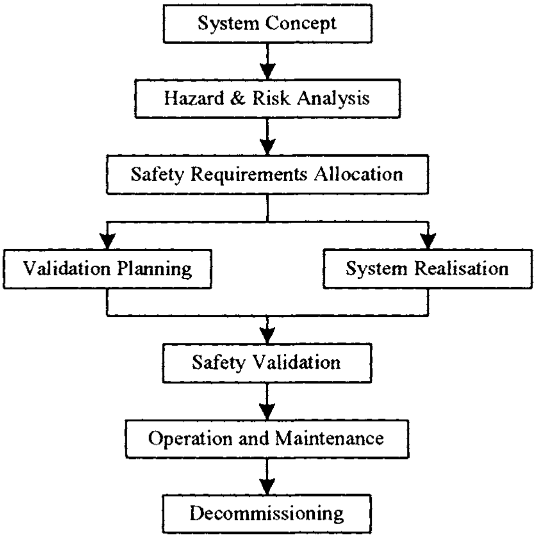


Figure 1. Simplified IEC 61508 Safety Lifecycle

SIL	**Low demand mode of operation** *(Average probability of failure to perform its design function on demand)*	**High demand, or continuous, mode of operation** *(Probability of dangerous failure per hour)*
1	$\geq 10^{-2}$ to $< 10^{-1}$	$\geq 10^{-6}$ to $< 10^{-5}$
2	$\geq 10^{-3}$ to $< 10^{-2}$	$\geq 10^{-7}$ to $< 10^{-6}$
3	$\geq 10^{-4}$ to $< 10^{-3}$	$\geq 10^{-8}$ to $< 10^{-7}$
4	$\geq 10^{-5}$ to $< 10^{-4}$	$\geq 10^{-9}$ to $< 10^{-8}$

Table 1. Target Failure Measures (IEC61508 1998-2000)

that it mandates (McDermid J and Pumfrey D J 2001). "There is an implicit assumption that following the process delivers the required integrity. More strongly, the use of a process becomes a self-fulfilling prophecy – 'I need SIL 4, I've followed the SIL 4 process, so I've got SIL 4.'" (ibid).

2.2 Security Systems and ISO/IEC 15408

The basic context in which ISO/IEC 15408 is intended to be used is shown in Figure 2. The threat agents may be actual or perceived, and the abuse commonly includes, but is not limited to, unauthorised disclosure (loss of confidentiality); unauthorised modification (loss of integrity), unauthorised deprivation of access to the asset (loss of availability). The owners of the assets will analyse the threats applicable to their assets and their environment, and determine the risks associated

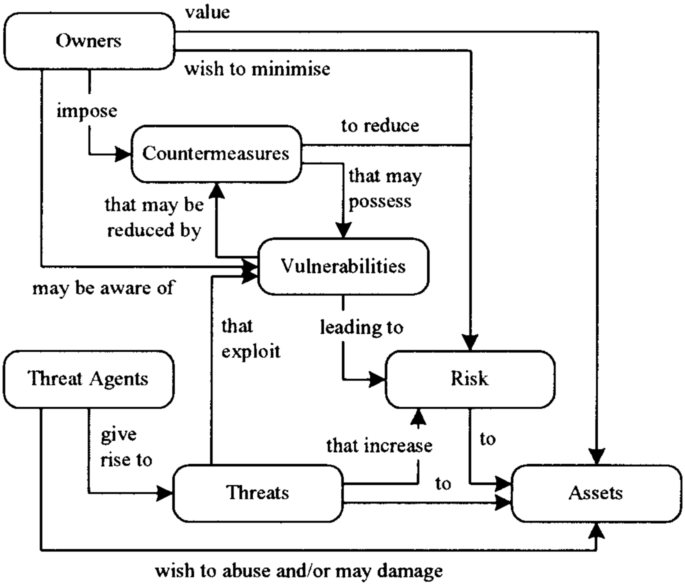


Figure 2. Security Concepts and Relationships (ISO/IEC15408 2005)

with them. They will then select countermeasures to reduce the risks to an acceptable level.

The owners need to be confident that the countermeasures are adequate to counter the threats. Since they may not be capable of judging all aspects of the countermeasures, or they may just wish to have third party assurance, an evaluation may be undertaken (see Figure 3). The Standard acknowledges that IT systems are likely to make maximum use of generic software products and hardware platforms, and that there are cost advantages in evaluating the security aspects of such a product independently and building up a catalogue of evaluated products.

The fact that an evaluation of a product may be independent of its final environment leads to three types of security requirement, whose relationship is shown in Figure 4.

- Package – a sub-set of reusable requirements that are known to be useful and effective in meeting identifiable objectives.
- Protection Profile – the implementation-independent expression of security requirements for a set of TOEs that will comply fully with a set of security objectives.
- Security Target – the expression of security requirements for a specific TOE that are shown, by evaluation, to be useful and effective in meeting the identified objectives.

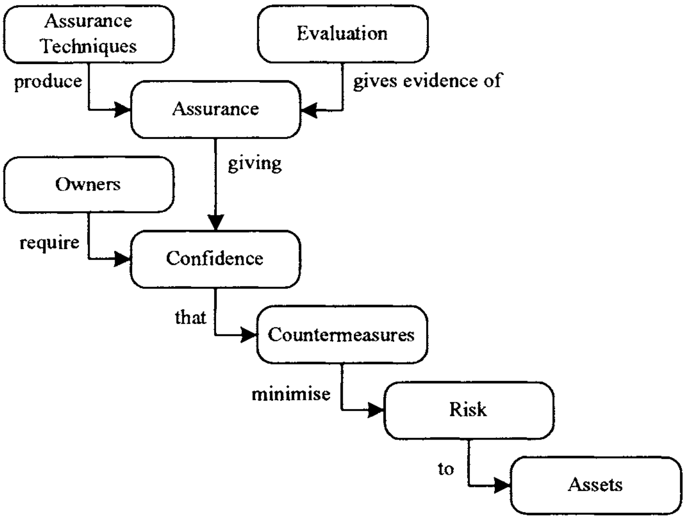


Figure 3. Evaluation Concept and Relationships (ISO/IEC15408 2005)

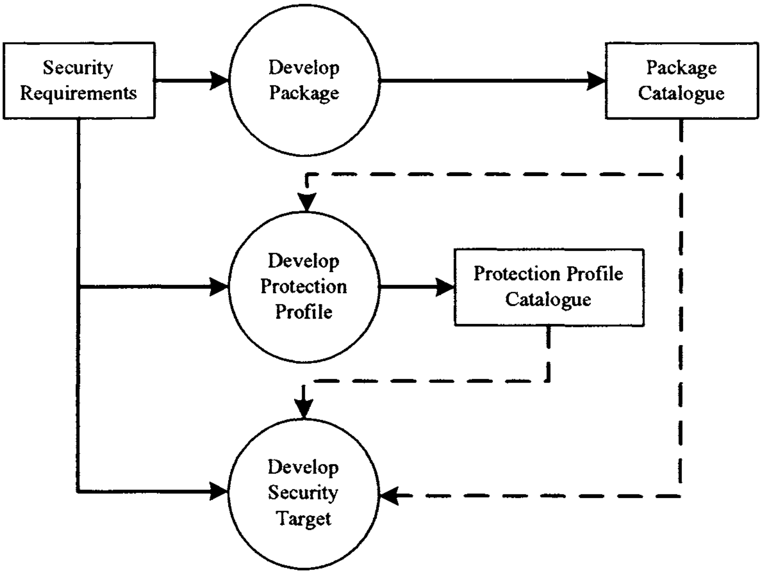


Figure 4. Use of Security Requirements (ISO/IEC15408 2005)

This leads to three types of evaluation:

- Protection Profile Evaluation – to demonstrate that the Protection Profile is complete, consistent and technically sound and suitable.
- Security Target Evaluation – to demonstrate that the Security Target is complete, consistent and technically sound and suitable; and, in the case that a

Security Target claims conformance to a Protection Profile, that it indeed does so.
- TOE Evaluation – to demonstrate that a specific TOE meets the security requirements contained in an evaluated Security Target.

The security requirements are likely to include an EAL which provides an increasing scale that balances the level of assurance obtained with the cost and feasibility of acquiring that degree of assurance (see Table 2). They are specified in terms of requirements for:

- Configuration Management
- Delivery and Operation
- Development
- Guidance Documents
- Lifecycle Support
- Tests
- Vulnerability Assessment

EAL	Assurance
0	Inadequate Assurance
1	*Functionally Tested* – This EAL provides a meaningful increase in assurance over an unevaluated IT product or system.
2	*Structurally Tested* – This EAL represents a meaningful increase in assurance from EAL 1 by requiring developer testing, a vulnerability analysis, and independent testing based upon more detailed TOE specifications.
3	*Methodically Tested and Checked* – This EAL represents a meaningful increase in assurance from EAL2 by requiring a more complete testing coverage of the security functions and mechanisms and/or procedures that provide some confidence that the TOE will not be tampered with during development.
4	*Methodically Designed, Tested and Reviewed* – This EAL represents a meaningful increase in assurance from EAL3 by requiring more design description, a sub-set of the implementation, and improved mechanisms and/or procedures that provide confidence that the TOE will not be tampered with during development or delivery.
5	*Semi-Formally Designed and Tested* – This EAL represents a meaningful increase in assurance over EAL4 by requiring semi-formal design descriptions, the entire implementation, a more structured (and hence analysable) architecture, covert channel analysis, and improved mechanisms and/or procedures that provide confidence that the TOE will not be tampered with during development.

EAL	Assurance
6	*Semi-Formally Verified Design and Tested* – This EAL represents a meaningful increase in assurance from EAL5 by requiring more comprehensive analysis, a structured representation of the implementation, more architectural structure (e.g. layering), more comprehensive independent vulnerability analysis, systematic covert channel identification, and improved configuration management and development environmental controls.
7	*Formally Verified Design and Tested* – This EAL represents a meaningful increase in assurance from EAL6 by requiring more comprehensive analysis using formal representations and formal correspondence, and comprehensive testing.

Table 2. Summary of Evaluation Assurance Levels (ISO/IEC15408 2005)

3. Merging the Two Approaches

The two approaches have a number of similarities, as well as a number of differences. Some of the differences are due to the contrasting cultures within safety engineering and IT security. Broadly speaking, safety is a characteristic that a system *needs*, and is often seen by senior management as an (undesirable) overhead, whilst security is a characteristic that the owner of a system *wants*, and is more willing to pay for.

Both approaches aim to manage risk, but whilst IEC 16508 requires an analysis to be made of each system concept to identify the hazards (only), ISO/IEC 15408 permits the existence of generic pre-defined hazards. It is also more explicit in how to use pre-evaluated products than IEC 61508.

The most obvious differences are contained within the means by which they achieve SILs and EALs respectively, and what they expect of them. Whilst IEC 61508 defines target failure measures for each SIL (see Table 1), ISO/IEC 15408 makes no attempt to do this for its EALs. ISO/IEC 15408 has been written for software only systems and the lack of any quantitative statement as to what each EAL will achieve is recognition that applying, say, failure rates to software is fraught with problems, especially in advance of any operational experience. Meanwhile, the existence of target failure measures in IEC 61508 encourages safety assessors to insist that they are met even in software intensive SRSs.

The lack of a stated, or obvious, logical basis continues in IEC 61508-3 with its recommendations on how to achieve each SIL. The various tables comprise a set of seemingly random techniques, some of which lost their topicality from the time when they first appeared in a Draft version. Since there is no statement as to what will actually be achieved by the application of a given set of techniques for a particular SIL, it is difficult to know how to replace them when using certain modern development methodologies. However, by taking an entirely different approach ISO/IEC 15408 does not have this problem with its EALs.

3.1 The MISRA Approach

The original MISRA Guidelines (MISRA 1994) were written before it was decided to make IEC 61508 a Basic Safety Publication. The authors were unhappy with the "tables of techniques" that had appeared in the first Draft, and looked for an alternative approach. The European Commission Framework II project DRIVE Safely (DRIVE Safely 1992) had had a similar problem, which it had solved by following the ITSEC approach for Assurance Levels (ITSEC 1991). Since DRIVE Safely wished to use the, then, five Safety Integrity Levels being proposed in the Draft version of IEC 61508, it was necessary to "squash" the requirements for the seven Assurance Levels of ITSEC into five levels.

The MISRA project took the results from DRIVE Safely and reviewed them. It realised that the levels were going through a series of stages that give increasing confidence that the product will behave as desired in its intended environment. These stages can be defined broadly as follows:

- Commercial considerations only
- Quality assurance procedures
- Structured approach (repeatability)
- Increasing justification
- "Proof"

The MISRA project felt that the concept of adding quality attributes as one progressed up through the Integrity Levels, as described in (ITSEC 1991), was one that both made sense, and could be "sold" to the automotive industry at the time. However, the descriptions of the Assurance Levels in (ITSEC 1991) were not as concise as they have become in (ISO/IEC 15408 2005), and so they were not used. The project also wished to provide some advice on the types of processes, techniques and tools that should be used, without mandating or forbidding anything specific, since giving high-level objectives only would be seen by many as being too academic to be practical. The review of the DRIVE Safely results therefore ensured that the advice being given provided an increase of assurance between each Integrity Level for each Development Processes, and that these increments were consistent between Development Processes. The resulting revised table of requirements for five levels of integrity (see Table 3) is supported by a Technical Report (MISRA 1995).

It should be noted that the table does not require any particular technique or technology to be used, and so it is future-proof in a way that IEC 61508-3 is not. It is also clear for both the developer and an assessor the type of process, technique or tool, and its degree of application, which is needed for the various Development Processes in the table.

A number of companies have been applying the MISRA approach since it was first published, and they have not reported any difficulties in applying it. Indeed the normal reaction is that it is preferable to using IEC 61508-3 because modern and

Development Process	Integrity Level				
	0	1	2	3	4
Specification and design	ISO 9001	Structured method	Structured method supported by CASE tool.	Formal specification for those functions at this level.	Formal specification of complete system. Automated code generation (when available)
Languages and compilers		Standardised structured language.	A restricted sub-set of a standardised structured language. Validated or tested compilers (if available)	As for 2.	Independently certified compilers with proven formal syntax and semantics (when available).
Configuration management: products		All software products. Source code.	Relationship between all software products. All tools.	As for 2.	As for 2.
Configuration management: processes		Unique identification. Product matches documentation. Access control. Authorised changes.	Control and audit changes. Confirmation process.	Automated change and build control. Automated confirmation process.	As for 3.
Testing		Show fitness for purpose. Test all safety requirements. Repeatable test plan.	Black box testing.	White box module testing – defined coverage. Stress testing against deadlock. Syntactic static analysis.	100% white box module testing. 100% requirements testing. 100% integration testing. Semantic static analysis.
Verification and validation		Show tests: - are suitable; - have been performed; - are acceptable; - exercise safety features. Traceable correction.	Structured program review. Show no new faults after corrections.	Automatic static analysis. Proof (argument) of safety properties. Analysis for lack of deadlock. Justify test coverage. Show tests have been suitable	All tools to be validated (when available). Proof (argument) of code against specification. Proof (argument) for lack of deadlock. Show object code reflects source code.
Access for assessment		Requirements and acceptance criteria. QA and product plans. Training policy. System test results.	Design documents. Software test results. Training structure.	Techniques, processes, tools. Witness testing. Adequate training. Code.	Full access to all stages and processes.

Table 3. Summary of MISRA Software Requirements (MISRA 1994)

familiar techniques and tools can be used to achieve a clear holistic objective or goal; i.e. actual confidence is obtained. When compared with IEC 61508-3 the requirements of each MISRA Integrity Level compared to the corresponding IEC SIL are broadly equivalent at SILs 3 and 4. IEC 61508 has some additional requirements at SILs 1 and 2 compared to MISRA Integrity Levels 1 and 2, but these relate to techniques that would nowadays be considered good practice in any software development process regardless of any SIL requirement. These requirements are concerned with black-box testing, language sub-sets, static analysis and impact analysis of changes.

3.2 Application to EMC

The MISRA approach to gaining confidence can be applied to other attributes for which failure rates are at best unobtainable and at worst contentious. Soon after the MISRA Guidelines were published the European Commission Framework III project EMCATT (EMCATT 1995) considered the functional safety issues of intelligent transport systems that result from a lack of electromagnetic compatibility (EMC).

The specification, development, operation and maintenance of an electrical or electronic system was considered and the processes that have an effect on the EMC identified. Each process was then considered in terms of the manner in which it was possible for one to gain confidence in what was done spread over five levels of integrity as defined above. The result can be seen in Table 4.

Development Process	Integrity Level				
	0	1	2	3	4
Specification and design	"EMC Directives".	Design for: - EMC - Maintenance	As for 1	Design for testing	Design for actual susceptibility
Test Plan	"EMC Directives".	Generic tests	Input → Output. Full operational profile.	Access to internal test points.	As for 3.
Test Conditions	"EMC Directives".	Static environments.	As for 1.	Reproducible operating environments. Worst case testing.	Actual operating environments
Validation	Commercial considerations only.	Show tests: - are suitable - are acceptable. Traceable corrections.	As for 1.	Justify: - design - test conditions. Prove calibration of equipment.	Proof (argument) of susceptibility. Justify test plan.
Physical robustness	Other relevant standards.	Unlikely possibility of change.	Remote possibility of change.	Very remote possibility of change.	Periodic re-testing.

Development Process	Integrity Level				
	0	1	2	3	4
Access for assessment	Commercial considerations only.	Requirements and acceptance criteria. Witnessed testing. QA and product plans. Training policy.	Design documents. Test documents. Training structure.	Techniques, processes, tools. Adequate training.	Full access to all stages and processes.
Preventative or Corrective Maintenance	"EMC Directives". No special tools or training required.	As for 0.	Trained technician only. Prevention of unauthorised access.	Mechanical prevention of unauthorised access.	Re-testing.
Perfective or Adaptive Maintenance	As for a new product.	As for 0.	As for 0.	As for 0.	As for 0.

Table 4. Summary of EMCATT EMC Requirements (EMCATT 1995)

Less experience has been gained with the use of the EMCATT approach, partly because it is less well known. In addition, for vehicles there is extensive reliance on an established EMC process which is derived from the requirements of the Type Approval framework. However it should be noted that many OEMs have more stringent requirements than the legislative limits for good practice reasons. These requirements are often derived from an assessment of the criticality of a failure; therefore some systems will be tested to an increased immunity requirement compared to others because a potential failure is less desirable. Thus it could be argued that an approach such as the EMCATT one is being followed on an informal basis. For software, when the MISRA Guidelines were introduced there was no pre-existing legislative requirement and hence a different approach could be introduced.

4. Conclusion

For a system to be either safe and/or secure the developer and the assessor need to have confidence that it will behave as desired in its intended environment. Whilst we tend to associate the word "integrity" with safety, and "assurance" with security, an inspection of a dictionary or thesaurus will show that they are not synonyms, and that "assurance" has a better association with safety and "integrity"! Given the similar attributes that should be owned by SRSs and by systems with security requirements, it is unfortunate that the two approaches for achieving them are currently so very different.

The MISRA approach is sometimes attacked because "it does not follow IEC 61508", but it has never been criticised for producing a system in which one could not have confidence in the safety of the final system at the required level. Meanwhile some of the techniques required by IEC 61508 are dated, and so are the references to them (e.g. who, apart from one of the authors, has access to a copy of a report from a TÜV Study Group on Computer Safety published in 1984, but which is still referenced in IEC 61508-7?). When this is combined with its required use on an increasing variety of system types, many of which were not considered by the committee that wrote it, we feel that an approach based on the properties that need to be demonstrated, rather than the techniques that should be applied, will be much easier to use, more reliable in addressing safety requirements, and more flexible in allowing the use of new and improved techniques in the future.

The manner in which EALs are defined in ISO/IEC 15408, and Integrity Levels are defined in the MISRA Guidelines, is generic. Processes must be undertaken, but the manner in which they are done is not specified, only the properties that need to be shown, or the degree to which they must be done. It is also clear how confidence in the safety or security of the system is increased up through the levels. We are certain that the automotive industry is not unique in its need to demonstrate both the safety and the security of future systems, and there will therefore be an increasing demand to use the same type of approach for both characteristics.

References

DRIVE Safely (1992). Towards a European Standard: The Development of Safe Road Transport Informatic Systems, V1051 DRIVE Safety Project of the Advanced Road Transport Telematics in Europe (ATT/DRIVE) Programme, Second Framework Programme (1989-91), 1992.

EMCATT (1995). Functional System Safety and EMC, V2064 EMCATT project of the Advanced Transport Telematics (ATT/DRIVE II) sector of the TELEMATICS APPLICATIONS Programme, Third Framework Programme (1991-94), 1995.

IEC61508 (1998-2000). Functional Safety of Electrical/Electronic/ Programmable Electronic Safety-Related Systems, International Electrotechnical Commission, 1998-2000.

ISO/IEC15408 (2005). Information Technology – Security Techniques – Evaluation Criteria for IT Security, International Electrotechnical Commission 2005.

ITSEC (1991). Information Technology Security Evaluation Criteria, Commission of the European Communities, 1991.

McDermid J and Pumfrey D J (2001). Software Safety: Why is there no Consensus? Proceedings of the 19th International System Safety Conference, Huntsville, AL, System Safety Society, P.O. Box 70, Unionville, VA 22567-0070.

MISRA (1994). Development Guidelines for Vehicle Based Software, MIRA, CV10 0TU, 1994.
MISRA (1995). Report 2 – Integrity, MIRA, CV10 0TU, 1995.

Dependability-by-Contract

Brian Dobbing and Samantha Lautieri
Praxis High Integrity Systems
Bath, UK
www.praxis-his.com
www.safsec.com

Abstract

This paper presents ongoing research by Praxis High Integrity Systems into a contract-based approach to the definition and composition of dependability characteristics of components of complex systems. The research is founded on the Correctness By Construction methodology with the main aim being to assist in the construction of a demonstrably dependable system, and of its supporting dependability case for the purposes of safety certification and/or security accreditation. Other aims are to maximise re-use, accommodate COTS, and minimise impact of change across the whole lifecycle, including re-certification. The ongoing research is based on the results of an MoD-inspired project known as SafSec – an integrated approach to safety and security argumentation.

1 Introduction

Correctness-by-Construction (CbyC) is a methodology devised by Praxis HIS (Croxford, Chapman 2005) - its primary function is to facilitate the engineering of demonstrably correct systems. It originated within the domain of software engineering, where it reversed the trend of developing software containing many bugs that are incrementally removed on detection, usually at the latter end of the software lifecycle. CbyC eschewed this approach, not only because of the extreme cost and disruption caused by tackling large numbers of bugs during system test and deployment, but also because of its lack of professionalism. Software professionals should practice the highest quality of software engineering, and that means constructing the software correctly in the first place. For an example of CbyC as applied to a software security system, see (Chapman, Hall 2002).

Praxis has migrated the principles of CbyC to other engineering disciplines, including requirements definition and systems engineering. Latterly, Praxis led a research project for the MoD known as SafSec into the definition of a methodology to exploit synergy between safety and security engineering processes for advanced avionic architectures. This research defined the term *dependable* as embodying both the safety and security properties of a system. The research culminated in the production of the SafSec Standard and Guidance documents (Praxis 2005a, Praxis 2005b) that define a common approach to the development of the dependability aspects of a system, together with its *dependability case* to support safety certification and security accreditation in an integrated manner, and to the highest levels of integrity.

Praxis continued to apply the principles of CbyC to the evolution of the SafSec methodology in the context of integrated safety and security engineering lifecycle processes. These principles include:

- use of a precise, unambiguous notation at all times;
- use of small, verifiable steps in deriving the output of a process from its predecessor, that also provides detailed step-wise traceability;
- use of verifiable contracts to define component interfaces and to assure composition;
- tackling the hard issues at the start of the lifecycle.

This paper outlines the realisation of CbyC principles within the SafSec definition, and also expands on one key aspect – the *Module Boundary Contract* – that is a critical component in establishing the dependability of a system composed of many disparate parts – providing **dependability-by-contract**.

2 Dependability Goals

Since dependability-by-contract is a method of establishing demonstrable safety and security of a system using contract-based composition of its parts, then the main goals for dependability-by-contract are:

- authoring a single Dependability Case (addressing both safety and security aspects);
- promoting modular certification;
- easing the uptake of Integrated Modular Avionics on platforms;
- enabling generic and legacy component re-use;
- accommodating COTS components;
- bounding the impact of change; and,
- reducing the costs of system safety certification and security accreditation.

Dependability can be demonstrated through the production of a compelling argument with supporting evidence for both the safety and security of a system - the Dependability Case. The Dependability Case argues the achievement of an acceptably safe and secure system, supported by sufficient *evidence* for the claimed level of assurance. The evidence can be informal or formal, although in the spirit of CbyC, formality is preferred, and strongly recommended for higher assurance levels.

The evidence within the Dependability Case can support the safety and/or security perspectives. The more evidence that can support both perspectives, the greater the savings in its production and hence lower overall costs and project risk.

In authoring an acceptable Dependability Case for a large or complex system, the project risk of certification/accreditation failure can be reduced by taking a modular approach. Modular certification entails achieving certifiability of parts of the system separately and then composing the certifiability of the parts into the certification of the whole system. There is of course a crucial activity in composing the modules that involves validating all assumptions and dependencies, but overall, this approach eases the re-use of legacy components, the accommodation of COTS, and can reduce the problems associated with obsolescence and re-certification after change.

A modular approach is particularly suited to Integrated Modular Avionics (UK MoD 2005a) where the sub-system boundaries are required to be clearly defined, and the modules are designed to be interchangeable. These modules, or other generic components, can be accompanied by either a generic or partial Module Dependability Case which can subsequently be completed when the module is integrated into the system, and when the full system context is known.

A dependability-by-contract approach that has been articulated in a Module Dependability Case bounds the impact of change, and therefore re-certification costs, when elements of the system are upgraded. Defining the system in a modular fashion, with the use of contracts at each module boundary, is a means of more clearly designing a system of systems, and is much preferred to a monolithic platform approach.

A further goal for dependability-by-contract is the unification of the safety and security process leading to a single risk management approach, as defined in the SafSec methodology. Combining the safety and security perspectives of a system using the SafSec methodology ensures that conflicts are addressed earlier, gaps are more easily identified and risks/costs are lower across the whole lifecycle. The greatest benefit lies at the programme or project level where the assessment of costs and timescales associated with dependability is clearer, and there is a reduction in both technical and project risk due to the breakdown of the safety and security silos.

3 Dependability Lifecycle

The dependability lifecycle processes may be summarised as follows:

- identification of the safety hazards and security vulnerabilities, known as *losses* within the SafSec definition;
- assessment of the *causes* of the losses, and their *consequences*;
- determination of the level of safety/security *risk* associated with each loss;
- determination of the measures to reduce each risk to a tolerable or acceptable level, and the required level of assurance;

- specification of *dependability properties* to satisfy those risk mitigation measures;
- architectural design of the *modular* aspects of the system, and apportionment of the dependability properties to the modules;
- definition of each module;
- design and implementation of each module;
- development of *arguments* and supporting *evidence* that the module architecture meets its dependability specification to the required assurance level.

This is illustrated in Figure 1.

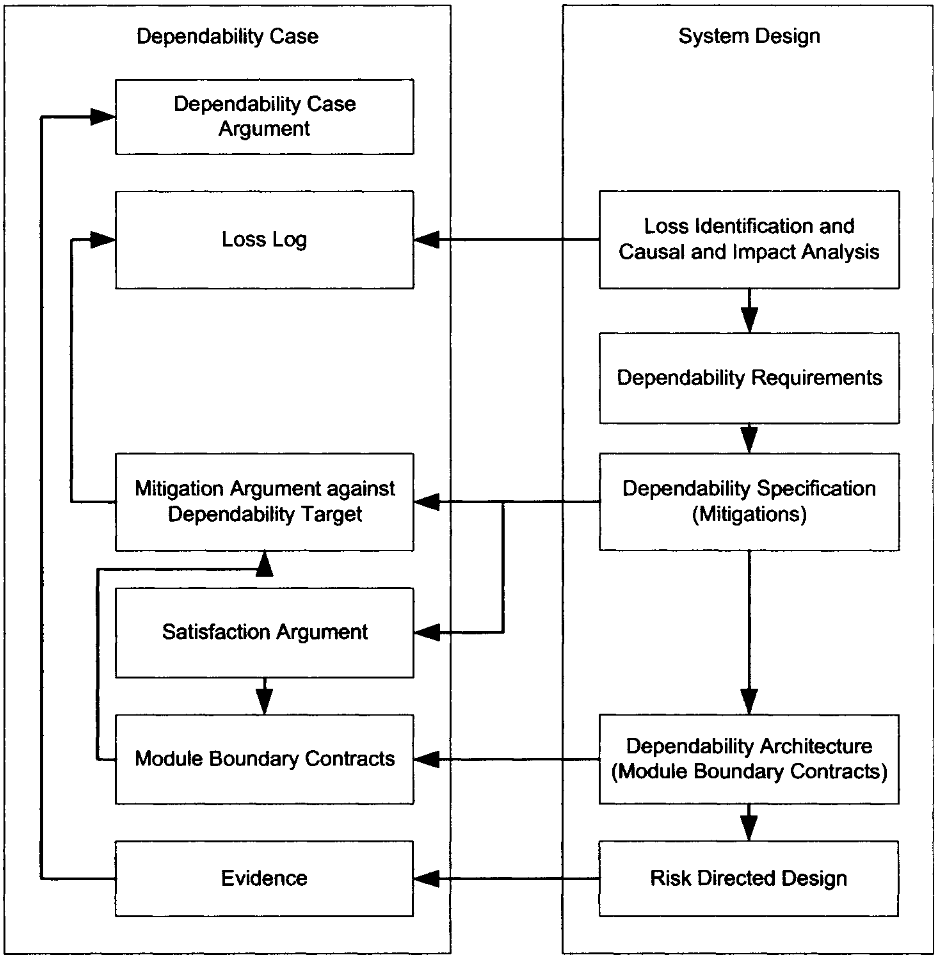

Dependability Case
Dependability Case Argument
Loss Log
Mitigation Argument against Dependability Target
Satisfaction Argument
Module Boundary Contracts
Evidence
System Design
Loss Identification and Causal and Impact Analysis
Dependability Requirements
Dependability Specification (Mitigations)
Dependability Architecture (Module Boundary Contracts)
Risk Directed Design

Figure 1: Dependability Lifecycle aligned with System Design Lifecycle

3.1 Dependability Assessment

The initial key process that is defined by the SafSec methodology is *Unified Risk Management* (URM). This process addresses the first four stages in the dependability lifecycle in the bulleted list above, covering loss identification, risk assessment, and risk mitigation.

The URM approach will firstly establish the boundary for (each part of) the system that is independently analysed. The boundary can be at a module, subsystem, system or platform level, but it must be clearly defined and understood. When decomposing a complex system into modules, it is desirable to consider what aspects of the system are most likely to be upgraded, the integrity requirements of differing components, the coherence and cohesion of components (particularly with respect to software), and the scope of supply and possible Intellectual Property Rights (IPR) issues.

With a clearly defined boundary the hazards and vulnerabilities can be identified through the use of traditional safety and security techniques. SafSec advocates concurrent execution of these techniques, so as to involve both the safety and security stakeholders. This provides the opportunity to *tackle hard issues first*, especially resolution of conflicts between safety and security needs.

The identification of the hazards and vulnerabilities leads to the definition of the *losses* - undesirable states at the defined boundary. The risks associated with the losses are managed in a unified process in the SafSec methodology.

In order to determine if a loss is relevant to the required dependability of the system, a *risk assessment* will be carried out for each loss. This entails undertaking both causal and impact analysis, to determine the likelihood and severity of each loss. These can be defined in either a qualitative or quantitative manner.

Each assessed risk is compared to an agreed dependability target for the loss. If the risk exceeds the target then mitigation is required to lower the risk. In determining suitable mitigation the cost and practicality of that mitigation is also a consideration. In some cases, the cost or practicality of achieving an acceptable level of risk may be prohibitive, and achievement of a *tolerable* level may be agreed to be sufficient, as defined by the *ALARP* principle (As Low As Reasonably Practicable) (HSE 1988).

Once the mitigations are agreed, and are traceable to the relevant losses, then these form the set of dependability requirements.

3.2 Dependability Specification

Having established the dependability requirements of the system, the SafSec methodology requires the development of a specification of the dependability properties that satisfy these requirements. This specification is defined by a set of Dependability Specifications (DSs) for the system.

The application of CbyC to this process strongly encourages use of a formal notation to express the set of dependability specifications. The use of a formal notation enables the specifications to be expressed unambiguously and to be proven to be complete and non-contradictory. This is in marked contrast to the common style of expressing system safety and system security requirements in English, where incompleteness, ambiguity and contradiction are much harder to eradicate, and are a frequent source of problems later in the life-cycle. The level of project and technical risk associated with sole use of imprecise notations such as English text is sufficiently great and well known that it is remarkable that formalism is not used more extensively. Perhaps it is just a fear that formalism is "too hard"?

In SafSec, each DS is assigned one or more Assurance Requirements (AR). The AR is the level of confidence in the predicates within the specification that must be assured by the implementation. These ARs are akin to the integrity levels and assurance levels that are defined by international safety and security standards. Since a DS may contribute to both safety and security assurance in varying degrees, it may be tagged with more than one AR, for example "Safety - High level of confidence (UK MoD 2005b); Security - EAL4 (ISO15408 1999)".

Use of a formal notation for the specification, coupled with assurance requirements, simplifies the development of correct mitigation arguments that demonstrate that the specification provides the required mitigations for the identified losses by meeting the dependability requirements. Praxis has found that use of the Z Notation (Spivey 1992) to express the specification formally is appropriate. However Z does not have any means to express concurrency, and hence the Z specification is generally complemented by a concurrency specification that embodies performance and temporal ordering predicates. In addition, a narrative in English is often provided to assist comprehension for those not fluent in formal notations, although the Z is always the definitive version and is used in the proof of completeness, correctness and lack of ambiguity.

Praxis has developed such formal specifications in several projects and the value of this approach has been apparent especially in exceptionally-low defect rates. The application of a formal specification to define the integrated safety and security properties introduces powerful rigour to the content of the dependability case, and is the dependability contract between the implementation and the safety and security requirements.

3.3 Dependability Architecture

The first step in the design and implementation of the dependability specification is the establishment of the dependability architecture, as part of the definition of the system architecture as a whole. This approach ensures that consideration of how to realise the required dependability is integrated with consideration of all other aspects of the architecture of the system, rather than being retro-fitted. It is consistent with the CbyC principle of "*tackling the hard issues first*".

Dependable systems are of course composed of many disparate components, including hardware, software, human factors, operational procedures etc. In addition, the components may be modifications of a legacy version, or COTS, or supplied by various different suppliers. The SafSec methodology uses the term *Module* to define all such components.

The dependability architecture is established by defining a hierarchy of interdependent modules and then apportioning the DSs with their Assurance Requirements to the modules of the hierarchy. In order to ensure that the DSs are at the correct level of information abstraction for the module to which they are apportioned, the DSs are themselves refined to the appropriate level of detail. For example, the Z specification for a system-level property is refined into an equivalent set of Z specifications for the various contributions to the system-level property that each component of the dependability architecture must make. The use of formal notations supported by proof tools allows such refinement to be achieved rigorously, whilst the addition of accompanying English narrative dispels the reader's fear of not being able to comprehend a cryptic set of symbols.

The final step in the dependability architecture is to define the dependability interface for each module. In this context, one of the biggest issues is that of composition of modules in a deterministic manner such that the required level of confidence in the establishment of the dependability properties of the whole can be demonstrated. This and other issues have been addressed by use of the *Module Boundary Contract* as the dependability interface for each module.

3.4 Module Boundary Contracts

3.4.1 Goals

The idea of using well-defined interfaces between modules is of course well established in almost all engineering disciplines. The Interface Control Document (ICD) is widely used to specify functional, operational and performance requirements at hardware and software module boundaries. In the case of software modules, these interfaces can be mapped for example into Ada (ISO 8652 1995) or SPARK (Barnes 2003) package specifications, which each boast comprehensive tool support for verifying compliance. Even within the safety domain, the idea of *Rely/Guarantee* contracts has been proposed for several years (Jones 1980) and is the subject of more recent development within the context of Integrated Modular Avionics and Advanced Avionic Architectures, for example (Rushby 2002, Blow, Cox, Liddell 2005). However the use of rigorous safety and security contracts cannot be described as mainstream at all.

From a CbyC viewpoint, the research aimed to develop a framework for a rigorous Module Boundary Contract with the following goals:

- to permit and encourage the use of a precise, unambiguous notation for the clauses of the contract;
- to support the use of strong tool-supported methods to validate the contracts against the system dependability specifications – not only how they meet the specification but also how they can fail to meet it;
- to encourage the definition of a module contract that is suitable for different levels of information abstraction, depending on whether it is for a high-level or

low-level module in the module hierarchy. This encourages taking small steps in the refinement of the dependability properties down the hierarchy;

- to support the use of rigorous methods to verify the composition of contracts, especially with regard to inconsistencies in assumptions, limitations or dependencies;
- to assist in the retrospective development of a contract for a legacy or COTS component.

3.4.2 Content

The Module Boundary Contract provides an abstract definition of the verified dependability properties of a module, in a form that does not include any information on *how* the dependability properties are implemented. It contains at least five types of clause:

1 Guarantee clause - defines the dependability properties that the module *guarantees* to hold true.
2 Rely clause - defines the dependability properties that this module *relies* upon in order to achieve its guarantees, including the assurance requirement for each property.
3 Context clause - defines the assumptions relating to the *operational context* of the system components that contribute to the Guarantee and Rely clauses.
4 Assurance Requirement clause - defines the *level of confidence or assurance* that is claimed for the contract, usually based on safety and/or security standards.
5 Counter-evidence clause - defines the limitations that exist in the contract. This clause may reference known defects in the implementation, as well as the residual risk in meeting its guarantees that has been assumed to be acceptable.

It should be noted that this is not an exhaustive list – further system-specific clauses may be added. The intent is that this list provides the minimum set of information for validation of a contract against the set of dependability specifications that are apportioned to it, and for verification of its composability with other modules.

The Guarantee clause specifies the *unconditional* responsibility that the module takes for reducing risks or supplying dependability-related services. The other clauses specify the *conditions* under which the Guarantee clause is valid.

The Rely clause defines all dependencies on safety and security properties of interfaced modules. This is the traditional definition, but is extended here to include the required level of assurance in the dependent property.

The Context clause is an important extension to the traditional Rely/Guarantee model. This clause defines any assumptions about the environment in which the module is used by any system in which it is deployed, that are needed by the Guarantee clauses. It addresses the *"it was never intended to be used in those conditions"* fault. The clause captures all operation contextual assumptions that must hold for the Guarantee clauses to be valid.

The Assurance Requirement (AR) clause is another important extension. It specifies the level of confidence in the Guarantee clauses that is claimed, or that has been independently assessed, based on the supporting verification evidence. Since a Rely clause entry also contains an assurance requirement, the composition of a Rely clause entry and a matching Guarantee clause entry is only valid if the assurance requirement associated with the Guarantee clause entry is sufficient.

The Counter Evidence clause is probably the most important extension. The requirement to address counter-evidence within a safety case has been introduced within interim Defence Standard 00-56 issue 3 (UK MoD 2005), and the identification of vulnerabilities is an established requirement within Common Criteria (ISO15408 1999) and other security standards.

The Counter Evidence clause may be viewed as an extension to the AR clause in that it defines the specific known limitations of the module. This may include built-in capacity or stress loading limits, and known faults/bugs, but only insofar as they relate to contributing to conditions on the validity of other clauses of the contract. The counter-evidence must also include the residual risk (effect and likelihood) of a failure or malfunction of the module to deliver the entries in the Guarantee clause. This includes the impact of failures of each dependent property stated in the Rely clause and each assumption in the Context clause, as well as the various types of internal failure. Note that this only addresses failures in the risk mitigation features that have previously been identified as being necessary by the earlier process that established the dependability requirements, based on the results of hazard/threat/vulnerability analyses. Assessment of the Section clause entries should result in re-visiting the earlier risk assessment to ensure that the mitigation strategy for the dependable system as a whole remains sufficient. It will be necessary to show in the dependability case that there is a compelling rebuttal for all residual risk of malfunction or failure of the dependability properties of any module.

3.4.3 Representation and Validation

The SafSec methodology does not prescribe any particular notation or representation for the clauses of the Module Boundary Contract. However the application of CbyC to the methodology strongly encourages a precise and unambiguous notation. If such a notation has been used for the apportioned dependability specifications, such as Z for example, then it is natural for this also to be used to define at least the Rely and Guarantee clauses within the module contract. As long as a similar level of abstraction is applied both to the module contract information and to the dependability specifications that are apportioned to it, the task of validating the contract (or a composition of sub-contracts) against the specification can make use of formality and support from proof tools.

3.4.4 COTS and Legacy Modules

Establishing sufficient confidence in the use of COTS products and legacy systems of "unknown pedigree" has been the bane of many a safety case and security argument. This is usually resolved at higher levels of integrity in the COTS case by vendors providing some form of certificate that the product has been independently audited against a recognised set of standards, and at a stated level of assurance, for example *Certification Packs* for COTS real-time operating systems. For legacy systems, and COTS products that do not have such accreditation, the case is usually based on sufficient "hours" of in-service history, or number of users, that have demonstrated a certain level of resilience and an acceptable number and severity of defects. However this is very much secondary, rather than primary, evidence.

The main issue is whether the evidence to support confidence in the dependability of such a component is valid and relevant to its deployment in the target system. For example, a general operating system may have been very reliable in a previous version of a system, but when new real-time safety-related constraints are introduced via an upgrade, the same operating system may be totally deficient in the additional services that are needed.

The application of Module Boundary Contracts to COTS and legacy systems can bring significant improvements in this area. The retrospective production of the contract, and its subsequent composition with other contracts of the system architecture, focuses on the applicability of the module to the dependability needs of its deployment environment, rather than as a free-standing entity or within a previous environment. For example, the following questions will naturally occur:

- Certain dependability specifications will have been apportioned to the COTS/legacy module. Can a contract be constructed such that a satisfaction argument can be made for the module guaranteeing to meet each apportioned dependability specification?
- What does the module Rely on in terms of external services, and how reliable do these services need to be?
- Is there any evidence that the module's guaranteed properties have been verified for use in the kind of operational environment in which it will be deployed?
- Have the specific services of the module that will be used for the dependability of the system passed accreditation to the required level of confidence? If not, is there any relevant in-service history for these specific services within a compatible operational environment, and if so, how much?
- Has it been possible to obtain all applicable counter-evidence regarding the required dependability properties of the module, for example, faults and limits, such that their impact can be assessed within the context of the target system?

It is important to note that the retrospectively-generated contract need only concern itself with the dependability properties that are required of it, and should be expressed at the right level of abstraction. One of the biggest challenges in this area is the establishment of a contract for a commercial operating system. For example, it is totally infeasible and impractical to attempt to produce a set of Guarantee clause entries for something like Windows XP if it were based on all of its API services - this would be prohibitively large and at totally the wrong level of abstraction. Instead, the clause entries should only state the dependability properties that correspond to the apportioned dependability specifications, within the context of the deployed system, for example – "the memory space of a process is protected from direct access by any other process at all times" or more formally:

$$\forall\, p_i, p_j : \text{PROCESS} \mid i \neq j \bullet \text{addressSpace}(p_i) \cap \text{addressSpace}(p_j) = \varnothing$$

More information on the construction of a module boundary contract for an operating system can be found in (Praxis 2005c).

3.4.5 Composition

A set of Module Boundary Contracts may be composed if they are consistent with each other. The SafSec Standard defines a minimum set of rules that must be satisfied for modules to be consistent. Use of a formal notation with tool support for at least the Rely and Guarantee clauses of the contracts eases verification of the consistency of sibling module contracts, and also that the parent contract is semantically equivalent to the union of the siblings.

However, the successful composition of Rely and Guarantee clause entries, including sufficiency of assurance requirement levels, must be supplemented by successful resolution of Context and Counter Evidence clauses. One of the most frequent sources of failure is "adverse unintended interaction" between components that each function adequately in "intended context". For hardware components, simple physical constraints such as "*you can't fit a square peg into a round hole*" can be applied, but more subtle interactions need expert eyes or the bitter light of experience, such as a rupturing aircraft tyre being able to penetrate adjacent fuel tanks. The problem is much more acute with software, especially since physically separated programs in federated systems are now being replaced by integrated programs executing in the same physical address space. This requires underlying support for partitioning of shared resources such as memory space, CPU time, and communication buses, such that each module can assume that it operates in a context where its execution environment cannot be tampered with, even in the presence of other malfunctioning modules.

In order for module contract composition to operate convincingly in practice, two principles of CbyC must be applied to the modules– ***strong cohesion and weak coupling***. It has long been understood that systems whose components have strong coupling and hence interactive complexity are much more susceptible to failure (Perrow 1984) and any safety or security assessor that is presented with a dependability case for a system containing a complex web of dependability interactions faces a daunting task in pronouncing the system as safe and secure.

This problem can be anticipated, and rectified early, by the examination of the interactions, and complexity of composition, of the module contracts. Any system that has many modules with large numbers of Rely clause dependency entries, Context clause assumptions, and Counter-Evidence limitations should be re-examined at the dependability architecture stage to see whether the dependability specifications can be met by a more cohesive structure.

By tackling this at the early dependability architecture stage, the realisation of the goal of dependability-by-contract, and the production of a compelling satisfaction argument that the module hierarchy meets its dependability specification, may be achieved by correctness of construction, rather than by excesses of complexity.

3.5 Dependability Design

As we have already described, the CbyC approach as applied to SafSec requires an integrated approach to addressing safety and security requirements with those of functionality, performance etc. We have seen that the dependability architecture should be developed as a key part of overall system architecture construction.

The step-wise refinement process of CbyC extends this integrated approach to systems and module design. One of the key processes defined by the SafSec methodology is *Risk-Directed Design*. The key point is that the decisions taken during design must be influenced by the required dependability to the same extent, if not more than, the required functionality and performance. For example, the choice of an operating system (O/S) can only be made when all required dependability properties of the O/S are known, and the required level of confidence in them has been established.

The design of the dependability characteristics of each module is captured in its dependability arguments. These arguments provide the backbone of the modular dependability case that the module implements its contract. CbyC encourages the use of a precise and verifiable notation for such arguments – for software modules, SPARK has been found to offer sufficient rigour and expressive power for design purposes (Amey 2001). The other advantage of using SPARK as a design language for software modules is that there is an automatic tool-based verification route to ensure that the code conforms to the design. Additionally, SPARK has recently been extended to provide support for assigning assurance levels to specific data and code objects (Amey, Chapman, White 2005).

Apart from SPARK, there is increasing use of graphical notations such as Goal Structuring Notation (GSN) (Kelly 1998) to express safety arguments – indeed GSN could also be used to express security arguments (Cockram 2006).

When using a notation such as GSN for the dependability argument, it may be necessary to extend the notation and to use conventions to circumvent the lack of formalism, and also deficiencies, such as the lack of a construct to specify counter-evidence, as identified for example in (Cockram 2006). Alternatively, GSN can be used merely as a narrative for a formal design expressed in a rigorous notation. Ultimately a complete, consistent and compelling dependability argument, supported by a body of evidence that is sufficient to meet the claimed Assurance Requirement, must be created for each module to show that its design and implementation satisfies its module boundary contract.

Generation of a dependability argument for a COTS component is a challenging prospect due to the "black box" nature of the product and the common lack of knowledge of its internal development process and reliability history. It is generally necessary to build the argument on the results of specific verification tests of the dependability properties of the COTS product that are required (those in its Guarantee clause). This primary evidence would carry more weight than any more general product-based evidence supplied by the vendor, based on development process, or by in-service history records based on a large install base.

3.6 Modular Certification and Dependability Cases

The final key process defined by the SafSec methodology is *Modular Certification.* Modular certification in this context applies both to safety certification and to security accreditation. The crux of modular certification is the validation of the module boundary contract, and the verification that the module dependability argument and supporting evidence are sufficient to substantiate the contract.

Certification of the module through validation of the module boundary contract is realised via a Modular Dependability Case. This can be developed in a number of ways, including a text editor, an html/web page package, and bespoke argument editors. Regardless of the method used, the property that is vital is that the dependability of the module, as defined by its module boundary contract, must be clearly established, traceable and coherent.

The SafSec methodology promotes the use of electronic web page dependability cases as they provide many advantages over the other forms, for example:

- hyper-linking enables traceability from elements of the module contract to their justification arguments and evidence;
- a navigation bar provides permanent access to the breadth of the contract and argument;
- tool-supported verification of module contract composition is possible;
- only the information that is pertinent can be made viewable when complying with formats mandated by standards, thereby removing nugatory reviewing effort;
- the ability to view and maintain the individual module cases remains, after the module dependability cases are composed to form a higher-level system dependability case, bounding the impact of change and re-certification costs.

Achieving modular certification can seem a misnomer, particularly within the safety community, due to the traditional means of certification being applicable only to platforms. However, as systems become larger, more intricate and include COTS, legacy, bespoke and generic components, the benefits of a modular approach are unquestioned, and the use of verifiable module composition becomes a prerequisite to the achievement of a dependable and maintainable system (Jones, Johnson 2002).

4 Practical Examples

The SafSec methodology has been exercised on two large case studies and several smaller ones. All case studies provided vital information that not only refined the definition of the methodology but also enabled authoring of a partial module boundary contract and dependability case.

One of the larger case studies was undertaken with a major MoD prime contractor to plan how the SafSec methodology could be used on a tactical processor within a military avionic system. As the functionality of the tactical processor is expected to change, a major goal was to de-risk re-certification of the upgraded system in a cost effective manner.

The system architecture was decomposed into a number of modules, and initial module boundary contracts were developed. At this early stage, the context clauses were skeletal and the counter-evidence clauses were not developed, so the main content was the Rely and Guarantee clauses. The initial versions of the clauses were expressed in English rather than using a formal notation. Extracts from a sample of these contracts, generalised for information confidentiality, are provided in Table 1.

Guarantee and AR	Rely and AR	Context
The environment shall be protected by physical security to prevent removal of the tactical processor AR = Serious breaches < 1 per year	-	The physical environment is an MoD facility covered by all the usual policies and procedures that MoD facilities must follow.
The display unit shall guarantee to display data it receives correctly AR = Fails < 1x10^{-4} per use	The tactical processor must provide processed position data when requested. AR = SIL 2	The display unit will be used in an aircraft cockpit.
The mission system shall guarantee to provide position data in XYZ format AR = SIL2	The operating system must transmit message data correctly when requested AR = SIL2	The mission system will be used in an aircraft
The tactical processor shall guarantee to provide processed position data when requested AR = SIL2	The environment must be protected by physical security to prevent physical removal of the tactical processor. AR = Serious breaches < 2 per year	The tactical processor will be used in an aircraft
The tactical processor shall guarantee to process position data in XYZ format correctly AR = SIL2	The mission system must provide position data in XYZ format. AR = SIL2	
	The operating system must transmit up to Xbytes/sec message data correctly when requested. AR = SIL2	Message transfer will be via a LAN with capacity Ybytes/sec
The operating system shall guarantee to transmit message data correctly when requested AR = SIL3	A LAN connecting the message sender and receiver, with sufficient capacity for the required message transfer rate, must be available	-

Table 1: Sample Contract extracts for a tactical processor within an avionic system.

In Table 1, the boundaries between the module contracts are illustrated using thick borders. The contracts show some of the relationships that exist between the Rely and Guarantee clauses, and their assurance levels, for example:

- The display unit relies (at SIL 2) on the tactical processor to provide correctly processed position data when requested.
- The tactical processor guarantees (at SIL 2) to supply correctly processed data, but relies (at SIL 2) on the mission system to provide position data in a given format, and on the operating system message services (at SIL 2) to transmit this data correctly.

Dependencies between modules can exist at differing levels within the module hierarchy, as well as between sibling modules.

The case studies made use of electronic web page technology for the dependability case based on the Praxis tool *eSafetyCase* (Cockram, Lockwood 2002). It was found that this technology eased the management and validation of the information in the clauses of the Module Boundary Contracts, and provided the means for configuration management to meet an acceptable standard. For example, a hyperlinked diagram could both enable, and appropriately restrict, access to parts of the dependability case. Similarly hyperlinked contract clauses were found to provide a means of navigating the argument of how system-level module composition satisfies the contract between the dependability specification of the system and its dependability requirements.

5 Conclusions

Correctness-by-Construction advocates that very low defect rates, high quality, and resilience to change, are best realised for systems and software by constructing correctly right from the start, supported by robust methods to ensure validity of the correctness claim. Dependability-by-contract applies this principle to the safety and security attributes of the system, in order to:

- increase confidence in the validity of the Dependability Case that addresses both safety and security aspects;
- increase confidence in the dependability of a deployed system that is composed of many disparate components, including COTS and re-used legacy systems;
- ease the uptake of advanced avionic architectures, including Integrated Modular Avionics, in platforms;
- bound the impact of change and upgrades due to obsolescence; and,
- reduce the lifecycle costs and technical and project risks associated with system safety (re-)certification and security (re-)accreditation.

The use of rigorously-defined Module Boundary Contracts is a key element in dependability-by-contract, which offers particularly strong solutions to the problems associated with obsolescence and upgrade, as expressed in (Tudor 2002).

References

Amey P (2001). A Language for Systems Not Just Software. SigAda 2001, Ada Letters Volume XXI, Num 4, December 2001, ACM Inc, NY, NY, 2001

Amey P, Chapman R, White N (2005). Smart Certification of Mixed Criticality Systems. AdaEurope 2005, Springer-Verlag London Ltd, ISBN 978-3-540-26286-2, 2005

Barnes J with Praxis Critical Systems (2003). High Integrity Software The *SPARK* Approach to Safety and Security. Addison-Wesley, London, ISBN 0-321-13616-0, 2003

Blow J, Cox A, Liddell P (2005). Modular Certification of Integrated Modular Systems. Safety Critical Systems Symposium 2005, Springer-Verlag London Ltd, London 2005

Chapman R, Hall A, (2002). Correctness by Construction: Developing a Commercial Secure System. IEEE Software, 2002, pp 18-25

Cockram T (2006). Is this the right room for an Argument – improving arguments for safety and security. ESREL 2006, Safety and Reliability for Managing Risk – Guedes Soares and Zio (eds), Taylor and Francis Group London, 2006, ISBN 0-415-41620-5.

Cockram T, Lockwood B, (2002) Electronic Safety Cases: Challenges and Opportunities. Safety Critical Systems Symposium 2003, Springer-Verlag Lonon Ltd, London, 2002

Croxford M, Chapman R (2005) Correctness by Construction: A Manifesto for High Integrity Software. In CrossTalk Vol 18 No 12 December 2005

HSE (1988). The Tolerability of Risk from Nuclear Power Stations. Health and Safety Executive, 1988

ISO8652 (1995). Ada 95 Reference Manual International Standard ANSI/ISO/IEC-8652:1995. International Standards Organisation, www.iso.org, 1995

ISO15408 (1999). Common Criteria for Information Technology Security Evaluation ISO/IEC-15408:1999. International Standards Organisation,Version 2.1 www.iso.org, 1999

Jones C B (1980). Software Development: A rigorous approach. Prentice Hall International , ISBN 0-13-821884-6, 1980

Perrow C (1984). Normal Accidents: Living with High Risk Technologies. Basic Books, New York, NY, 1984

Rushby J (2002). Modular Certification. NASA Contractor Report NASA/CR-2002-212130, NASA Langley Research Center, December 2002

Jones J, Johnson M (2002). Affordable Avionics – the MoD Strategy. UK MoD 2002

Kelly T. (1998). Arguing Safety – A Systematic Approach to Managing Safety Cases. DPhil Thesis, Department of Computer Science, University of York, 1998

Praxis (2005a). SafSec Standard. Praxis High Integrity Systems Ltd, 2005

Praxis (2005b). SafSec Guidance. Praxis High Integrity Systems Ltd. 2005

Praxis (2005c). Practical Guide to Certification and Re-certification of AAvA Software Elements COTS RTOS. Praxis High Integrity Systems Ltd, http://www.ams.mod.uk/ams/content/docs/rtos_guid.pdf 2005

Spivey J. M. (1992). The Z Notation: A Reference Manual. Prentice Hall International (UK) Ltd, 1992

Tudor N, (2002). Realising Integrated Modular Avionics in Military Aircraft. UK MoD 2002

UK MoD (2005a). Defence Standards Series 00-74, 00-75, 00-76, 00-77, 00-78. UK Ministry of Defence Directorate of Standardisation, 2005

UK MoD (2005b), Defence Standard 00-56 Safety Management Requirements for Defence Systems Part 1 and 2. UK Ministry of Defence Directorate of Standardisation, 2005

Acknowledgements

We are grateful to Mark Suitters FBG-3d DPA, for his support and funding for the SafSec work, and to all the SafSec stakeholders for their invaluable contributions.

Demonstrating Safety

Achieving Integrated Process and Product Safety Arguments

Ibrahim Habli and Tim Kelly
Department of Computer Science, University of York,
York, United Kingdom

Abstract

Process-based certification standards such as IEC 61508 and DO-178B are often criticised for being highly prescriptive and impeding the adoption of new and novel methods and techniques. Rather than arguing safety based on compliance with a prescribed and fixed process, product-based certification standards require the submission of a well structured and reasoned safety case. Ideally, the safety case presents an argument that justifies the acceptability of safety based on product-specific and targeted evidence. However, the role of process assurance should not be underestimated even in product arguments. Lack of process assurance can undermine even the seemingly strongest product safety evidence. However, unlike the SIL-based process arguments, the process argument of the type we suggest are targeted and assured against specific safety case claims. In this way, a close association between product and process safety arguments can be carefully maintained. This paper shows how integrated process and product safety arguments can be achieved using the modular features of the Goal Structuring Notation (GSN).

1 Introduction

The assurance of safety-critical systems is typically demonstrated against certification guidelines. Currently, there are two different approaches to safety certification: process-based and product-based. In process-based certification standards, developers demonstrate that the system is acceptably safe by applying a set of techniques and methods that the standards associate with a specific safety integrity level or risk classification. Process-based certification standards are often criticised for being highly prescriptive and impeding the adoption of new and novel methods and techniques (McDermid 2001). The fundamental limitation of process-based certification lies in the observation that good tools, techniques and methods do not necessarily lead to the achievement of a specific level of integrity. The correlation between the prescribed techniques and the failure rate of the system is often infeasible to justify (Redmill 2000).

For example, the certification assessment of civil airborne software is performed against predefined and fixed process objectives and activities and is not driven by the consideration of the *specific* safety requirements and hazard and risk analysis of the software. To demonstrate the certifiability of such software, developers submit plans, such as software development and verification plans, that show that the development and verification of the software have been performed as prescribed in the certification guidelines, namely RTCA/DO-178B (EUROCAE 1994). Any deviation or alternative means for compliance should be justified in the Plan for Software Aspects of Certification (PSAC) and Software Accomplishment Summary (SAS). As a result, the norm in the development of civil airborne software is to apply the methods and techniques as prescribed in the certification guidelines, regardless of the specific safety requirements of the software, and hence avoid the challenge of justifying any new technique.

Rather than arguing safety based on compliance with prescribed methods and techniques, product-based certification standards require the submission of a well structured and a reasoned safety case. Ideally, the safety case presents an argument that justifies the acceptability of safety based on product-specific and targeted evidence. A safety case is defined in the UK Defence Standard 00-56, Issue 3, as (UK Ministry of Defence 2004):

> *"A structured argument, supported by a body of evidence that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given operating environment"*

Process-based standards such as DO-178B and IEC 61508 may implicitly provide an argument, supported by evidence. However, the fundamental limitation is that this argument and the items of evidence are prescribed. Developers end up arguing and assuring the satisfaction of the certification requirements and objectives rather than justifying the validation and satisfaction of the safety requirements.

This paper does not, however, suggest that the process in safety-critical system development is insignificant. The criticisms above tackle unfocused processes with their indeterminable relationship to claims of product integrity. The role of the process assurance is still important even in product-based arguments. Lack of assurance about the provenance of evidence in product arguments can undermine even the seemingly strongest product argument. The process of the type we suggest are targeted and consequently justified and assured in the context of the product safety argument. In this way, a close association between product and process safety arguments can be carefully maintained.

The rest of this paper is structured as follows. Section 2 explains the role of process evidence in product-based certification. A product argument is then presented in Section 3. Section 4 shows how confidence in a product argument can be weakened by lack of process assurance and how this can be addressed by integrated product and process arguments. Explicit arguments about the trustworthiness of the process and the relationship between the integrity of the product and the integrity of the process are discussed in Section 5. This paper concludes with a summary in Section 6.

2 Role of Process Evidence in Safety Arguments

The suggestion that process evidence is needed in a product-based safety case is not new. The role of process evidence in safety arguments has been emphasised in work by Weaver (Weaver 2003) and Caseley (Caseley, Tudor and O'Halloran 2003) and in certification standards such as the UK Defence Standard 00-56 (Issue 3). Def Stan 00-56 expresses the role of the process in the safety case as follows (UK Ministry of Defence 2004):

> *"The Safety Case contains all necessary information to enable the safety of the system to be assessed. It will contain an accumulation of both product and process documentation"*

Particularly, Def Stan 00-56 specifies three categories of evidence in a safety argument, namely:

- **Direct evidence:** Evidence generated from analysis, review and demonstration (e.g. testing)
- **Process evidence:** Evidence appealing to "good practice in development, maintenance and operation"
- **Qualitative evidence:** Evidence of good engineering judgment and design

Listing evidence types based on the precedence above and limiting 'directness' to analysis, review and demonstration in Def Stan 00-56 may underestimate the role of process assurance in establishing confidence in the pieces of evidence in the safety argument. Directness is a relative attribute that depends on the claim, i.e. if the claim is about the process, the process evidence should be direct.

Similarly, Caseley et al identify four types of evidence as a basis for an evidence-based approach to the assurance of software safety (Caseley, Tudor and O'Halloran 2003):

- **Process evidence:** Qualitative indicator, based on factors such as quality management systems and staff competency. Process evidence is only supportive. It is not the primary evidence as processes cannot be directly related to failure rates.
- **Historic evidence:** Quantitative indicator, based on failure data or reliability models. However, the applicability of historic data to software has been limited due to the difficulty of capturing accurate characteristics of the software environment.
- **Test evidence:** Prevalent, focused and effective in systems safety development, taking two forms: dynamic and static. However, testing is normally costly and limited in the sense that unlike mathematical proofs, it cannot demonstrate freedom of error (i.e. when exhausted testing is not possible).

- **Proof evidence:** Best form of verification, based generally on formal methods. It is typically complementary to testing because of the infeasibility of verifying the entire system formally.

Caseley et al argue that none of the above-mentioned types of evidence alone can provide conclusive proof that the system is acceptably safe. Each has its weaknesses and strengths. A safety argument is best constructed based on diverse and complementary types of evidence.

Unlike, yet complementary to, the abovementioned approaches to evidence, i.e. defining the types of evidence that can be used to support a safety argument, Weaver in (Weaver 2003) goes further by identifying the items of evidence that assure a safety argument, namely:

- **Relevance:** Evidence directly related to, and covers sufficiently, the requirements
- **Independence:** Diverse pieces of evidence for satisfying the requirements, e.g. conceptually and mechanistically dissimilar pieces of evidence
- **Trustworthiness:** "Expression of the process evidence related to generating the evidence" in terms of process factors such as tool integrity and competency of personnel

Apart from Weaver's reference to the role of process in showing the trustworthiness of the evidence, process evidence, as approached in product-based certification, is disjoint from the specific claims and integrity of the system. Appealing to good practice in development and verification provides some level of process assurance. However, process arguments and evidence are not limited to such a secondary role. Keeping the process argument implicit may increase the risk of producing untrustworthy evidence, even though such evidence may seem to be relevant and independent.

Developers should explicitly provide a process argument demonstrating '*how*' the development and assessment process targets the production of trustworthy product evidence, and hence our suggestion that the process argument is inseparable from the product argument. Therefore, similar to the product evidence, process evidence must be directly related to the claims and product evidence in the safety case.

The suggestion that direct evidence is limited to product evidence such as testing and analysis (e.g. as stated in Def Stan 00-56) underestimates the contribution of process arguments in assuring the trustworthiness of evidence in a safety argument. A product argument, such as that shown in Figure 1, represents an example of an argument where the trustworthiness of the process behind the generation of the product evidence is not fully demonstrated. Referring to a process compliant with ISO 9001 or CMMI provides a certain level of confidence about the 'quality' of the process, i.e. general quality issues such as consistent documentation and controlled configuration. However, it reveals little about the suitability of the process in targeting claims about the assurance of specific process elements such as the trustworthiness of the testing or formal analysis process. Many assumptions, dependencies and rationale of the process are hidden behind the claim of compliance with ISO 9001 or CMMI, which may reveal little about the relationship between the integrity of the product and the integrity of the process.

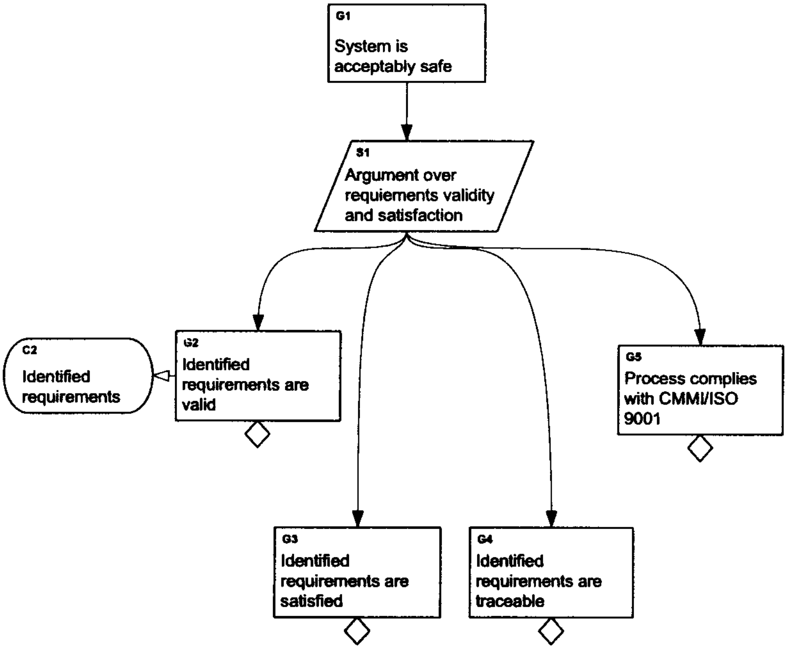

G1
System is acceptably safe
S1
Argument over requiements validity and satisfaction
C2
Identified requirements
G2
Identified requirements are valid
G3
Identified requirements are satisfied
G4
Identified requirements are traceable
G5
Process complies with CMMI/ISO 9001

Figure 1. Unfocused Process Argument

In summary, the balance and association between arguing about the product and the process should be carefully managed. A process argument should be explicitly articulated. Otherwise confidence in the product evidence may be weakened by the implicit assumptions of the process.

3 An Example Product Argument

Figure 2 shows an example goal structure of a product argument. The goal structure is for an argument for the safe operation of a sheet metal press. This operation is controlled by an operator via a simple control system based on a Programmable Logic Controller (PLC). In this structure, as in most, there exist 'top level' goals – statements that the goal structure is designed to support. In this case, "C/S (Control System) Logic is fault free", is the (singular) top level goal. Beneath the top level goal or goals, the structure is broken down into sub-goals, either directly or, as in this case, indirectly through a strategy. The two argument strategies put forward as a means of addressing the top level goal in Figure 3 are "Argument by satisfaction of all C/S (Control System) safety requirements", and, "Argument by omission of all identified software hazards". These strategies are then substantiated by five sub-goals. At some stage in a goal structure, a goal statement is put forward that need not be broken down and can be clearly supported by reference to some evidence. In this case, the goal "Press controls being 'jammed on' will cause press to halt" is supported by direct reference to the solutions, "Black Box Test Results" and "C/S State Machine".

The argument in Figure 2 makes it clear how the safety requirements are achieved by the software-specific product evidence (solutions). Black box testing and state machine analysis provide explicit and independent evidence that is related to the software artefact rather than appealing to the quality of the development process.

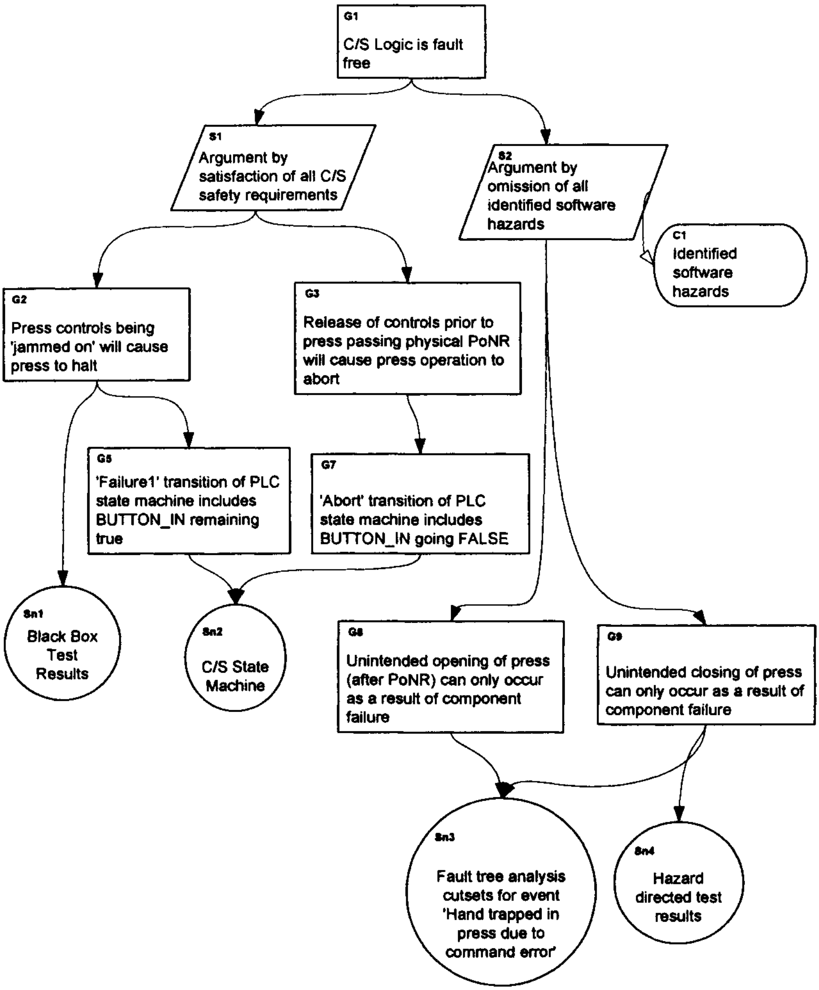

G1
C/S Logic is fault free
S1
Argument by satisfaction of all C/S safety requirements
S2
Argument by omission of all identified software hazards
C1
Identified software hazards
G2
Press controls being 'jammed on' will cause press to halt
G3
Release of controls prior to press passing physical PoNR will cause press operation to abort
G5
'Failure1' transition of PLC state machine includes BUTTON_IN remaining true
G7
'Abort' transition of PLC state machine includes BUTTON_IN going FALSE
Sn1
Black Box Test Results
Sn2
C/S State Machine
G8
Unintended opening of press (after PoNR) can only occur as a result of component failure
G9
Unintended closing of press can only occur as a result of component failure
Sn3
Fault tree analysis cutsets for event 'Hand trapped in press due to command error'
Sn4
Hazard directed test results

Figure 2. Product Argument

The evidence provided in the argument relies also on diverse solutions, hence avoiding common mode failures. Testing and analysis are dissimilar conceptually and mechanistically (providing the highest form of independence (Weaver 2003)). Conceptual diversity relies on two different underlying concepts while mechanistic diversity relies on two different applications of same underlying concepts. However, in the next section we show how the confidence in the argument gained by this apparently independent and direct product evidence can be undermined by lack of process assurance.

4 An Example Process Argument

The product argument depicted in the goal structure in Figure 2 lacks a clear reference to any process argument that addresses the trustworthiness of the provenance of the product evidence (i.e. black box testing and state machine analysis). Firstly, black box testing (*Sn1*) is an effective verification technique for showing the achievement of safety requirements. However, confidence in the black box testing depends on assuring the testing process.

For example, factors that need to be addressed by process evidence include issues such as:

- The testing team is independent from the design team
- The process of generating, executing and analysing test cases is carried out systematically and thoroughly
- The traceability between safety requirements and test cases is well established and documented.

Similarly, state machine analysis (*Sn2*) is a powerful formal method for specification and verification. Nevertheless, process information is required to reveal the mathematical competence of the verification engineers and their ability to demonstrate correspondence between the mathematical model and the software behaviour at run-time (Hall 1990). Mistakes can be made in formal proofs the same way that they can be made in coding. Therefore, the quality of the verification process by means of formal methods is as important as the deterministic results such methods produce.

To address the above limitation, we propose addressing process uncertainty through linking process arguments to the items of evidence used in the product safety argument. Such process arguments address issues of tool and method integrity, competency of personnel, and configuration management. The rest of this section elaborates on the use of modular GSN in linking process arguments to pieces of product evidence.

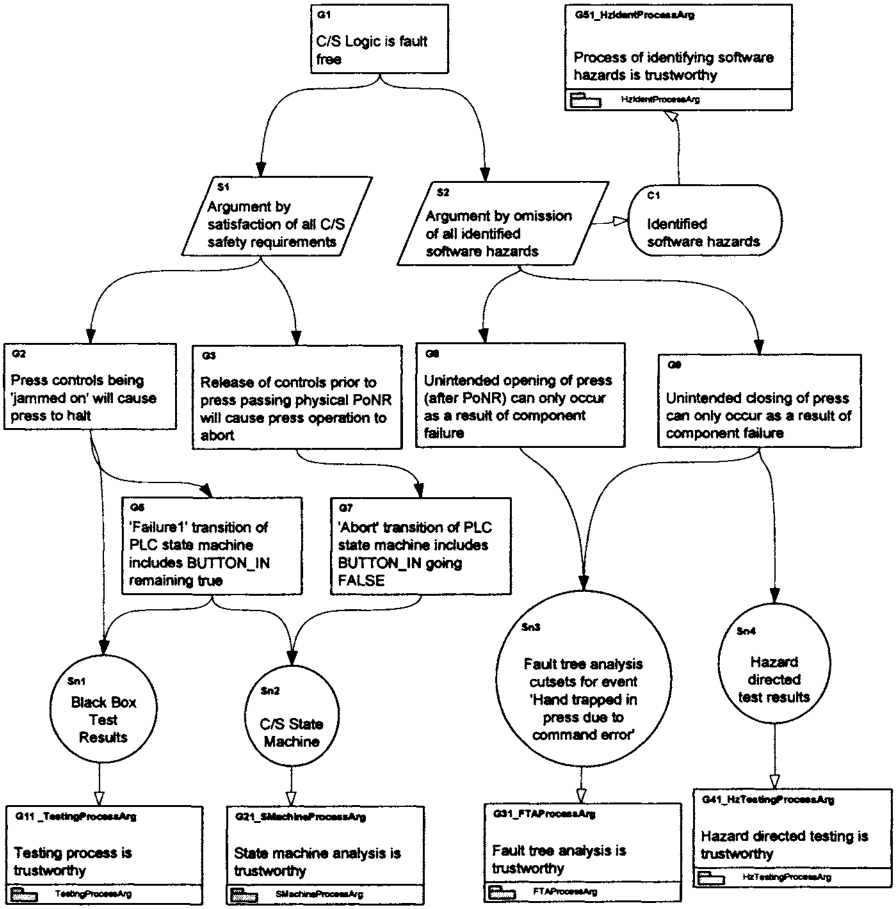

G1
C/S Logic is fault free
G51_HzIdentProcessArg
Process of identifying software hazards is trustworthy
HzIdentProcessArg
S1
Argument by satisfaction of all C/S safety requirements
S2
Argument by omission of all identified software hazards
C1
Identified software hazards
G2
Press controls being 'jammed on' will cause press to halt
G3
Release of controls prior to press passing physical PoNR will cause press operation to abort
G8
Unintended opening of press (after PoNR) can only occur as a result of component failure
G9
Unintended closing of press can only occur as a result of component failure
G6
'Failure1' transition of PLC state machine includes BUTTON_IN remaining true
G7
'Abort' transition of PLC state machine includes BUTTON_IN going FALSE
Sn1
Black Box Test Results
Sn2
C/S State Machine
Sn3
Fault tree analysis cutsets for event 'Hand trapped in press due to command error'
Sn4
Hazard directed test results
G11 _TestingProcessArg
Testing process is trustworthy
TestingProcessArg
G21_SMachineProcessArg
State machine analysis is trustworthy
SMachineProcessArg
G31_FTAProcessArg
Fault tree analysis is trustworthy
FTAProcessArg
G41_HzTestingProcessArg
Hazard directed testing is trustworthy
HzTestingProcessArg

Figure 3. Integrated Product and Process Argument

Figure 3 shows a modified version of the goal structure of the sheet metal press safety argument. This version uses an extension to GSN (Kelly 2001) – the 'Away' Goal (e.g. G11_TestingProcessArg and G21_SMachineProcessArg) to attach process arguments to the GSN solutions. Away Goals are used within the arguments to denote claims that must be supported but whose supporting arguments are located in another part of the safety case. Away Goals were developed to enable modular and compositional safety case construction.

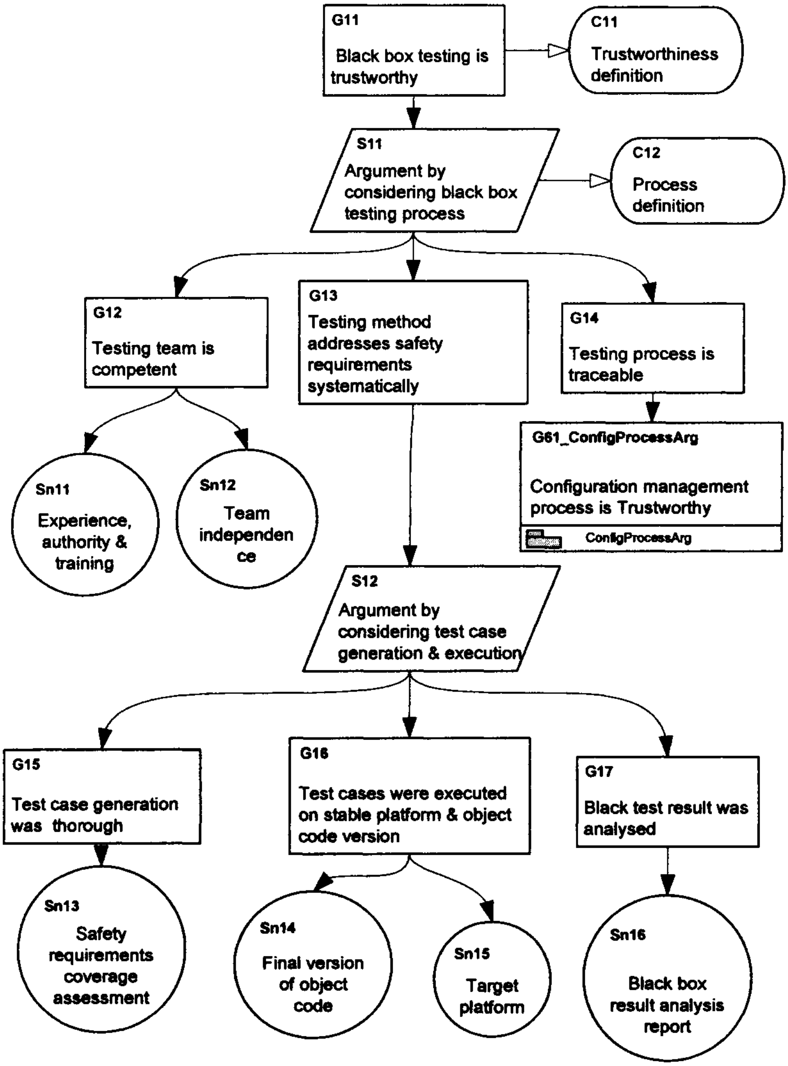

G11
Black box testing is trustworthy
C11
Trustworthiness definition
S11
Argument by considering black box testing process
C12
Process definition
G12
Testing team is competent
G13
Testing method addresses safety requirements systematically
G14
Testing process is traceable
G61_ConfigProcessArg
Configuration management process is Trustworthy
ConfigProcessArg
Sn11
Experience, authority & training
Sn12
Team independen ce
S12
Argument by considering test case generation & execution
G15
Test case generation was thorough
G16
Test cases were executed on stable platform & object code version
G17
Black test result was analysed
Sn13
Safety requirements coverage assessment
Sn14
Final version of object code
Sn15
Target platform
Sn16
Black box result analysis report

Figure 4. Black Box Process Argument

Figure 4 shows the goal structure for the *G11_TestingProcessArg* away goal. Here, the argument stresses the importance of process evidence to justify the trustworthiness of the black box testing evidence. The process evidence addresses team competency, test case generation, execution and analysis, and testing traceability. Firstly, the competency of the testing team (goal: *Sn11*) is supported by claims about the team's qualifications and independence from the design team (avoiding common mode failures with the design team). Secondly, the goal structure contains an argument that claims that the process of generating, executing, and analysing test cases is systematic (argument: *S12*). This argument is supported by items of evidence such as the fact that the test cases cover all defined safety requirements and executed on the final source code and target platform. Finally, the goal structure shows that the black box testing process is traceable. However, in order to avoid complicating the goal structure, the justification argument for traceability is documented elsewhere (module: *ConfigProcessArg*).

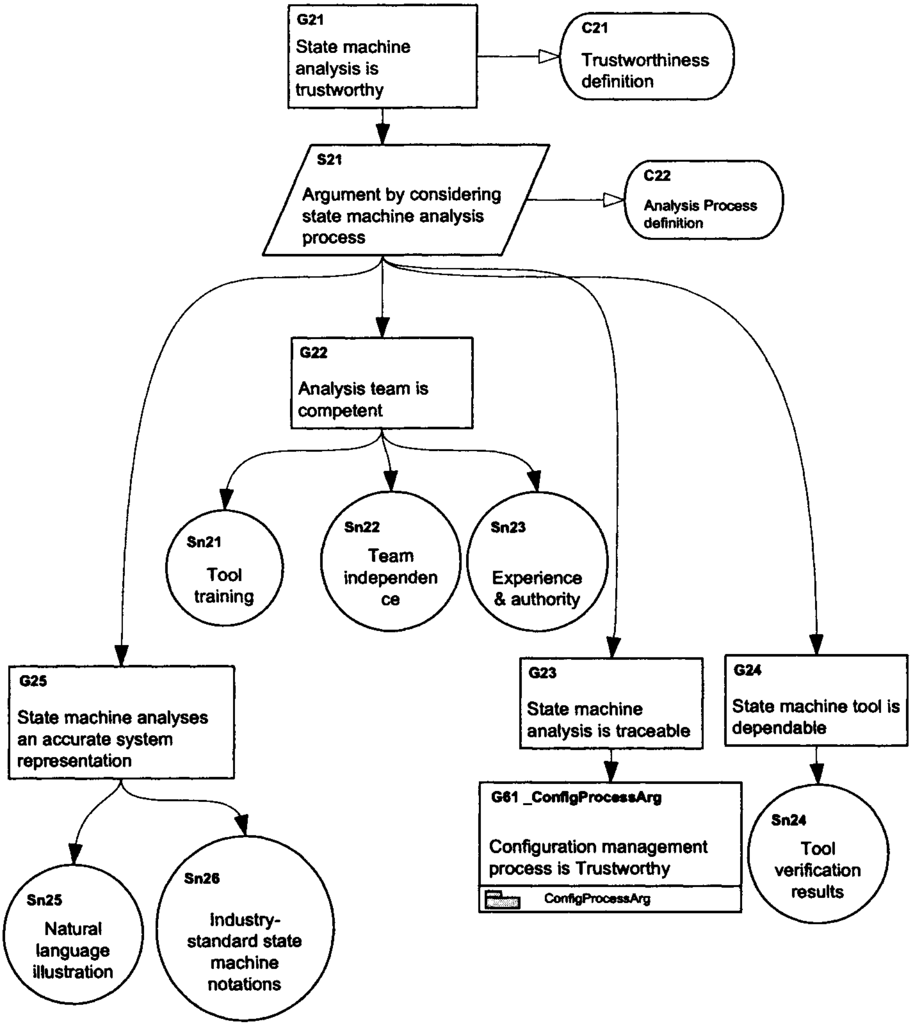

G21
State machine analysis is trustworthy
C21
Trustworthiness definition
S21
Argument by considering state machine analysis process
C22
Analysis Process definition
G22
Analysis team is competent
Sn21
Tool training
Sn22
Team independen ce
Sn23
Experience & authority
G25
State machine analyses an accurate system representation
G23
State machine analysis is traceable
G24
State machine tool is dependable
G61 _ConfigProcessArg
Configuration management process is Trustworthy
ConfigProcessArg
Sn24
Tool verification results
Sn25
Natural language illustration
Sn26
Industry-standard state machine notations

Figure 5. State Machine Process Argument

Similarly, the goal structure depicted in Figure 5 justifies the trustworthiness of the state machine analysis by referring to items of process evidence. In addition to the consideration of staff competency and process traceability, this goal structure depends on the state machine's tool and notations. The dependability of the state machine tool is verified against the tool's operational requirements (solution: *Sn24*).

A formal approach such as state machine analysis requires a simple and unambiguous representation. This facilitates the definition of correct correspondence between the formal model and the actual software artefact. This claim about correct representation is supported by referring to the adequacy of the notations and the clarity of the accompanying natural language (solutions: *Sn25* and *Sn26*).

In short, in this section we have showed how to attach process-based arguments to the product evidence. In the next section, the advantages of arguing explicitly about the trustworthiness of the process and the relationship between the integrity of the product and the integrity of the process are discussed.

5 Discussion

One of the goal-based standards that begins to address the suggestions made in this paper is the SW01/CAP 670 regulations for Software Safety Assurance in Air Traffic Services (Civil Aviation Authority 2003). SW01 mandates that arguments and evidence should be submitted that demonstrate that the risk posed by the software is tolerable. The arguments and evidence should show the achievement of five regulatory objectives, namely:

- Safety requirements validity
- Safety requirements satisfaction
- Safety requirements traceability
- Non-interference by non-safety functions
- Configuration consistency

The guidance of SW01 states that the achievement of the above objectives should be substantiated by a combination of direct and backing evidence. What distinguishes SW01 from other goal-based standards, such as Def Stan 00-56, is that the direct evidence is not limited to product evidence. For example, one of the direct evidence for the demonstration of requirements validity is expressed as follows:

> *"The software safety requirements should be specified in sufficient detail and clarity to allow the design and implementation to achieve the required level of safety."*

The above type of evidence is considered 'direct' with regard to requirements validity given that this is a process claim, which shows that the concept of directness is relative to the claim being made and does not always have to be related to the product evidence. This emphasises the observation that directness is a relative attribute that depends on the claim, i.e. if the claim is about the process, the process evidence should be direct. In general, process assurance is mostly directly related to claims about the credibility of a piece of evidence.

Arguing explicitly about the trustworthiness of the process protects the safety case argument against counter-arguments based on argument deconstruction and common mode failures.

Firstly, argument deconstruction attempts to undermine an argument by referring to doubts about hidden assumptions and conclusions (Armstrong and Paynter 2004). However, by justifying the process behind the generation of the evidence, safety engineers can address and mitigate the impact of such hidden assumptions explicitly early on during the safety case development. For example, in the sheet metal press safety argument shown in Figure 3, the Context *C1 "Identified Software Hazards"* is supported by an argument that justifies the trustworthiness of the hazard identification process (*HzIdentTrustworthy*). By arguing explicitly about the trustworthiness of the hazard identification process, the safety argument mounts a defence against counter-arguments that question the completeness of the list of defined hazards.

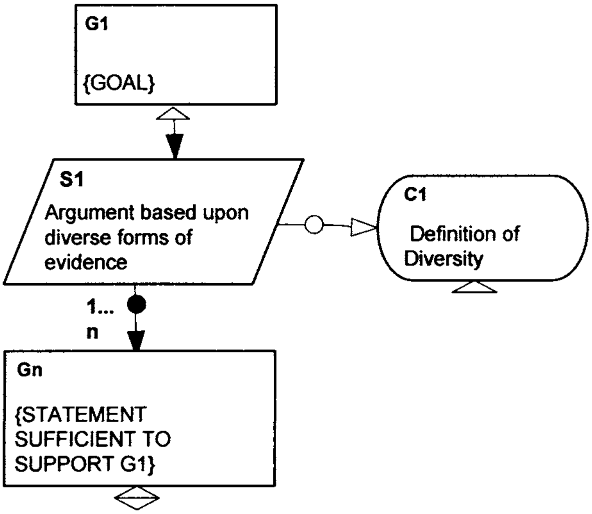

G1
{GOAL}
S1
Argument based upon diverse forms of evidence
C1
Definition of Diversity
1...
n
Gn
{STATEMENT SUFFICIENT TO SUPPORT G1}

Figure 6. Diverse Argument Safety Case Pattern (Kelly 1998)

Secondly, arguing explicitly about the trustworthiness of the process can demonstrate independence between the evidence items provided in a safety argument. Evidence independence is particularly necessary to protect against common mode failures. The goal structure in Figure 6 depicts a safety case pattern for a diverse argument (Kelly 1998). The diversity strategy (*S1*) is supported by one or more distinct statements (*Gn*). Although diversity might be proven by referring to the conceptual and mechanistic differences between evidence types (e.g. analysis and testing), underestimating diversity at the process level (e.g. independence of personnel and verification environment) can challenge the diversity of product evidence.

Figure 7 depicts an extended version of the above safety case pattern (diverse argument). An away goal (*GArgDiverse*) is attached to the argument strategy (*S1*). This away goal is used to justify diversity of the items of evidence (*Gn*) at both the product and process levels.

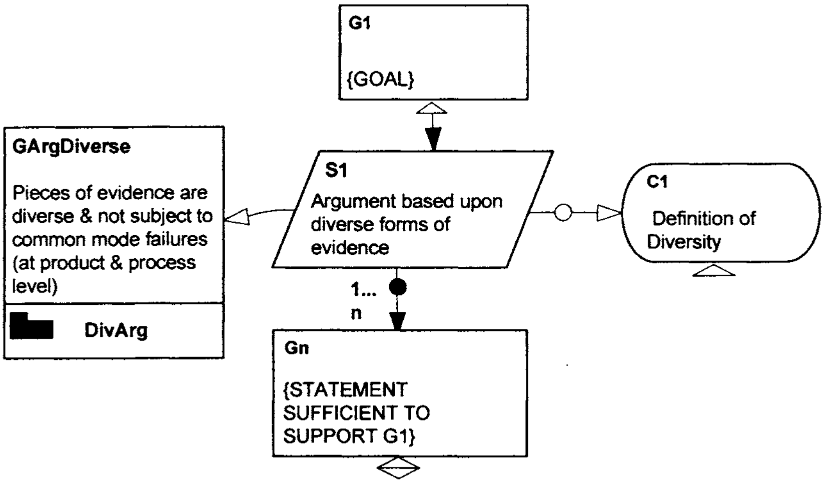

G1
{GOAL}
GArgDiverse
Pieces of evidence are diverse & not subject to common mode failures (at product & process level)
DivArg
S1
Argument based upon diverse forms of evidence
C1
Definition of Diversity
1...
n
Gn
{STATEMENT SUFFICIENT TO SUPPORT G1}

Figure 7. Extended Diverse Argument Safety Case Pattern

It is also important to address the approach presented in this paper from a practical perspective. It may be complicated to attach a process argument to each item of product evidence. However, this can be simplified by using GSN modular features, i.e. away goals. Away goals can support process claims by arguments located in another part of the safety case (modules). It may also be possible to present process justification in less detail. Instead of linking a process argument to each item of product evidence (i.e. solutions), it may be feasible to link the process argument to a high-level strategy, as shown in the safety case pattern in
Figure 7. Additionally, not all safety arguments are of the same significance. Safety case developers may choose to elaborate only on high-priority safety arguments, where hidden process assumptions have direct impact on undermining confidence.

Finally, although deductive safety arguments are advantageous, there will always be inductive elements, especially in software safety arguments, due to the systematic nature of software failures, i.e. failures resulting from faults in specifications and design. Such inductive elements make it infeasible to declare that the premises of the safety argument are true. Therefore, process arguments of the type we presented address partly this problem. They uncover flaws, related to the human factors, in the way the specification and design of the software are carried out, by tackling the otherwise implicit assumptions about the consistency and correctness of the process underlying the generation of the certification evidence.

6 Summary

In this paper, we have argued that lack of process assurance can undermine even the seemingly strongest product safety argument. The role of the process should not be underestimated even in product arguments. However, unlike SIL-based process arguments that attempt to lead to claims of product integrity, the process of the type we presented are targeted and consequently justified and assured in the context of the product. To show how to achieve integrated product and process safety arguments, we have used features of GSN introduced to handle modularity. We have linked items of product evidence (Solutions) to process arguments (encapsulated into Away Goals). In this way, it is possible to expose hidden arguments and assumptions concerning the provenance of the product evidence.

References

Armstrong J and Paynter S (2004). The deconstruction of safety arguments through adversarial counter-argument. Proceedings of the Computer Safety, Reliability, and Security, 23rd International Conference, SAFECOMP 2004, Germany, September 21-24, 2004 (Lecture Notes in Computer Science 3219 Springer 2004)

Civil Aviation Authority (2003), SW01 - Regulatory objective for software safety assurance in air traffic service equipment, CAP 670: air traffic services safety requirements, published by the UK Civil Aviation Authority, 12 June 2003

EUROCAE (1994). ED-12B/DO-178B: Software considerations in airborne systems and equipment certification. EUROCAE, 1994

Caseley P, Tudor N and O'Halloran C (2003). The case for an evidence based approach to software certification. Safety Standards Review Committee, UK Ministry of Defence, 2003

Hall A (1990). Seven myths of formal methods. IEEE Software archive, Volume 7, Issue 5, 1990

Kelly T P (1998). Arguing safety – a systematic approach to safety case management. DPhil Thesis, Department of Computer Science, University of York, UK, 1998

Kelly T P (2001). Concepts and principles of compositional safety cases. COMSA/2001/1/1, Research report commissioned by QinetiQ, 2001

McDermid J A (2001). Software safety: where's the evidence?. Proceedings of the Sixth Australian Workshop on Industrial Experience with Safety Critical Systems and Software, Australian Computer Society, 2001

Redmill F (2000). Safety integrity levels – theory and problems, lessons in system safety. Proceedings of the Eighth Safety-Critical Systems Symposium, 1-20 Redmill F and Anderson A (eds), Southampton, UK, Springer Verlag, 2000

UK Ministry of Defence (2004). 00-56 Safety Management Requirements for Defence Systems, Part 1: Requirements, Issue 3. UK MoD, 2004

Weaver R A (2003). The safety of software – constructing and assuring arguments. DPhil Thesis, Department of Computer Science, University of York, UK, 2003

The Benefits of Electronic Safety Cases

Alan Newton and Andrew Vickers
Praxis High Integrity System Limited
Bath, England

Abstract

There is an increasing demand by society for safety critical systems not only to be safe, but also to be seen to be safe. With increasing product complexity this is placing a significant load upon both the safety community and project budgets. Partially in response to these challenges Praxis has been increasingly using processes that make use of internet-technologies to develop electronic safety cases for a number of years, both to improve quality and to reduce costs and workload upon project teams. This initially ad-hoc approach to the use of internet technologies has now evolved into a commercial product - the eSafetyCase toolkit. The toolkit extends the basic process further and allows other organisations to gain the benefits of this approach. Use of this toolkit has significantly informed our opinions on the use of electronic safety cases.

This paper outlines the potential benefits of developing electronic safety cases to those in the safety engineering chain including: society, regulators, ISAs, companies, projects and safety case developers. Additionally this paper shows how some of these benefits can be achieved using the eSafetyCase Toolkit.

1 Introduction

As regulatory regimes are tightened and systems become more complex, the need for high-quality safety cases with coherent arguments is becoming increasingly important. Safety cases must not only contain the information necessary to justify the use of the equipment, but must do so in an easily accessible manner. Reviewers often want to pursue "safety threads" throughout a safety case to test how particular issues are dealt with in a 'top to bottom' manner. Such a review is often required by Independent Safety Assessors (ISAs) who want to trace and verify particular lines of enquiry. Given the complexity of modern systems and their arguments it is increasingly becoming less acceptable to simply provide a safety case supported by large quantities of paper evidence and expect reviewers such as the ISA to satisfy themselves that safety claims made by the equipment provider have been successfully demonstrated. More help needs to be provided to support the accessibility of safety cases to review. Supporting review of complex documents is a genuine business requirement, since if the ISA (or others) cannot cost-effectively assure themselves that risk has been reduced to an acceptable level, then introduction of equipment might be delayed.

Praxis High Integrity Systems (Praxis HIS) has a history of making safety cases more accessible to review. Our approach to solving this problem is embodied within the eSafetyCase methodology and toolset. The eSafetyCase methodology for presenting safety case information was developed over 5 years ago and Praxis has used it on a wide range of projects, including families of systems involving over 100 electronic safety cases. These tools have not only provided the benefits in safety case review that were originally envisaged, but the existence of the toolset has helped us to consider a variety of other aspects of electronic safety case management including support for team-working and 'template-based guidance' for the safety process.

The purpose of this paper is to present some emerging views on how electronic management of safety information can offer a number of benefits. The paper does this by addressing three challenges to managing safety programmes:

- the scarcity of safety engineering resource,
- the monitoring of safety case progress,
- and the cost-effective exploitation of safety information.

For each challenge, the relevant issues are outlined, before an electronic mechanism for addressing the challenge is described. Conclusions on the use of an electronic medium to support overall safety case management are then drawn.

2 Making the most of scarce resource

With increased safety engineering workloads, there is a need for a larger pool of engineers to do the necessary work. However industry is currently suffering a shortage of suitably qualified safety engineers. As senior safety engineers become busier they will lack the time to provide training support for junior safety engineers, which further exacerbates the long-term resourcing problem. A tool-based approach can support the training and development of new safety engineers within an active project. Tools can provide a double benefit to companies by providing additional resource to a project and by reducing (although not eliminating) the degree of support required for junior safety engineers for certain elements of safety case production.

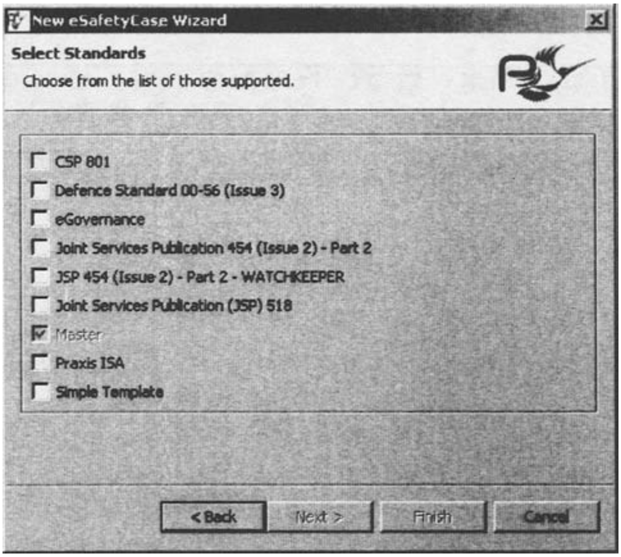
A number of features within the eSafetyCase Toolkit support this approach, including:

- Template guidance.
- Controlled multi-user team working.
- Management of domain knowledge.

2.1 Generating a safety case framework using templates

Safety standards can be generic (such as IEC 61508) or sector specific (such as Def Stan 00-56). An individual project may be required to be compliant with one or more safety standards. Whenever a new project is started, the relevant standards have to be analysed and a framework safety case document created with the appropriate sections ready for the safety engineering team to populate. If a junior safety engineer is involved within the project team, then typically additional guidance information will need to be included within the framework document, or day-to-day support will have to be provided to assist with safety case population. Whichever approach is chosen, setting up the framework can result in cost and time delays to the start of the safety aspects of the project.

The eSafetyCase toolkit addresses these issues through the use of pre-populated templates, one for each standard required by the purchaser of the tool. The templates are designed to optimise (minimise) the number and content of pages that must be populated in order to satisfy all the standards adopted by a particular project. Additionally the templates include guidance text that explains to tool users what information should appear on a particular page. The guidance text has been produced by senior safety engineers within Praxis HIS and only has to be defined once when a new standard is issued. The guidance is then available to all users of the tool. The eSafetyCase toolkit provides a 'wizard' (recipe-led document construction) that leads the user through the process of setting up a safety case document framework. The process includes selecting the appropriate standards. The whole process of setting up a safety case framework document for a collection of standards only takes a few minutes.



New eSafetyCase Wizard

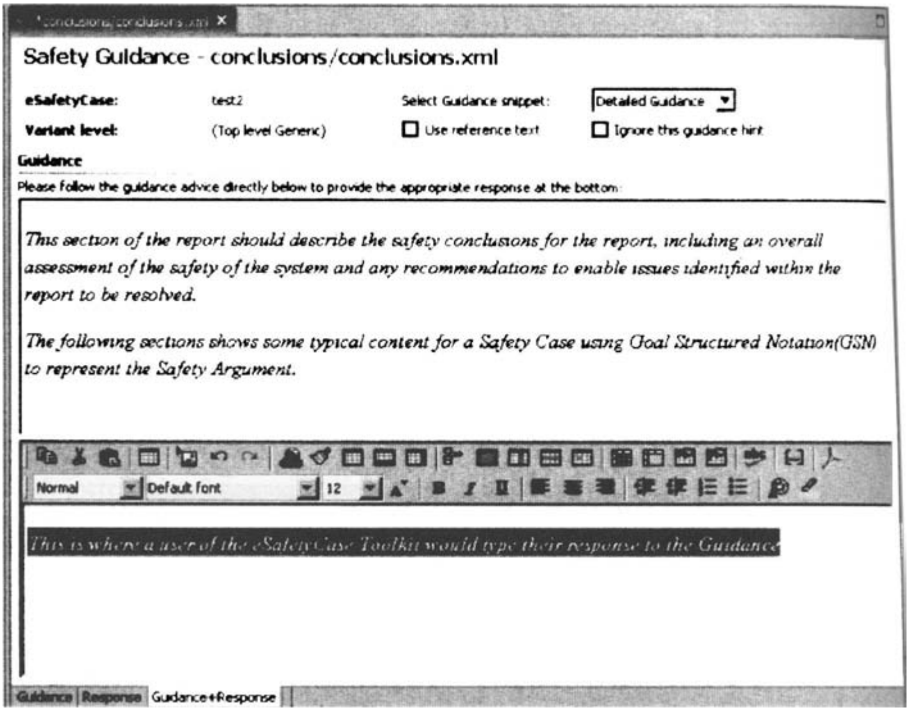
Select Standards

Choose from the list of those supported.

☐ CSP 801
☐ Defence Standard 00-56 (Issue 3)
☐ eGovernance
☐ Joint Services Publication 454 (Issue 2) - Part 2
☐ JSP 454 (Issue 2) - Part 2 - WATCHKEEPER
☐ Joint Services Publication (JSP) 518
☑ Master
☐ Praxis ISA
☐ Simple Template

Figure 1. The Wizard page allows Users to select standards when constructing a framework document; the example shows a sample set for a military project

Having completed the construction of the safety case framework document, the content of the safety case has to be provided. Typically this requires a senior safety engineer to generate the text or to provide detailed day-to-day guidance to more junior staff. The load on senior safety staff can be reduced if appropriate general project guidance is available to team members. An electronic approach to safety case generation can provide the opportunity to provide this detailed project guidance during the safety case production process. One example of this that occurs within eSafetyCase is the integrated editor (figure 2). The integrated editor allows users to view guidance provided by the templates whilst authoring the actual safety case contents. Every time a safety engineer edits the contents of a page the guidance information is automatically available. Although the guidance is stored within the document model used by the tool, it is not published as part of the delivered report unless specifically selected by the user of the tool.



Safety Guidance - conclusions/conclusions.xml

eSafetyCase: test2 Select Guidance snippet: Detailed Guidance

Variant level: (Top level Generic) □ Use reference text □ Ignore this guidance hint

Guidance

Please follow the guidance advice directly below to provide the appropriate response at the bottom:

This section of the report should describe the safety conclusions for the report, including an overall assessment of the safety of the system and any recommendations to enable issues identified within the report to be resolved.

The following sections shows some typical content for a Safety Case using Goal Structured Notation(GSN) to represent the Safety Argument.

This is where a user of the eSafetyCase Toolkit would type their response to the Guidance

Figure 2. The integrated editor allows users to view the guidance whilst providing new content.

2.2 Cooperative tool-based working

Having defined a safety case framework document, often the next task is to follow the guidance located on each page of the document and insert the relevant project related information. There are a number of approaches that may be adopted when populating the contents of a safety case, including:

- A single safety engineer is provided with the project information and is expected to gain a very detailed knowledge of the system in order to provide concise and relevant supporting descriptions.
- A team of engineers can work together on a single document where each engineer works on the document in turn, however this approach can lead to bottlenecks in document production and in many cases the design engineers do not form part of the safety engineering team.
- An alternative team working approach is for each engineer to write their section separately and then one member of the team has to compose the parts to make a complete and consistent document.

In each of these cases there are drawbacks to the working approach, a more viable approach is for a document to exist as a number of editable fragments, e.g. pages, and for users to be able to modify different pages in the report simultaneously, avoiding the bottlenecks in the production process. Such a fragment-based system could restrict editing privilege for each part of the repository to specific users, in this way a team could be composed of engineers with different skills each allowed to modify a small part of the document. This is the approach adopted by the eSafetyCase toolkit allowing safety case documents to be shared over a network whilst they are being developed. Clearly the overall coherence of the document still requires review.

2.3 Management of domain knowledge

When developing a safety case it is often necessary to populate a number of prescribed sections that do not require safety engineering knowledge. Such sections might provide contextual information for the reader of the safety case. For example the system definition documentation might be supplied to the safety engineer with a view to forming the 'system description' section of the safety case. In such a case the safety engineer would need to extract the particular system description information that was relevant to the safety case – an approach that is not always correct first time.

An alternative approach is to include a developer within the safety engineering team and provide them with suitable tool-based guidance. The approach to including a developer within the team and providing them with suitable guidance is analogous to that used for supporting junior safety engineers. In both cases the guidance can be made appropriate to the safety standards being addressed (see §2.1 and §2.2). In addition to the potential of providing improved quality within the system description, this approach has the added benefit that it can reduce the workload placed upon the safety engineers, and has a higher probability of being correct using fewer iterations of the process.

3 Knowing when to stop

When developing a safety case, a key issue can be determining when 'enough is enough'. It is clearly important to ensure that sufficient safety information is supplied but also important to ensure that excess information is not supplied.

Although safety standards may mandate a number of requirements upon the safety case developer, they will typically not tell you the level of detail and depth to which evidence should be provided. The level of detail depends upon the degree of safety risk as well as the opinions of the Project Safety Engineer and ISA. Additionally, when producing a safety case report that adheres to a specific standard there is a need to ensure that all the requirements of the standard have been fully addressed when the Safety Case is published. This can be difficult to verify when producing manual reports only using text editors, since other than visual inspection there is no way to ensure that all the issues identified in the standard have been considered and addressed.

The following sections of this paper examine how a safety case tool can address the problem of showing that the safety case contains a complete argument and that checking that sufficient evidence to meet the requirements of both the standard and the argument has been gathered.

3.1 Why are we still arguing?

Within the critical systems industry we have come across many cases where the terms Safety Case and Safety Argument have been used interchangeably. For the purposes of the discussion within this section we take the words "Safety Case" to be the Safety Case Report including the Safety Argument and its supporting evidence. We use the words "Safety Argument" to denote the structured decomposition of the claim that a system may be safely used within its defined parameters and environment. Increasingly Goal Structured Notation (GSN) (YSE, 1997) is being used to represent these arguments and this is the approach adopted within the eSafetyCase Toolkit.

Demonstrating that a system is safe to use is not achieved by purely structuring the safety argument, it is also necessary to ensure that the evidence gathered is sufficient for the argument to be credible. Only when a suitable level of credibility has been attained can evidence-gathering be deemed complete. For complex systems the judgement of when a safety case is complete can become a significant safety management challenge. Safety arguments can be very large and weaknesses in argument branches due to partial or missing evidence can be difficult to identify.

The adoption of appropriate electronic safety case tools makes this safety management issue easier to address. Once the "Safety Argument" has been captured electronically it is possible to model some simple properties and use this information to manage the assessment of completeness of evidence gathering.

Figure 3 shows a fragment of a typical GSN safety argument, in this case all the branches of the tree are assumed to have equal weighting (importance) – other weightings could of course be used. Within this fragment we are attempting to demonstrate that goal 4, "All hazards are adequately mitigated", has been met. The goal is decomposed into some lower-level goals whose solutions are easier to demonstrate - when they are all satisfied, the goal is met. If one solution is fully satisfied then Goal 4 is 25% satisfied, and if that solution is only 50% complete then Goal 4 will only be 12.5% satisfied. These satisfaction numbers can be inherited all the way up to the top-level goal, and this can give a simple measure of the completeness of the argument.. Clearly it should be not be used as an absolute measure of completeness since it is does not capture the difficulty of gathering different types of evidence. The eSafetyCase toolkit provides a viewer that allows the completeness of the argument to be easily reviewed (Figure 4). This is a simple measure of completeness, but for complex arguments even simple measures can be useful in informing progress.

Such an approach to measuring completeness can be enhanced by adding different weightings to key branches. Weightings can be used to indicate different importance in evidence, e.g. "Primary" versus "Backing" evidence. Different weightings can be combined with the simple completeness measure described earlier to provide a more accurate measure of the argument completeness.

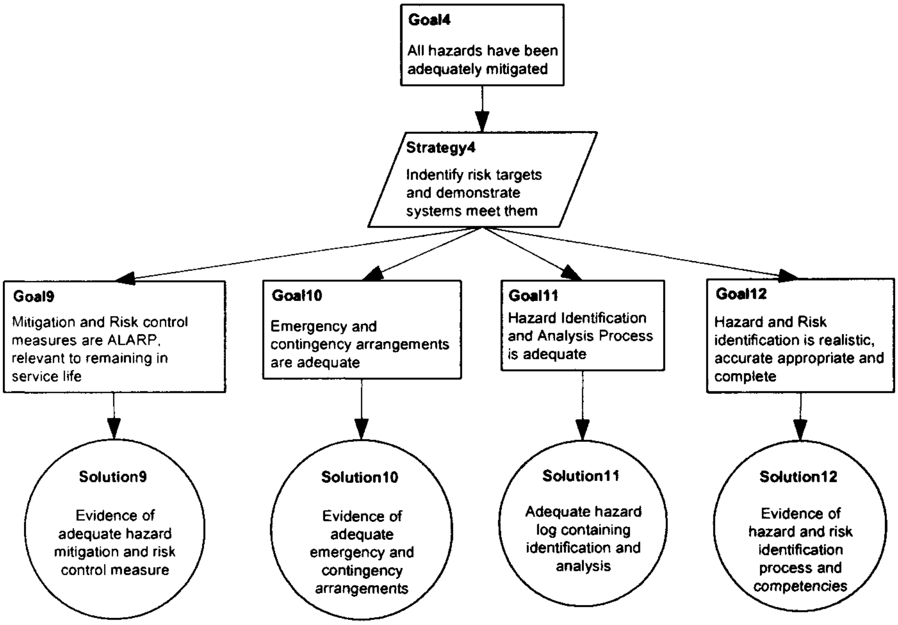
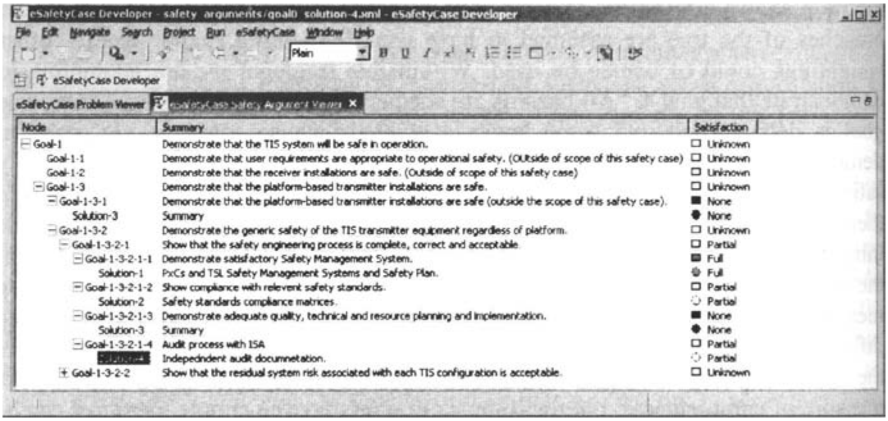

Goal4
All hazards have been adequately mitigated
Strategy4
Indentify risk targets and demonstrate systems meet them
Goal9
Mitigation and Risk control measures are ALARP, relevant to remaining in service life
Goal10
Emergency and contingency arrangements are adequate
Goal11
Hazard Identification and Analysis Process is adequate
Goal12
Hazard and Risk identification is realistic, accurate appropriate and complete
Solution9
Evidence of adequate hazard mitigation and risk control measure
Solution10
Evidence of adequate emergency and contingency arrangements
Solution11
Adequate hazard log containing identification and analysis
Solution12
Evidence of hazard and risk identification process and competencies

Figure 3. A Safety Argument can be decomposed until the evidence to demonstrate the safety of the system can be clearly identified


eSafetyCase Developer - safety arguments/goal0 solution-4.xml - eSafetyCase Developer
File Edit Navigate Search Project Run eSafetyCase Window Help
Plain
eSafetyCase Developer
eSafetyCase Problem Viewer
eSafetyCase Safety Argument Viewer

Node	Summary	Satisfaction
Goal-1	Demonstrate that the TIS system will be safe in operation.	□ Unknown
Goal-1-1	Demonstrate that user requirements are appropriate to operational safety. (Outside of scope of this safety case)	□ Unknown
Goal-1-2	Demonstrate that the receiver installations are safe. (Outside of scope of this safety case)	□ Unknown
Goal-1-3	Demonstrate that the platform-based transmitter installations are safe.	□ Unknown
Goal-1-3-1	Demonstrate that the platform-based transmitter installations are safe (outside the scope of this safety case).	■ None
Solution-3	Summary	● None
Goal-1-3-2	Demonstrate the generic safety of the TIS transmitter equipment regardless of platform.	□ Unknown
Goal-1-3-2-1	Show that the safety engineering process is complete, correct and acceptable.	□ Partial
Goal-1-3-2-1-1	Demonstrate satisfactory Safety Management System.	■ Full
Solution-1	PxCs and TSL Safety Management Systems and Safety Plan.	● Full
Goal-1-3-2-1-2	Show compliance with relevent safety standards.	□ Partial
Solution-2	Safety standards compliance matrices.	○ Partial
Goal-1-3-2-1-3	Demonstrate adequate quality, technical and resource planning and implementation.	■ None
Solution-3	Summary	● None
Goal-1-3-2-1-4	Audit process with ISA	□ Partial
Solution-4	Indepedndent audit documnetation.	○ Partial
Goal-1-3-2-2	Show that the residual system risk associated with each TIS configuration is acceptable.	□ Unknown

Figure 4. The eSafetyCase Toolkit provides a viewer which allows the completeness of the argument to become immediately apparent using a “Traffic Light” indicator

3.2 Have we finished yet?

As discussed in §2.1 a template-based approach can provide support in ensuring that a standard has been fully considered during the production of a safety case. Investment is required up-front in the safety engineering process to produce high-quality templates with detailed guidance but the benefit from these templates can be recouped for each safety case subsequently created. If each page of the safety case template is reviewed and the appropriate information and evidence, as specified by the template guidance, is inserted into the safety case then this will go a long way in ensuring that the standards-based requirements imposed upon the safety case have been addressed.

The eSafetyCase Toolkit provides an example of how this could be achieved. The template generation approach gives a collection of pages that make up the reports for each standard, each of these pages is made up of guidance snippets which interpret the standard requirements in a practical way.

3.3 Are you sure?

This template-based approach to safety case development ensures that safety case engineers are aware (reminded) of all the issues identified within the standard definition and encourages them to structure their argument accordingly. However, it is not possible for any tool to prove that the evidence within the safety case is adequate; it can only show that evidence exists within the document to address the safety issues, and can support ISAs by ensuring that they examine all parts of the safety case during the assessment phase.

4 Cost-effective exploitation of safety information

Ideally, when producing a safety case, the evidence presented should satisfy each of the stakeholders first time. One key element in achieving this is to make different safety reports available in a timely manner with levels of detail that match the progress made to date. It can be quite common for outline reports to be made available early in the process to obtain stakeholder buy-in and avoid nasty shocks later in the project. However producing reports or reformatting data to extract data subsets can be a difficult and costly task, since this process not only involves extracting data but also identifying the key issues and structuring them in a way that fulfils the purpose of the published outline document. Such difficulties in document publication can mean that fewer progressive reports are produced than is really necessary to manage the acceptance risk – particularly when acceptance is a multi-stakeholder activity.

Not only does the ease of publishing electronic safety cases address this issue, but it can also provide new opportunities to effectively manage data and its representation. The following sections discuss the ease with which information can be re-used, the means by which reports with different structures can be created, and how reports can be output in different formats to support different modes of working.

4.1 Why only re-use software?

In our experience safety cases are often written bespoke for a particular equipment item, perhaps using similar documents as a template, with boilerplate descriptions for particular company-related activities, e.g. quality management, safety management, etc. However, there are a number of opportunities for formal re-use of safety information that can make the process of safety case authoring more efficient. These include:

- Re-use of Global Company Information
- Re-use of Product Family Information

One approach to constructing a safety case is to start by creating a framework document and then to insert standard global company information, perhaps describing and justifying the tools, methods, procedures and staff used by an organisation. This approach to constructing the initial document can be carried out in an ad-hoc manner, with fragments of information cut and paste into the document from a number of disparate sources. This means that the editing/composition process can be susceptible to corruption. When some changes occur corporately in one of the main processes, all the documents describing this process have to be updated independently. This is a mundane and often expensive or error-prone process.

Global company information can be effectively managed within toolsets that support electronic safety cases by including the global company information within the templates used during the template generation process (see §2.1). By making this information generic to all safety cases, all safety cases can be updated simply once information in one location has been modified.

When a company produces a large number of configurable equipment items, used in different locations there may be a need for a family of related safety cases. This need for a collection of safety cases might be as a result of different components within the equipment, or different mitigations and constraints imposed upon the use of the equipment due to its location. In all these cases a significant proportion of the safety cases can be common. Ideally the common information should only be authored once, with information reused wherever possible.

Families of safety cases can be supported electronically by having a suitable underlying data model. Ideally, the model should allow data to be structured in a hierarchical manner that represents the variations between safety cases. The top-level node should contain all the generic information, whilst the lower-level nodes should only contain the variations in the safety case compared to the safety case nodes above them in the tree structure. Figure 5, shows an example of such a structure, in this case the safety case is made up of four pages, if a safety case was published from the generic level then the pages published would all be generic and contain only company wide information, as the user publishes safety cases from variants at lower levels of the tree then generic pages are masked by local pages which make the safety case specific to a particular equipment item used in a specific location. This approach is exploited all the way down the tree, with each safety case relying upon all the information available above them. In Figure 5, the safety case produced for variant 2-1 consists of local information, as well as from variant 2 and the generic safety cases. All of this can be achieved automatically, with a low level of user involvement. This is the approach to information reuse is the one we have adopted within the eSafetyCase toolkit.

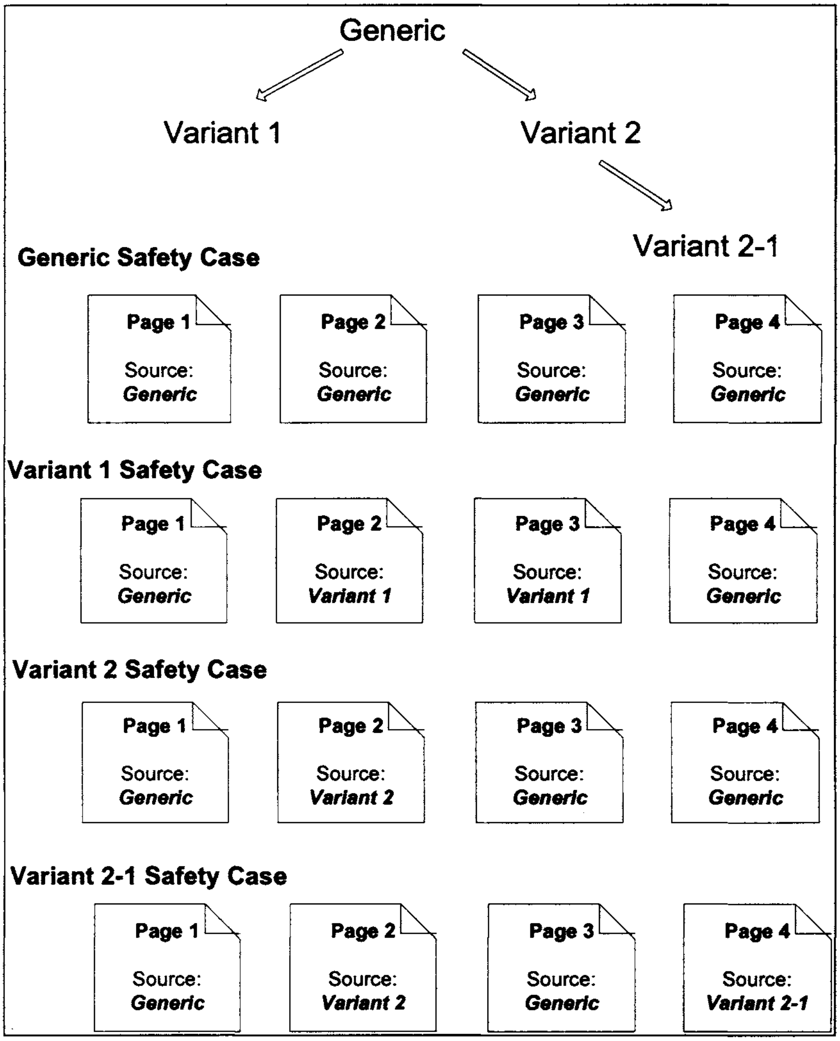
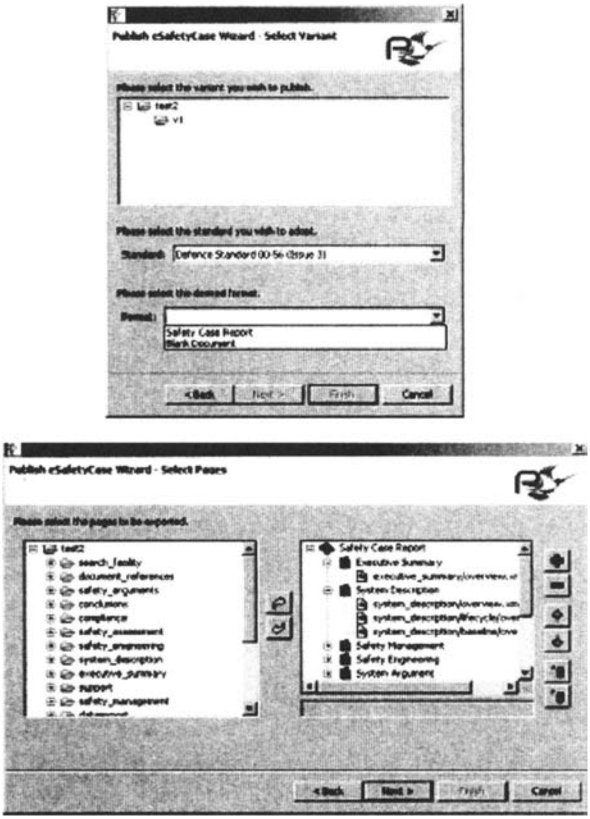


Figure 5. A tree structure document model can allow pages of information to be reused efficiently

4.2 One project, many reports

Having identified the standards to be addressed, and built the safety case framework, it is likely that the contents of the case will be populated in stages. Whilst the population of a safety case is taking place it is likely that different stakeholders, e.g. ISA, Project Manager, Product Developer, etc. may want different reports and sub-reports e.g. Preliminary Safety Case, Interim Safety Case, and Developer reports. Producing many stakeholder-specific reports can be difficult to justify if the cost of producing such reports is prohibitive. However, electronic approaches to managing safety information mean that it is easy to generate reports. The eSafetyCase toolkit is a good example of this approach. A wizard (Figure 6) guides you through the publication process, each of the standards templates used to create the framework document contains one or more predefined document definitions composed of pages within the template. The document definitions may be extended further by adding additional pages; this is particularly useful if you have added new pages to the safety case. An alternative to using the predefined document templates is to start with a blank template and add sections and pages to create a bespoke definition. Any tailored document definitions can be saved and retrieved to allow any project specific documentation to be quickly re-published. This example shows how electronic document management can easily respond to the reporting needs of a project, some of which may not be known at the outset of the project, by allowing quick and easy publication of reports or fragments of reports.



Publish eSafetyCase Wizard - Select Variant

Please select the variant you wish to publish.

test2
v1

Please select the standard you wish to adopt.

Standard: Defence Standard 00-56 (Issue 3)

Please select the desired format.

Format:

Safety Case Report
Blank Document

Publish eSafetyCase Wizard - Select Pages

Please select the pages to be exported.

test2
search_facility
document_references
safety_arguments
conclusions
compliance
safety_assessment
safety_engineering
system_description
executive_summary
support
safety_management

Safety Case Report
Executive Summary
executive_summary/overview.ut
System Description
system_description/overview.utm
system_description/lifecycle/over
system_description/baseline/ove
Safety Management
Safety Engineering
System Argument

Figure 6. Documents are easily published using the eSafetyCase toolkit through the use of Wizards

4.3 One report, many formats

Some of the key advantages of presenting information electronically are that:

- Electronic documents are easy to disseminate.
- Reviewers can easily follow "safety threads" through the document using hyperlinks, working from high-level hypothesis to evidence.
- The evidence documents can be supplied on CD/DVD with the safety case report and easily accessed.
- Large arguments can be easily partitioned into manageable chunks that can easily be navigated through the use of hyperlinks.

Typically, there are three types of electronic formats suitable for the publication of safety cases and given the correct structuring of the underlying data model it should be possible to generate any of the outputs from the same data source.

- Read only documents, which may or may not contain hyperlinks that allow the reader to navigate around the report, but do not allow access to the supporting evidence. PDF documents are typical examples of this form of presentation. In this format the supplier of the delivered document can be sure that it is not modified without their knowledge.
- Editable documents containing all the responses to the guidance notes, do not contain links to the supporting evidence and may be modified outside of the safety case authoring environment. Rich Text Files (RTFs) is a typical example of this form of file which can be modified post-export using tools such as Microsoft Word.
- A third form of output that may be generated from the data model could be an HTML Intranet website, which renders all the pages of the data model as HTML Files, which are accessed via a menu system, and which can be used to access the supporting evidence files. This format provides the easiest way of following safety threads through the safety case to the evidence, and is probably of most use to the ISA in analysis of the safety case.

Each of these data formats supports a different way of developing and reading an electronic safety case and all of these formats are supported by the eSafetyCase toolkit.

4.4 Efficient communication just the start

This section has discussed how electronic safety cases can be used efficiently to aid communication between the safety engineering team and other stakeholders. This is achieved by making work undertaken by the team widely visible. However, the easy availability of safety reports should clearly not be used as a replacement for effective meetings between the safety team and the stakeholders, but instead as a support mechanism for this process.

5 Conclusions

With increasing system complexity, the compilation of safety cases is becoming more challenging. The distinction between safety argument and safety case is a valuable distinction in helping to 'see the wood for the trees' but there remain challenges for the compilers and assessors of safety cases.

In this paper we have discussed a number of those challenges in detail, including:

- The need to make safety cases verifiable in a cost-effective manner
- The need to find effective means of using scarce safety engineering resource
- The need to cost-effectively automate those aspects of the safety case production that can be safely automated
- The need to support team safety working
- The need to safely be able to reuse information when supporting safety cases for families of systems.

There are different ways to solve these problems, but the approach Praxis has been using for five years now is to make use of electronic safety cases. This paper has discussed how electronic approaches can be used to solve each of these problems.

The approaches discussed here are not intended to "dumb down" safety engineering, but to provide examples of how it is possible to support a scarce group of engineers by allowing them to work more effectively, whilst at the same time providing training support for new talent within the industry. Relatively simple improvements can have significant benefits.

6 Acknowledgements

The authors would like to thank the Management, Software and Safety Engineers working at Praxis HIS, for their support during the development of the eSafetyCase toolkit. Particularly Keith Williams (Praxis Managing Director) and Dr Trevor Cockram whose original ideas sparked the development of the first eSafetyCase. The authors would also like to thank John Harvey and Felix Redmill who provided a number of detailed and useful review comments on the paper.

7 References

YSE (1997). *Goal Structured Notation Handbook*, York Software Engineering Ltd., 1997

Safety Management

A Longitudinal Analysis of the Causal Factors in Major Maritime Accidents in the USA and Canada (1996-2006)

C.W. Johnson
Dept of Computing Science, Univ. of Glasgow,
Glasgow, Scotland, G12 9QQ.
johnson@dcs.gla.ac.uk, http://www.dcs.gla.ac.uk/~johnson

C.M. Holloway
NASA Langley Research Center,
Hampton, VA 23681-2199, USA
c.m.holloway@nasa.gov

Abstract

Accident reports provide important insights into the causes and contributory factors leading to particular adverse events. In contrast, this paper provides an analysis that extends across the findings presented over ten years investigations into maritime accidents by both the US National Transportation Safety Board (NTSB) and Canadian Transportation Safety Board (TSB). The purpose of the study was to assess the comparative frequency of a range of causal factors in the reporting of adverse events. In order to communicate our findings, we introduce J-H graphs as a means of representing the proportion of causes and contributory factors associated with human error, equipment failure and other high level classifications in longitudinal studies of accident reports. Our results suggest the proportion of causal and contributory factors attributable to direct human error may be very much smaller than has been suggested elsewhere in the human factors literature. In contrast, more attention should be paid to wider systemic issues, including the managerial and regulatory context of maritime operations.

1 Introduction

This paper stems from a continuing study to validate assertions about the distribution of causes in adverse events. We are particularly concerned to establish whether or not the majority of accidents are 'blamed' on direct operator error. The results of an initial investigation into the causes of all major accidents and incidents in North American aviation from 1996 to 2003 cast doubt on previous studies that had asserted the importance of individual human factors in the immediate causes of adverse events (Johnson and Holloway, 2004). This work led to wider studies into accident reports across a range of other industries including rail and highway transportation (Holloway and Johnson, 2005, 2006). In contrast, this paper reports on our work to replicate the previous study and identify the proportion of causes and contributory factors associated with human error in the North American maritime industries across the decade from 1996 to 2006.

As mentioned, our initial work focused on the distribution of causal factors identified in aviation accident and incident reports. This decision was justified by the prominence of claims about human error in this industry. The first study focussed on all major adverse event reports issues by the US National Transportation Safety Board (NTSB) and the Canadian Transportation Safety Board (TSB) between 1996 and 2003 (Holloway and Johnson, 2004). This yielded a total of 26 US and 27 Canadian aviation investigations. Later sections will discuss the methods used in more detail. For now it is sufficient to observe that two analysts went through each of these reports developing their own independent classification scheme to distinguish between broad categories of causal and contributory factors. This identified approximately 40 causes and 53 contributory factors in the NTSB dataset and 50 causes with 53 contributory factors for the TSB. The subsequent classifications showed that only 37% of causal factors in the NTSB study related to individual human error. In contrast, 48% of causes and contributory factors can be categorized as organizational. 12% related to equipment. 'Other' causes accounted for 3%. In contrast, for the TSB 50% of the causes and contributory factors were related to individual error, 22% to organizational issues, 20% to equipment and 8% to 'other' factors. Although human error remains a significant factor in many of these accident reports, it is not true that investigatory agencies ignore the organizational issues that create the context for adverse events. It is also apparent from our study that the differences between the NTSB and the TSB reflect important differences in the types of air operations, and hence accidents, that occur in US and Canadian air space (Johnson and Holloway, 2004). The Canadian datasets contain far more incidents involving private pilots and technologically unsophisticated small aircraft. There are thus correspondingly fewer opportunities for organizational issues to intervene in these incidents, where single individuals will be performing most of the operations. We have recently extended these initial aviation studies by analysing the causes and contributory factors cited in NTSB reports for three different sample periods 1976-1984, 1996-2004 and 2004-2006. The preliminary results show considerable differences over this time period. The proportion of direct operator 'errors' diminishes as the proportion of organisational causes rises between the first and second samples. There is evidence that the proportion of causes due to human error has risen again in the most recent group of accident reports (Johnson and Holloway, in press).

The results of this initial study may not be typical of other safety-critical industries. The relatively high levels of training and regulatory control make it likely that organisational issues would be more prominent than human error in aviation when compared to other domains. Such caveats motivate the study, reported in this paper, of North American maritime accidents. Further motivation is provided by a recent survey commissioned by the UK Marine Accident Investigation Branch (2004). This examined 66 accidents. The report argued that one third of all the groundings involved a fatigued officer who was alone on the bridge at night. Two thirds of all the vessels involved in collisions were not maintaining 'a proper lookout'. An important strength of the MAIB study was that it published the methodology that was used to support these findings; 'Once selected, the accidents were then reviewed in detail by MAIB nautical inspectors in order to complete a questionnaire (Annex A) covering many aspects of bridge watch keeping practice, which had been developed for this study. The data gathered was input to a human factors database before analysis.' This scientific approach enables subsequent analysts to replicate their methods and, therefore, validate their results. Many previous studies have failed to provide readers with this methodological information. However, a number of caveats can be raised about the manner in which the accidents were selected for the MAIB study. The study excluded fishing and commercial vessels under 500 gross tons. Accidents involving vessels berthing, at anchor, or under pilotage, were also excluded. This reflected the study's focus on bridge watch-keeping during a passage rather than on navigation or maneuvering. The study also focused on the insights obtained by individual investigators looking at each accident. There does not seem to have been any attempt to conduct inter-analyst comparisons for individual reports.

2 Method

We were concerned to develop results that could be challenged or replicated by other researchers. All of the materials used in this study are available on-line and can be accessed by contacting the first author. We were also concerned to assess the validity of our results by comparing the insights obtained from different analysts. We, therefore, used two investigators to extract the causes and contributory factors from the accident reports that we studied. Each had more than a decade's experience in the development and analysis of safety-critical systems. Each studied the same sample of maritime accident reports. By choosing a ten-year window, the sample yielded a total of 22 accident reports from the NTSB and 160 from the TSB. This imbalance partly reflects the relative prominence of the Canadian maritime industries. It also reflects the way in which the TSB groups major and minor incidents within a single reporting framework. In contrast, the NTSB explicitly separates major accident reports from accident briefs, which were excluded from our study. Rather than impose our own arbitrary distinctions about the seriousness of each adverse event, we chose to analyse all of the TSB reports containing chapter headings presented within the period of our study. The reports ranged from high profile, multiple fatality accidents such as the Fire on Board the Panamanian passenger ship Universe Explorer through to less severe grounding incidents.

Our analysis progressed by extracting the causal and contributory factors that were identified in the aftermath of each investigation. This preprocessing stage was necessary to insure that each of the analysts focused on the same source, given that most of the documents were many pages in length. The identification of all relevant sections in each report was performed as a collaborative activity between the analysts. There were, however, important differences in the treatment of the documents. These stemmed from the way in which the Canadian and US agencies structure their findings. The NTSB provides a summary that distinguishes between probable causes and contributory factors in the following way:

> The National Transportation Safety Board determines that the probable cause of the collision between the Coast Guard patrol boat CG242513 and the small passenger vessel Bayside Blaster was the failure of the coxswain of the Coast Guard patrol boat to operate his vessel at a safe speed in a restricted-speed area frequented by small passenger vessels and in conditions of limited visibility due to darkness and background lighting. Contributing to the cause of the accident was the lack of adequate Coast Guard oversight of non-standard boat operations. (NTSB MAR-02/05)

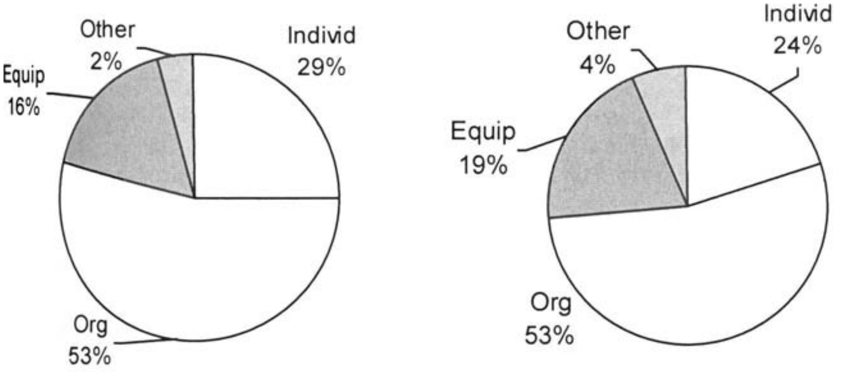
Canadian TSB reports contain a section entitled 'Findings as to Causes and Contributing Factors'. The analysis was less straightforward, however, because these documents did not explicitly separate causes and contributing factors. Each analyst, therefore, had to separate probable causes from contributory factors in TSB reports even though the distinctions were clearly presented in the NTSB reports. All subsequent stages were also performed in isolation until the results were available for comparison. We assigned each probable cause and contributory factor to a number of common categories. We did not use a pre-defined taxonomy. Each analyst created their own classification as they progressed through the incidents. As before, everyone involved in the project could assign any labels that they chose. The classification process raised several practical problems. For example, the following section is taken from an NTSB maritime report:

> Contributing to the amount of property damage and the number and types of injuries sustained during the accident was the failure of the U.S. Coast Guard, the Board of Commissioners of the Port of New Orleans, and International RiverCenter to adequately assess, manage, or mitigate the risks associated with locating unprotected commercial enterprises in areas vulnerable to vessel strikes (NTSB MAR-98/01)

This passage could yield three contributory factors; one associated with the U.S. Coast Guard, another with the Board of Commissioners of the Port of New Orleans and one with the International RiverCenter. Another analyst might identify three factors associated with a failure to adequately assess, manage, or mitigate the risks of vessel strikes. Conversely, this passage could yield the cross product of nine contributory factors where each agency failed in each of these three ways. We imposed no constraints on this issue except to agree that compound statements could be interpreted to yield several individual causes or contributory factors. It was left up to the reasoned judgement of each analyst on a case-by-case basis. The results of this process were then collated. There were some obvious differences in the terms used but there were also strong similarities. For instance, one analyst identified 'weather' as a contributory factor while another identified the 'environment' and so on. Where such disagreements occurred we used a process of discussion to agree on a common term to support comparisons between the classifications. Distinctions were preserved between different terms where no agreement could be reached between the analysts.

	Analyst C			**Analyst M**	
P – Probable Cause, C - Contributory	**P**	**C**		**P**	**C**
Design	**4**	7		**4**	**10**
Human Error	**11**	7		**9**	7
Maintenance	**2**	**2**		**2**	**0**
Company/Organisation	**12**	**6**		**15**	**6**
Regulatory	7	**9**		**4**	**15**
Weather	**1**	**0**		**2**	**0**
Equipment	**0**	**1**		**0**	**2**
Physical Structure	**0**	**0**		**2**	**0**
Industry	**0**	**0**		**0**	**1**
Unknown	**2**	**0**		**0**	**0**
Total	**39**	**32**		**38**	**41**

Table 1: Analysis of Causes and Contributory Factors in NTSB Maritime Accidents (1996-2006)



Analyst C

(22 Reports,

71 Causal & Contributory Factors)

Analyst M

(22 Reports,

79 Causal and Contributory Factors)

Figure 1: Pie-Charts of Causes and Contributory Factors in NTSB Maritime Accidents (1996-2006)

3 US National Transportation Safety Board Results

Table 1 and Figure 1 summarize the results of this classification process for both the probable causes and the contributory factors in the NTSB reports. The 22 incidents yielded a total of 39 and 38 probable causes for the two analysts. There were 32 and 41 contributory causes. Across all incidents, there was a mean of 1.75 causes per incident with a mean of 1.65 contributory factors per incident. The classification in Table 1 represents the product of an initial amalgamation, using the method described in the previous section. In contrast, Figure 1 uses an additional phase of generalisation that eases comparisons between the aviation data introduced in previous paragraphs and the results of the maritime analysis. This generalisation groups equipment failures and design issues. It also combines regulatory issues, maintenance problems, company specific factors and organisational issues. As can be seen, there are strong similarities both between the different analysts and between the NTSB maritime and aviation data sets. For example, the combined causal and contributory factors in the NTSB aviation study yielded 48% related to organisational factors, 37% to individual issues, 12% related to equipment and 3% to other factors (Johnson and Holloway, 2004).

The slight disagreement over the total number of contributory causes between the investigators might appear to be confusing given that the NTSB explicitly labels probable and contributory causes. As mentioned, however, some probable causes described several different problems. For example, the report into a collision between a US Coast Guard vessel and a small passenger boat contains the following argument;

> The National Transportation Safety Board determines that the probable cause of the collision between the Coast Guard patrol boat CG242513 and the small passenger vessel Bayside Blaster was the failure of the coxswain of the Coast Guard patrol boat to operate his vessel at a safe speed in a restricted-speed area frequented by small passenger vessels and in conditions of limited visibility due to darkness and background lighting. Contributing to the cause of the accident was the lack of adequate Coast Guard oversight of non-standard boat operations. (US NTSB MAR-02/05)

Analyst C classified the causes as human error and weather. Analyst M identified human error and the environment. The contributory causes were listed as 'organizational' by analyst C and regulatory by analyst M. As can be seen, this form of analysis depends upon a degree of subjective interpretation within the statements of probable cause and contributory factors. Hence Figure 1 indicates a surprising level of agreement between the analysts. Many NTSB reports yielded only a single probable cause. For instance, NTSB report MAR-02/03 contained the following summary:

> The National Transportation Safety Board determines that the probable cause of the grounding of the Finest was the failure of the vessel master to use appropriate navigational procedures and equipment to determine the vessel's position while approaching the Shrewsbury River channel. Contributing to the cause of the grounding was the lack of readily visible fixed navigational aids.

> Also contributing to the cause of the grounding was the failure of New York Fast Ferry to require the use of installed navigation equipment and to set guidelines for operations in adverse environmental conditions. (US NTSB MAR-02/03)

Both analysts identified the single probable cause as an instance of human error. In contrast to this simple case, our analysis identified a small number of incidents that proved to be extremely complex at least in terms of the number of causes and contributory factors. For instance, the NTSB report into the ramming of the Eads Bridge by barges in the Admiral St. Louis Harbor in Missouri provided the following summary of probable and contributory causes:

> The National Transportation Safety Board determines that the probable cause of the ramming of the Eads Bridge in St. Louis Harbor by barges in tow of the Anne Holly and the subsequent breakup of the tow was the poor decision-making of the captain of the Anne Holly in attempting to transit St. Louis Harbor with a large tow, in darkness, under high current and flood conditions, and the failure of the management of American Milling, L.P., to provide adequate policy and direction to ensure the safe operation of its towboats. The National Transportation Safety Board also determines that the probable cause of the near breakaway of the President Casino on the Admiral was the failure of the owner, the local and State authorities, and the U.S. Coast Guard to adequately protect the permanently moored vessel from waterborne and current-related risks (NTSB MAR-00/01)

Analyst C identified six probable causes; three regulatory failures, two organisational failures and one instance of human error. Analyst M classified seven causes; one environmental problem; one organisational issue; three regulatory problems; one company issue and an instance of human error. Neither analyst identified any contributory factors. Such findings illustrate considerable differences in interpretation and classification. Given the limited sample size and the small number of analysts it is difficult to draw firm conclusions about the analysis of particular incidents. However, the growing body of evidence from this and previous studies does illustrate that such incidents are the exception rather than the norm. This methodology can yield a surprising level of agreement in the identification of causal and contributory factors in official investigation reports.

Both analysts identified a large number of systemic causes and contributory factors throughout the sample of NTSB reports. Overall managerial or organisational failures accounted for approximately 53% of all probable causes and contributory factors. Individual forms of 'human error' only represented 27% of the total. Equipment failures came to 17% and 3% fell into the 'other' classification. Even after the results from our previous aviation study, these findings came as a considerable surprise. In particular, we had anticipated a higher proportion of equipment related problems in the maritime industry. However, the NTSB reports seem to reveal the commitment that investigators within this agency have to look beyond immediate causes and investigate the organisational and regulatory issues contributing to incidents and accidents.

As mentioned, the 22 NTSB reports in our sample yielded approximately 70 causes and contributory factors for each analyst. This provided useful insights about individual and organisational factors when aggregated across the decade. However, the sample arguably yielded insufficient evidence to support clear conclusions about trends between 1996 and 2006. This point can be illustrated by the Bar Chart in Figure 2. The diagram provides a year-by-year distribution of causes and contributory factors for our sample of accident reports. The dates refer to the year in which the documents were published rather than to the accidents themselves, this follows the convention used in Appendix A and enables cross referencing with the NTSB library. The number of causal and contributory factors in each category is strongly influenced by individual accidents. Longer term trends are obscured by the characteristics of particular incidents. For example, a single accident in 2000 accounted for all of the regulatory and organisational causes in that year. Similarly, the same accident was caused by both instances of human error recorded in 2001.

The Bar Chart in Figure 2 also illustrates the difficulty of visualising the results of an analysis into the causal and contributory factors in major accident reports. Simple graphs cannot easily convey the changing proportions of causes in different categories when the number of factors is partly determined by the number of accident reports that are issued in each year. In our sample of NTSB reports, the frequency of particular causal factors is strongly determined by the number of reports which varies from none in 2003 to 5 in 2002. Figure 3, therefore, uses a J-H area graph to map the changing percentage of causes and contributory factors for each year from 1996 to 2006. This helps to ease the visualisation problems by normalising across accident frequencies, although the count is still shown on the X-Axis. As can be seen, the lack of any coherent pattern confirms our previous argument based on the Bar-Chart in Figure 2, that the individual causes and contributory factors of a small number of accidents obscures any longer term trends over the sample. Although this visualisation provides a normalised view of causes over time, it does not resolve the problems associated with a limited sample size. The following section, therefore, provides a more sustained analysis of 160 Canadian TSB maritime reports compared with only 22 in the NTSB sample.

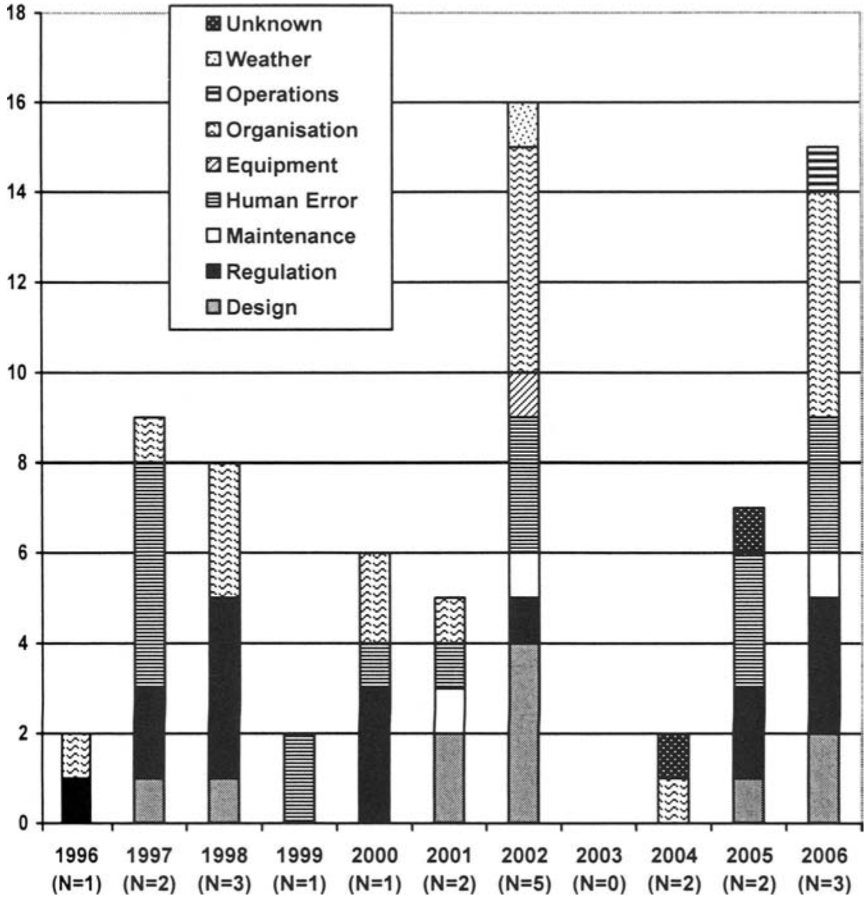


Figure 2: Distribution of Causes and Contributory Factors in NTSB Maritime Reports
by Year (Analyst C)

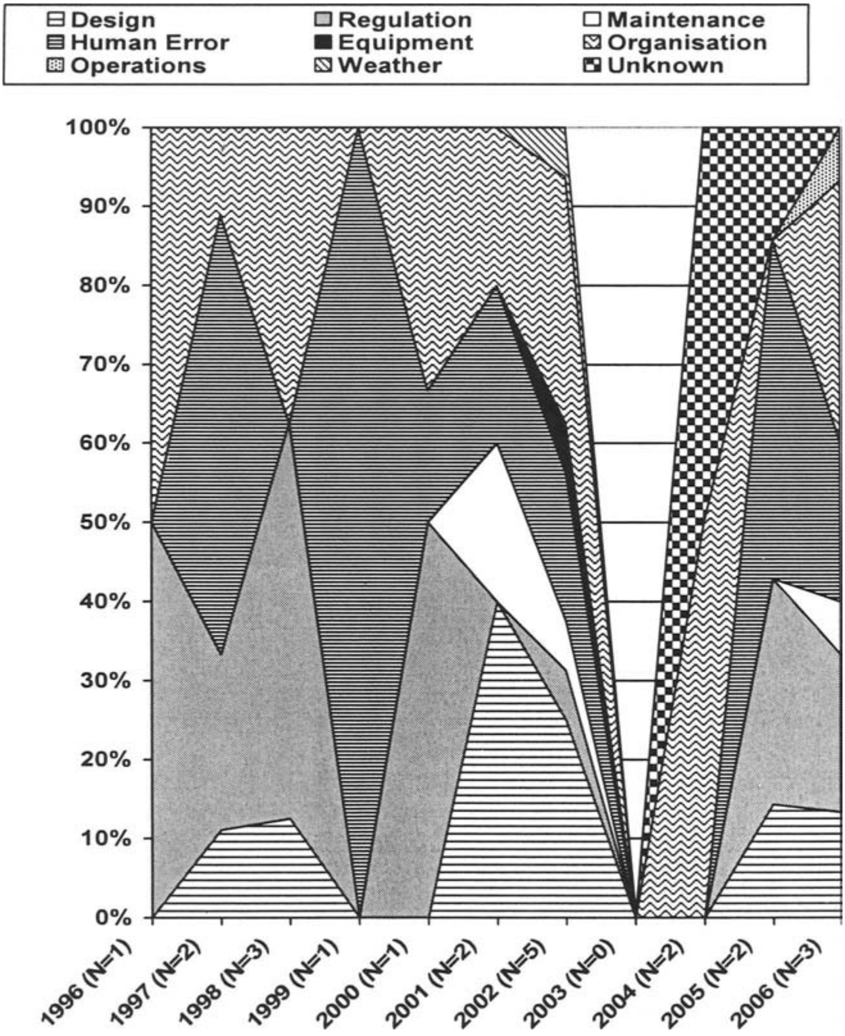


Figure 3: J-H Graph of Causes & Contributory Factors in NTSB Maritime Reports by Year (Analyst C)

4 Canadian Transportation Safety Board Results

We were anxious to determine whether the US NTSB was atypical in the prominence of regulatory and organizational factors in their major maritime and aviation accident reports. The results of our previous study had already identified some differences in the aviation data between Canada and the USA. As mentioned, these differences stem from the traffic patterns in each country. They may also be due to differences in the training of investigators and the reporting procedures used in each country (Johnson, 2003). We were, therefore, anxious to determine whether these patterns could also be seen in maritime accident reports. Our work proceeded in a similar manner to that of the NTSB sample. The first stage was to make our initial selection of incidents from the many thousands of adverse events that are reported to the TSB each year. We focused on the longer more sustained accident reports; that is, those containing numbered chapter headings. These did, however, include near miss incidents as well as events leading to multiple fatalities. We identified a far larger sample compared to either our aviation datasets or to the NTSB major maritime incident reports. In the previous studies, we had used a heuristic to cut down the TSB aviation corpus so that we only focussed on the most serious incidents and accidents. This left a total sample of 27 TSB aviation documents compared to 26 reports from the NTSB. In contrast, our more ambitious maritime study yielded 22 accident reports from the NTSB and 160 marine reports from the TSB. The problems of obtaining comparable samples might seem like a relatively trivial methodological issue. However, the different ways in which the NTSB and TSB group their major accident reports has important consequences for anyone attempting to identify patterns in the causes of adverse events across different countries. It can be hard to make comparisons between incidents in different countries and this can impair the exchange of lessons learned from previous failures.

The TSB documents included sections on "Findings as to Causes and Contributing Factors", "Findings as to Risk", and "Other Findings". We focussed on the sections detailing causes and contributory factors. This task was complicated because some reports used a slightly different format with two sections entitled "Causes" and "Findings". As might be expected, we focused on the section describing the causes of the adverse event. As before, we independently categorised the probable causes and contributory factors. There was no expectation that each analyst would use the same categories that had emerged from the analysis of the NTSB maritime reports. This posed several problems that had not arisen during the previous studies. For example, many hours of analysis were required to work through all of the 160 reports. It was difficult for analysts to ensure that they applied the same classification criteria at the end of the period as they had used at the start of their analysis. As we shall see, the very diversity of the incidents included in this larger sample also forced the analysts to develop a far wider range of categories for the TSB sample. Further problems stemmed from the way in which the TSB group together probable causes and contributory factors within their reports. For instance, the section of a report into the 'Capsizing and Loss of Life on a Small Fishing Vessel *Cap Rouge II* off the Entrance to Fraser River, British Columbia' contains the following list:

3.1 Findings as to Causes and Contributing Factors

1. Inherent transverse stability was progressively reduced by structural additions and the installation of more and heavier fishing gear, including the adoption of a "West Coast" seine net of 7.4 tonnes, all of which were located at or above the main deck level.
2. The installation of additional gear and its effects on stability were not monitored or assessed by a suitably qualified person, nor brought to the

attention of Transport Canada (TC) inspectors, between or during routine quadrennial inspections.

3. The watertight integrity of the main deck was compromised by the ineffective gaskets of five flush-fitting manhole covers, which resulted in extensive downflooding, a marked increase in after trim, and reduced transverse stability.
4. Because of their limited knowledge of basic principles of trim and stability, the additional weight of the seine net, the inherent heel to starboard, the routine presence of water on deck, and the towing resistance of the seine skiff were not considered by those on board the *Cap Rouge II* to present any undue risk to vessel operation.
5. The vessel lost transverse stability due principally to the cumulative free surface effects of water shipped and retained on the main deck and other liquids in four partially full fish holds, four fuel tanks, a freshwater tank, and the lazarette.
6. The rapidity of the capsizing precluded orderly abandonment of the vessel.

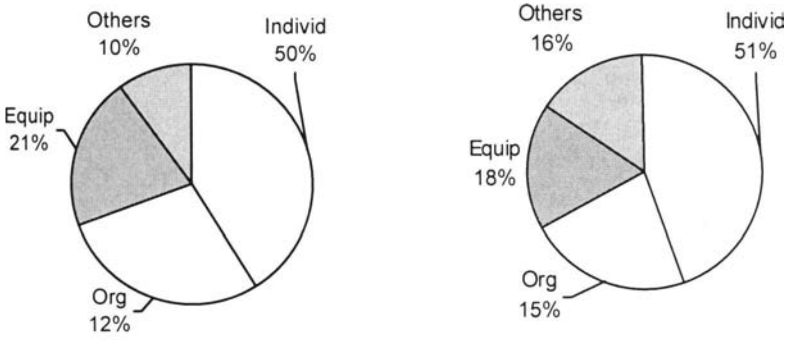
(TSB report M02W0147)

As can be seen, the TSB provide no explicit indication between causes and contributory factors in this list. Each analyst, therefore, had to arrive at this classification independently. In consequence, analyst C identified two causal factors of design and regulation. Analyst M identified design and equipment failure. Analyst C found four contributory factors. These included maintenance, human error, design and 'other'. Analyst M identified human error; environmental factors and regulatory issues.

	Analyst C		**Analyst M**	
	Probable cause	**Contributory Factor**	**Probable cause**	**Contributory Factor**
Clothing	**0**	**0**	**1**	**2**
Company/Organisation	**16**	**54**	**15**	**60**
Design	**33**	**50**	**23**	**45**
Emergency responders	**0**	**0**	**0**	**3**
Environment/Weather	**30**	**20**	**35**	**27**
Equipment failure	**31**	**12**	**32**	**8**
Health	**0**	**0**	**0**	**1**
Human Error	**106**	**146**	**120**	**142**
Maintenance	**15**	**21**	**10**	**18**
Operations	**33**	**23**	**8**	**8**
Physics	**0**	**0**	**15**	**11**
Regulator	**3**	**9**	**2**	**12**
Unknown	**5**	**2**	**4**	**2**
Others	**0**	**4**	**0**	**0**
Total	**272**	**341**	**265**	**339**

Table 2: Causal Information in the TSB Maritime Dataset

The decision to allow each analyst to identify multiple causes and contributory factors within the lists presented by the TSB led to some differences in the analysis provided by each investigator. The 160 maritime reports yielded a total of 272 probable causes for analyst C and 265 for analyst M. Analyst C also identified 341 contributory factors while analyst M found 339. Table 2 provides a more detailed distribution of these causes and contributory factors within the various categories that were induced during our analysis. The variance between the investigators could have been reduced if a more formal method for distinguishing causes from contributory factors had been used. For instance, the PRISMA analysis technique provides a flow chart that investigators can work through to identify the role that various factors can play in an incident or accident (Johnson, 2003). At the start of the study, we decided not to use this approach because the development of appropriate root cause analysis techniques remains an active area for research. We are currently exploring the impact that more formal techniques might have on the results of our analysis.



Analyst C (160 reports, 613 causes/factors)

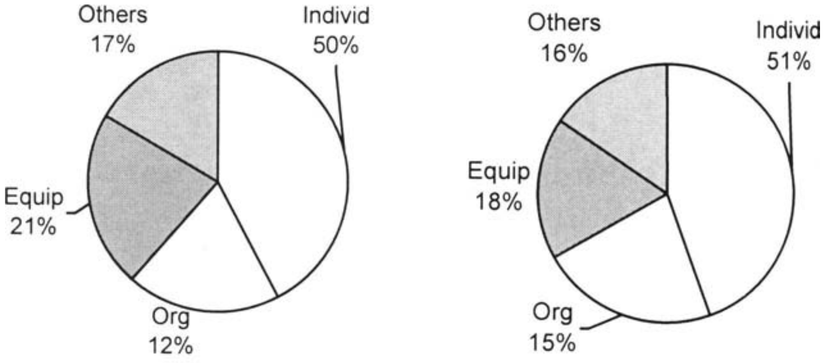
Analyst M (160 reports, 604 causes/factors)

Figure 4: Categorisation of Causes and Contributory Factors in the TSB Maritime Dataset

Figure 4 presents an overview of the more detailed classification illustrated in Table 2. By combining causes and contributory factors we can abstract away from some of the individual classification differences that were mentioned in the previous paragraphs. Company/organisational issues were grouped with regulatory factors, maintenance and operations. Equipment related causes and contributory factors were combined with design issues. ‘Others’ included environmental conditions, meteorological factors, unknown and other issues. As can be seen, the TSB maritime reports show a pattern that is very similar to the results from our previous studies of the TSB reports for major aviation accidents. Our earlier work on NTSB aviation reports showed that approximately 50% of all causes and contributory factors could be related to individual ‘error’ within our sample of aviation reports.

20% stemmed from equipment related issues, 22% to organisations and 8% to other factors. Here we can see a remarkably similar pattern in the maritime incidents, especially between analysts C and M. The initial analysis indicates that individual error plays a more prominent role in the TSB dataset than in the NTSB and that this pattern reflects the results from our previous aviation study. This can also be explained in similar terms. For example, the Canadian reports contain many incidents involving small charter vessels and owner-operators. In such cases, there is less opportunity for larger management structures and external organisations to create the preconditions for failure. Many of these incidents occur in remote locations well away from busy, regulated passages. Finally, it might also be argued that the prominence of individual error is an artefact of the different analytical techniques being employed by each agency (Johnson, 2003)

It is important not to exaggerate the prominence of human error in our study. The 50% of causal and contributory factors identified for individual failure in Figure 4 is relatively low compared to most estimates made in the wider human factors literature. It should also be remembered that this range is still much higher than our results for the NTSB dataset. Within the Canadian incidents, Figure 5 illustrates the consistency of the analysis by aggregating across both analysts but distinguishing between the proportion of causes and contributory factors in each of the four high level categories. This is an important analysis because it shows that there is no particular focus on individual error as a primary cause rather than a contributory factor nor can it be argued that the TSB investigators focus on organisational issues as contextual issues rather than more 'direct' causes.



Causes

(Both analysts, 160 reports, 537 causes)

Contributory Factors

(Both analysts, 160 reports, 604 factors)

Figure 5: Categorisation of Causes and Contributory Factors in the TSB Maritime Dataset

Figures 6 and 7 use 'J-H Graphs' to map the distribution of causes and contributory factors across the study period 1996-2004. This end-point reflects the latest collection of reports released by Transport Canada at the time of writing (late 2006). The y-axis shows the percentage of reports in each category, which is mapped as a percentage of the total causes for that year in Figure 6 and as a percentage of total contributory factors in Figure 7. As can be seen from the x-axis, this helps to normalise for a strong decline in the number of maritime reports issued; from 49 in 1996 to only 4 in 2004. A number of arguments can be used to explain this decline. The fall may reflect a genuine improvement in maritime safety over the period studies. This, in turn, may reflect changes in market structure as high-risk, single operator work has arguably decreased. Alternatively, the decline in major accident reports may reflect institutional changes in the investigation and reporting of major accidents by Transport Canada.

The increasing focus on organisational factors is readily apparent in Figures 6 and 7, from relatively small beginnings at the start of the sample to an increasing proportion of the causes and contributory factors in more recent reports. The focus on human error seems to have fluctuated from year to year. As with the NTSB results this may simply reflect the influence of particular adverse events on the totals for a particular year. However, there does appear to be a declining focus on individual error as a contributory factor over the study period even though Figure 3 shows that the proportion of contributory factors related to human error is comparable to the proportion of causes in this classification. Further work is required to determine whether this is part of a sustained trend within the TSB reporting of maritime accidents. In particular, it is important also to identify causal explanations for any patterns that are sustained. For example, Ayeko (2002) has described the influence that Reason's (1997) work on organisational causes of accidents has had upon investigators' training with the TSB. It could be argued that systematic changes in the causal analysis of major accidents should be apparent in the J-H graphs as increasing numbers of inspectors are exposed to these initiatives.

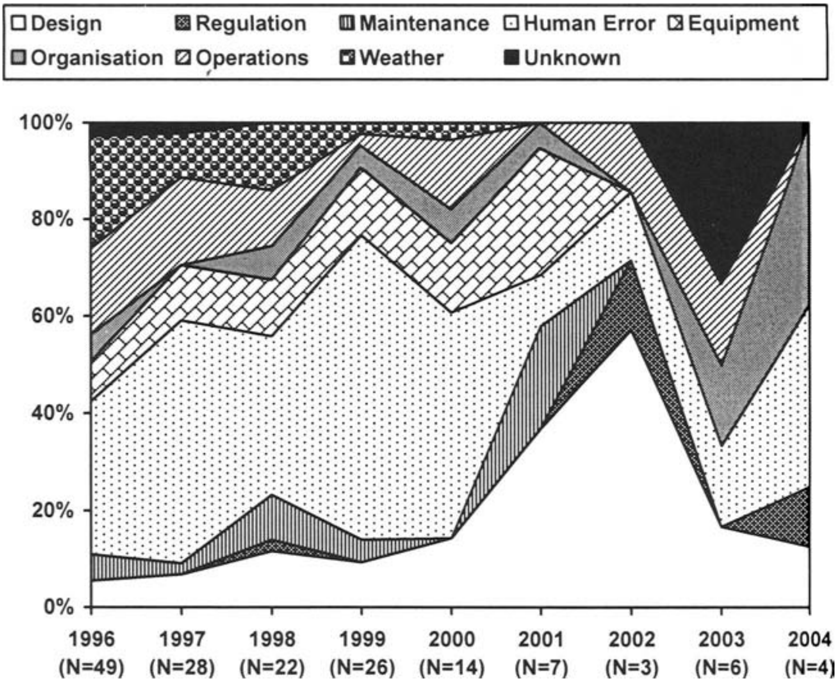


Figure 6: J-H Graph of Causes in TSB Maritime Reports by Year (Analyst C)

5 Conclusions

We have described the results of an independent analysis of the primary and contributory causes of maritime accidents in both the United States and Canada between 1996 and 2006. The purpose of the study was to assess the comparative frequency of a range of causal factors in the reporting of adverse events. Our results suggest that many of these high consequence accidents were attributed to human error. However, the overall proportion was very much smaller than has been suggested elsewhere in the human factors literature. A large number of reports also mentioned wider systemic issues, including the managerial and regulatory context of maritime operations. Based on these results we believe that is inaccurate to assert, as some have, that most investigations stop as soon as they find someone to blame, or that organizational causes are usually ignored. There are wider implications. For example, some have used the supposed predominance of human error as a primary cause in accidents to justify automation as a means reducing operator intervention. By restricting the scope for human 'error', it should be possible to reduce the overall accident rate (Johnson, 2003). This paper undermines these arguments by challenging the claimed prominence of human error in incidents and accidents.

In order to communicate our findings, we have introduced J-H graphs to visualise the proportion of causes and contributory factors associated with human error, equipment failure and other high level classifications in longitudinal studies of accident reports. These diagrams provide means of normalising across the causes and contributory factors that lead to rare and atypical events. The J-H charts show that our limited sample of NTSB reports could not be used to identify emerging patterns in the proportion of accidents associated with human error, equipment failure or organisational issues for twelve month intervals from 1996 to 2006. However, it is possible to discern a rise in the proportion of organisational issues that are identified as contributory factors in a broader sample of TSB maritime accident reports from 1996 to 2004, which includes the most recent publications.

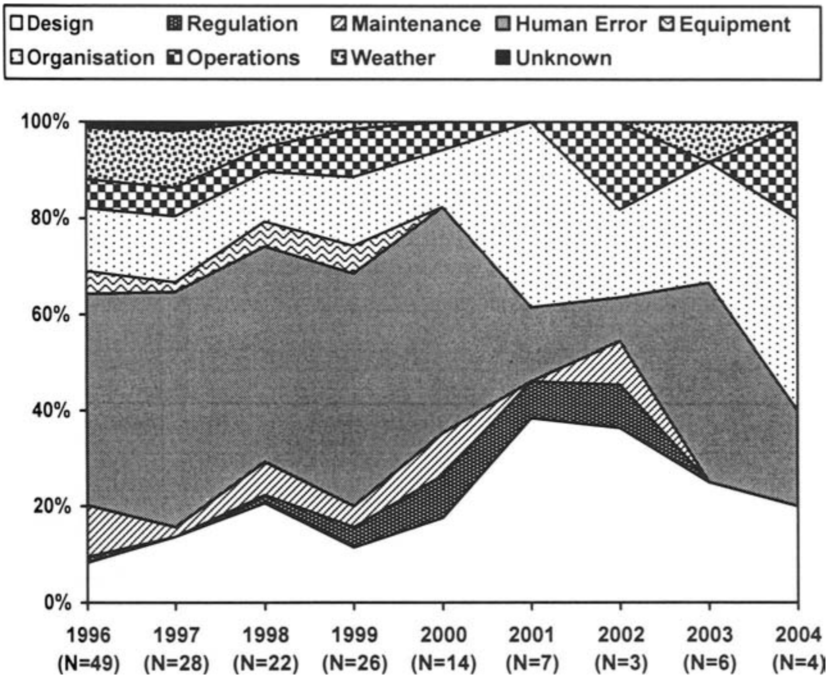


Figure 7: J-H Graph of Contributory Factors in TSB Maritime Reports (Analyst C)

A key finding from our research is that investigatory organizations show a similar distribution of causes and contributory factors between individual, organizational and equipment failures across different modes of transportation. Hence, there are strong similarities between the prominence of organizational factors in the NTSB reports in aviation and the maritime industries. Similarly, close comparisons can be made between the classifications for the TSB aviation and maritime reports. There are, however, considerable differences between the NTSB and TSB distributions in both modes. We conclude that these results are due to differences in the operational profile in each country. For instance, the TSB reports document a larger number of incidents involving private pilots and owner-operator vessels in remote areas than their NTSB counterparts. These differences may also be due to different analytical techniques, such as the TSB Integrated Safety Investigation Methodology approach (Ayeko 2002, Johnson 2003). Further work is required to more accurately trace the impact that investigator training has on the conclusions of accident and incident reports. Such studies must also consider the knock-on effects that these findings will have on the engineering of safety-critical systems across many different industries.

Acknowledgement

Michael Holloway's participation was funded by a NASA Langley Research Center Floyd Thompson Fellowship.

References

M. Ayeko, Integrated Safety Investigation Methodology (ISIM) - Investigation for Risk Mitigation. In C.W. Johnson (ed.) Proceedings of the First Workshop on the Investigation and Reporting of Incidents and Accidents, 115-126, Glasgow University Press, Scotland, 2002.

C. M. Holloway and C. W. Johnson, On the Prevalence of Organizational Factors in Recent U.S. Transportation Accidents. In Proceedings of the 23rd International System Safety Conference, 22-26 August 2005, San Diego, California, International Systems Safety Society, Unionville, VA, USA, 2005.

C. M. Holloway and C. W. Johnson, Why System Safety Professionals Should Read Accident Reports. In T. Kelly (ed.) The First IET International Conference on System Safety, Institute of Engineering and Technology, Savoy Place, London, pages 325-331, ISBN 0-86341-646-2, 2006.

C. W. Johnson and C. M. Holloway, 'Systemic Failures' and 'Human Error' in Canadian TSB Aviation Accident Reports between 1996 and 2002, In A. Pritchett and A. Jackson (eds.) HCI in Aerospace 2004, EURISCO, Toulouse, France, pages 25-32, 2004.

C.W. Johnson and C.M. Holloway, A Longitudinal Analysis of the Causal Factors in US Aviation Accidents, Department of Computing Science, University of Glasgow, Scotland. In press, contact the authors for preprints.

C.W. Johnson, The Failure of Safety-Critical Systems: A Handbook of Accident and Incident Reporting, Glasgow University Press, Glasgow, 2003.

UK Marine Accident Investigation Branch, Bridge Watch Keeping Safety Study, Safety Study 1/2004, UK MAIB, Southampton, United Kingdom. Available on: http://www.maib.dft.gov.uk/cms_resources/dft_masafety_030084.pdf

J. Reason, Managing the Risks of Organisational Accidents, Ashgate, Aldershot, 1997.

Appendix A: Data Sources Used in the Study

An important aim of our study was to enable others to replicate our work. The NTSB and TSB documents in our data set included all major maritime reports between 1996-2006.

For the NTSB they were:

> MAR-96/01, MAR-97/01, MAR-97/02, MAR-98/01, MAR-98/02, MAR-98/03, MAR-99/01, MAR-00/01, MAR-01/01, MAR-01/02, MAR-02/01, MAR-02/02, MAR-02/03, MAR-02/04, MAR-02/05, MAR-04/01, MAR-05/01, MAR-05/02, MAR-06/01, MAR-06/02, MAR-06/03.

We also included the NTSB report into the allision between the tow boat *Robert Y. Love* with Interstate 40 on 26th May 2002. This document appears in both the marine and highways archive, following the NTSB's approach we use the highway identifier (HAR-04/05).

The TSB reports in our data set were:

> M96C0022, M96C0032, M96C0032, M96C0056, M96C0062, M96C008, M96C0090, M96C0093, M96F001, M96F0023, M96F0025, M96H0016, M96L0006, M96L0017, M96L0037, M96L0043, M96L0052, M96L0059, M96L0069, M96L0083, M96L0111, M96L0112, M96L0116, M96L0131, M96L0142, M96L0146, M96L0148, M96L0156, M96M0002, M96M0038, M96M0090, M96M0128, M96M0132, M96M0144, M96M0150, M96M0176, M96M0178, M96N0047, M96N0061, M96N0063, M96W0025, M96W0061, M96W0100, M96W0109, M96W0175, M96W0183, M96W0187, M96W0243, M96W0250, M97C0013, M97C0055, M97C0057, M97F0002, M97F0027, M97L0019, M97L0021, M97L0030, M97L0035, M97L0050, M97L0076, M97M0005, M97M003, M97M0094, M97M0141, M97N0067, M97N0071, M97N0073, M97N0099, M97N0129, M97W0022, M97W0044, M97W0048, M97W0152, M97W0194, M97W0197, M97W0236, M98C0004, M98C0015, M98C0026, M98C0040, M98C0046, M98C0066, M98F0009, M98F0023, M98F0039, M98L0097, M98L0120, M98L0139, M98L0149, M98L0165, M98M0003, M98M0061, M98M0078, M98N0001, M98N0064, M98W0019, M98W0045, M98W0245, M99C0003, M99C0005, M99C0008, M99C0016, M99C0019, M99C0027, M99C0048, M99F0023, M99F0038, M99F0042, M99L0011, M99L0098, M99L0099, M99L0126, M99M0062, M99M0142, M99M0161, M99W0033, M99W0058, M99W0078, M99W0087, M99W0095, M99W0116, M99W0133, M99W0137, M99W0145, M00C0026, M00C0033, M00C0053, M00C0069, M00H0008, M00L0034, M00L0114, M00N0098, M00W0005, M00W0044, M00W0059, M00W0230, M00W0265, M00W0303, M01C0033, M01C0054, M01L0080, M01L0112, M01M0100, M01N0020, M01W0006, M02C0030, M02W0147, M02C0064, M03L0026, M03C0016,

M03W0073, M03N0050, M03M0077, M03L0124, M04L0050, M04L0066, M04L0099, M04L0105.

Appendix B: Additional Graphs

Figure B-1 illustrates the year by year breakdown on causes and contributory factors for analyst C across the NTSB sample using the high-level classifications that were introduced in previous sections. As can be seen, this confirms the lack of any apparent pattern in the small number of reports (22) even though they yield more than 70 causal/contributory factors.

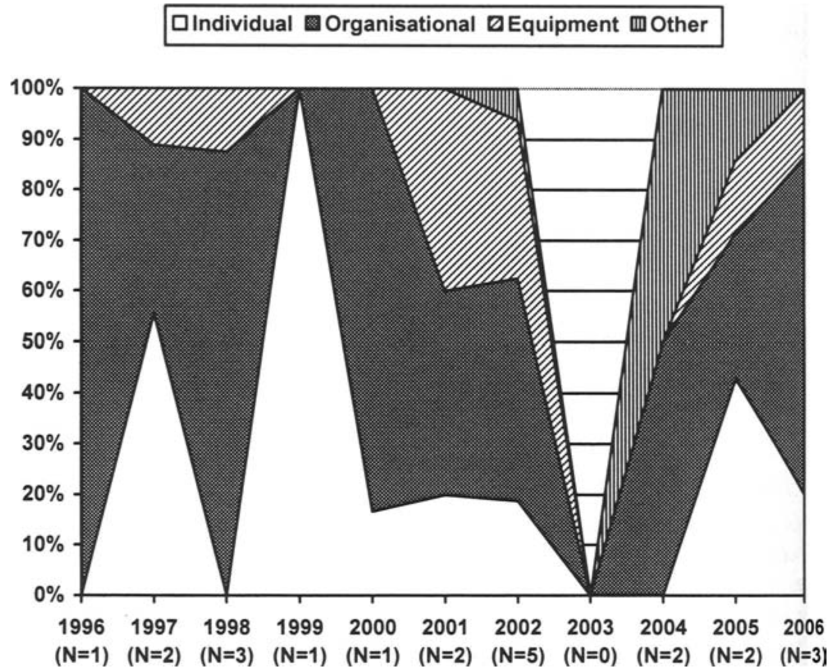


Figure B-1: J-H Graph of Causes & Contributory Factors in NTSB Maritime Reports by Year (Analyst C)

A Proactive Approach to Enhancing Safety Culture

Liz Beswick and Jonathan Kettleborough
British Energy
Barnwood, Gloucester, England

Abstract

Within high impact failure organisations [nuclear, petrochemicals, aerospace] the need to create an appropriate safety culture is paramount. Whilst on the outside it may seem that production is the key - making more electricity, drilling more wells and flying more miles - the underlying truth is that a safe organisation is a productive and profitable one and that the cultural values which underpin safety also support commercial success.

And it should be remembered that culture is not wholly self-sustaining; it takes time and effort to sustain a culture, and as outlined in this paper, those at the top of an organisation have a pivotal role to play.

This paper explains the importance of safety culture to British Energy and provides the drivers for continued improvement in safety culture - a safe organisation as a productive and profitable one.

The paper outlines the elements of the British Energy safety culture enhancement programme, in line with the International Atomic Energy Agency (IAEA) definition and model for safety culture reported in Safety Series Report No. 75-INSAG-4. It also draws on INSAG-15 and the World Association of Nuclear Operators (WANO)/Institute of Nuclear Power Operators (INPO) Principles for a Strong Nuclear Safety Culture, together with lessons learnt from a number of safety significant events such as Chernobyl and Challenger.

1 Introduction

With many disasters there's a focus on the moments immediately before the critical instant – the ones we remember and the ones the worlds press reports over and over again. Yet the seeds of failure are more often than not sewn years before; laying the foundation for the inevitable consequences that follow.

Culture is a strange thing. We see it every day; it drives our thinking, our values, what we eat, what we wear, how we behave and what we believe in. Yet within an organisation we need to bring together a widely diverse set of individuals who outside of the organisation live to their own cultural norms. At work they need to understand, adapt and adopt the cultural norms of the organisation, and this is exceptionally polarised in organisations where the cost of failure is high.

Within high impact failure organisations [nuclear, petrochemicals, aerospace] the need to create an appropriate safety culture is paramount. Whilst on the outside it may seem that production is the key - making more megawatt hours, drilling more wells and flying more miles - the truth is that safe organisations are also productive and profitable ones.

And it should be remembered that culture is not wholly self-sustaining; it takes time and effort to establish the correct culture, and enormous effort to maintain it, with much of that effort coming from senior levels within the organisation as we shall see later.

LIMITATION OF LIABILITY - Whilst British Energy Generation Ltd believes that the information given in this document is correct at the date of publication it does not guarantee that this is so, nor that the information is suitable for any particular purpose. Users must therefore satisfy themselves as to the suitability of the information for the purpose for which they require it and must make all checks they deem necessary to verify the accuracy thereof. British Energy Generation Ltd shall not be liable for any loss or damage (except for death or personal injury caused by negligence) arising from any use to which the information is put.

2 Organisation and Cultural Drivers

Let's a take a moment and consider the following: Barings Bank, Chernobyl and Challenger.

Barings was not some backwater financing company, it was the Queen's bank. Chernobyl was the jewel in the crown of Soviet nuclear production and Challenger was the iconic representation of the USA space programme. What really grabbed the world's attention was the manner of failure; with Barings it was the actions of a single trader based at a small office in Singapore. With Challenger it was the failure of an O-ring, possibly one of the cheapest components on the shuttle, and with Chernobyl it was the actions of a few people which caused such devastation.

If we all have the right tools, experience and time to do a job then we usually manage to get it right. Pressures of time, pressures from peers and pressures to deliver, no matter what, are key drivers that can lead to organisational breakdown.

So the culture of the organisation is paramount in allowing a safety culture to exist, be nurtured and to grow. Just as an organisation can willingly accept a questioning attitude and respond to it, it can also stifle such attitudes and turn the culture into one of 'Just do it!'

The impact of organisational culture is therefore an immense driver on establishing, nurturing and sustaining a strong safety culture environment.

We'd now like to focus on work that has been carried out at British Energy, the UK's largest provider of nuclear power generation where a safety culture enhancement programme has been run to keep the company at the forefront of safety culture adoption.

3 Safety Culture and the Nuclear Industry

Safety culture as a defined concept wasn't always part of the nuclear industry, so let's start by looking at when and why it was introduced.

The introduction of safety culture began following the Chernobyl accident in 1986. The term 'safety culture' was introduced by the International Atomic Energy Agency (IAEA) to the nuclear industry in 1991 with its production of Safety Series Report No. 75/INSAG-4 on Safety Culture. It was also included by both nuclear power overseers, the Institute of Nuclear Power Operations (INPO) and the World Association of Nuclear Power Operators (WANO), both of whom included safety culture in their measures for what makes a good nuclear power operator.

The definition of safety culture provided in INSAG 4 is adopted for the purpose of this paper:

> 'That assembly of characteristics and attitudes in organisations and individuals which establishes that, as an overriding priority, nuclear plant safety issues receive the attention warranted by their significance'.

The WANO/INPO Performance Objectives for safety culture expand on this definition:

> 'Individuals at all levels of the organisation consider nuclear plant safety as the overriding priority. Their decisions and actions are based on this

> priority and they follow up to verify that nuclear safety concerns receive appropriate attention. The work environment, the attitudes and behaviours of individuals and the policies and procedures foster such a safety culture.'

As already noted, the Chernobyl accident led to the use of the term 'safety culture' in the nuclear industry. Subsequent events in the nuclear and other safety critical industries have highlighted the importance of safety culture as a root cause or contributory factor. Other examples in the nuclear industry include Tokaimura (1999), Davis-Besse (2002), and THORP (2005). Examples in other safety critical industries include Columbia (2003), Texas City (March 2005) and Buncefied (December 2005).

Returning to Chernobyl for a moment – this was one of the most prominent industrial disasters the world has known, and certainly is the largest nuclear disaster ever. To the people on the outside it's probably hardly surprising that this event occurred. Some aspects of Russian nuclear technology may have been criticised by the West, but aren't these the same people who invented the satellite, who were key contestants in the space race and who ran a space station for many years. Here are people who understand engineering and clearly have the competencies to design, build and operate nuclear facilities; after all, they've had nuclear power plants in submarines for years. So where did it all go so wrong?

As with other safety influenced industries, the heart of the problem is the culture which takes over the organisation and allows it to fail. Consider for a moment the jobs you might do around your house; have you ever wired a plug knowingly using the wrong fuse because you knew it would be 'OK'? Have you ever driven faster than is really necessary because you were late and have you ever cut yoúr lawn or hedge without the use of eye protection, gloves and steel toe-capped boots? Of course you have; we all have. But then if you got it wrong at home what's the worst that's going to happen, and be honest how likely is it to happen?

Now translate that thinking into an organisation. Fitting the wrong fuse may initially seem a minor issue but what if that means a safety circuit does not trip out as designed – leading to disastrous consequences?

Clearly we all want major safety related organisations to behave in a thorough and responsible manner; but do they? Do they always put safety before production and do they actively encourage staff to challenge operational performance and established practice? Just think, if the engineers in the control room at Chernobyl had said 'no' then a disaster would have been averted. The same challenges could have averted other disaster. A cultural change would have made all the difference between success and failure

And in the nuclear world, failure is not something we contemplate, it is something we work every day to ensure does not happen.

The worldwide events discussed above have provided drivers for improvements in safety culture in all safety critical industries, especially nuclear and, together with a desire to address all aspects of performance improvement provided the impetus to develop and implement a company wide nuclear safety culture enhancement programme within British Energy.

4 Safety Culture Enhancement Plan

The British Energy safety culture enhancement plan is part of an overall wide-ranging Performance Improvement Programme within the organisation. This enhancement plan contains a set of activities which actively promote and develop a positive nuclear safety culture. In developing these activities it was necessary to understand that the broader concept of safety culture recognises that nuclear safety is upheld by a continuing search for the weaknesses in the safety management systems – including people, plant and processes. It is manifest by an unwavering drive for excellence and quality, intolerance to poor results and an open pursuit of the underlying causes so that they are correctly identified and rectified.

The key activities for enhancing nuclear safety culture within the organisation were:

- Development and roll out of a Nuclear Safety Policy, based on the definition and model of safety culture outlined in INSAG 4
- Company wide self assessment of Nuclear Safety Culture, to provide a baseline for the enhancement programme and identify specific areas for improvement
- Ongoing effectiveness reviews and assessments to measure progress and identify further activities for continuous improvement
- Proactive consideration of safety throughout the organisation through the development, implementation and reinforcement of daily safety messages based around a number of agreed safety themes, to focus the organisation on common safety issues and foster a questioning attitude to safety.
- Development and roll out of a company wide policy for procedural use and adherence, to align the organisation to a common set of requirements as a starting point for continuous improvement
- Development and delivery of company wide nuclear safety culture workshops, to raise awareness of safety culture throughout the organisation
- Development and implementation of local improvement plans in every business unit, to address the areas for improvement arising from the nuclear safety culture self assessment and the nuclear safety culture workshops.

A number of other key improvement activities have also been implemented throughout the organisation as part of the Performance Improvement Programme and provide a platform for the enhancement of nuclear safety culture. The main areas of continuous improvement that support the enhancement of nuclear safety culture are:

- Development of the Corrective Action Programme, to allow all personnel to raise safety and quality issues. Enabling the organisation to address near misses and precursors to safety significant events and foster the concept of a learning organisation

- A formal Operational Decision Making process, to provide a structured and documented process for making and communicating decisions that impact nuclear safety
- Enhanced focus and resource for staff training
- A Human Performance enhancement programme to train leaders in Task Observation & Coaching and educate the workforce in error prevention tools and techniques
- Leadership programme to underpin attitudes, behaviours, tools and techniques
- Equipment Reliability Programmes
- and, System Health programmes, both of which are designed to keep the right plant working in the right way at the right times

5 Nuclear Safety Culture Workshops

The critical activity at the core of the safety culture enhancement programme described above was the development and roll out of systematic and structured nuclear safety culture workshops to educate all levels of the organisation, followed by activities to encourage and embed a sustained improvement. Deployment of the overall plan commenced in January 2006 at the first of the eight nuclear power stations in the British Energy Fleet with parallel roll out in the Central Support Functions over the period July to October 2006.

Based on what has become know as the four 'Es' model, the enhancement process consists of:

- **Engagement** of each business unit (Power Station or Central Support Function) in the requirements and expectations for the programme roll out
- **Education** of all members of the business unit
- **Encourage** phase where the business units start to put into practice the lessons they have learned from the workshops
- **Embed** phase consisting of ongoing effectiveness reviews and assessments to identify areas for continued improvements and longer term activities to sustain a strong nuclear safety culture

The above phases are expanded on below.

5.1 The Engage Phase

It was imperative that each station or business unit had a full understanding of the overall programme and the implications for roll out within the business unit in advance of delivery. Ownership by the business unit, allocation of resources to deliver the programme and integration with the work planning processes were key activities within the Engage Phase. Whilst generic workshop packages had been developed for delivery to all leaders and staff, a locally generated Encourage Phase plan was agreed by each business unit during the Engage phase to take account of other local activities and specific findings from the local safety culture assessment/baseline.

5.2 The Educate Phase

At the heart of the educate phase was the roll out of company wide workshops to all core and major support staff and partners. These consisted of mixed groups of personnel to enhance the learning environment by identifying and sharing common barriers to a strong nuclear safety culture and identifying how improvements could be made in individuals own behaviours and work activities.

This education actively drew on INPO, WANO and IAEA practices (INPO/WANO Performance Objectives and Criteria 1999 and Principles for a Strong Nuclear Safety Culture 2004, INSAG-4 1991, INSAG-15 2002).

The workshops used the principle of experiential learning via the use of a number of worldwide nuclear case studies to identify and understand the impact of safety culture as an underlying root cause to significant events or significant near misses. This also allowed an appreciation of the behaviours and values required for a strong nuclear safety culture and enabled participants to develop their own list of principles for a strong nuclear safety culture.

Members of the management teams of each business unit delivered the workshops to demonstrate their ownership and commitment to the programme.

In addition to the main workshops further education of leaders was provided. These were used to provide them with additional information on the role of leaders in promoting and supporting the expected behaviours for a strong nuclear safety culture. It enabled them to demonstrate their commitment by their visible deeds and actions by what is commonly termed 'walking the talk'.

This ensured support and understanding to individuals and teams in implementing commitment made in the workshops to implementing improvements and enhancement of nuclear safety culture.

The following is an outline of the content covered during the nuclear safety culture workshops:

5.2.1 Introduction

- Communicating the expectations of the Companies senior executives: Via policy, words and actions. Attendees saw a specially created DVD of the senior team talking about the importance of nuclear safety culture to themselves, the organisation and all those who support the organisation
- Understanding that nuclear is unique and therefore the need for everyone to respect the nuclear core and the energy we are working with
- Understanding the importance of achieving the correct balance between introduction of additional preventative measures and the requirements for production and the use of support activities, processes and procedures designed to achieve this balance (Reason 1997)
- Understanding the behaviours and environmental condition that led to the Chernobyl accident. These were not bad people, but their focus was on

achieving their target whilst not recognising the need to stop and review warning signs.

5.2.2 Describing Culture and Nuclear Safety Culture

5.2.2.1 Culture

A simple model of culture was adopted to explain how culture may be internalised at a number of levels, aligned with:

- Behaviours
- Values
- Basic underlying assumptions.

The 'lily pond' model analogy was used to provide further explanation and support the internalisation of this concept:

- The behaviours being visible at the surface, such as lily pads on the surface of a pond
- The values below the surface, being less visible but linking the nutrients at the bottom of the pond through the water, roots and stems
- The basic underlying assumptions being the nutrients and soil, not visible at the surface but driving the long term health of the pond.

(Schein 2004, View from the Lighthouse 2004)

This model was also used to align with the IAEA model of safety culture (INSAG-4 1991) and a corporate policy level commitment, manager/leader commitment and individual level commitment to give further demonstration of the 'layers' within culture that must align to achieve organizational consistency and excellence in nuclear safety culture.

5.2.2.2. Nuclear Safety

An overview of nuclear safety was provided to remind everyone that nuclear technology has unique, potentially adverse effects: Maintained through positive control of reactivity, core cooling and containment and the responsibility given to nuclear operators by society.

5.2.2.3. Nuclear Safety Culture

The model of culture and nuclear safety were drawn together at the descriptive level from the above to gain an initial understanding of culture as how we normally do work around here which is driven by everyone's values and what we believe is normal. Due to the uniqueness and potential adverse effects it is then clear that everyone in the industry therefore needs to develop and maintain a work environment (the pond) that consistently nurtures and supports **nuclear safety as the overriding priority.**

In simple terms everyone is encouraged to align their behaviours at all times by doing the right thing when no one is looking, not only when being supervised, coached or observed.

5.2.3 Internalising Safety Culture by the use of Case Studies – Developing Principles for a Strong Nuclear Safety Culture

So much for theory. We needed to make this speak to all participants and this 'theory' was then brought to life by the use of a number of case studies to show how culture is multi dimensional and is something that can grow or deteriorate over a number of years to achieve a 'norm' that is well below what we would all expect.

This use of case studies enabled participants to identify values and underlying behaviours that led to the events or near misses and link these to what they observe and experience in their own work environment. By looking at the converse of these substandard values and behaviours they were then able to identify the main principles to achieve a strong nuclear safety culture and ensure a culture of continuous improvement.

These were then compared with and seen in all cases to align with those developed by INPO and WANO for a strong nuclear safety culture. It was good to see that when we stop and think, we generally all know what 'good looks like'.

5.2.4. Identifying Individual Impact on Nuclear Safety

The internalisation of nuclear safety culture and the principles for a strong nuclear safety culture were then used by individuals at the workshop to identify just exactly how their work could impact nuclear safety.

Yet, if the payroll clerk gets it wrong, what impact may this have on those performing front line safety work? To date, over 3000 staff and long term contactors from all of BE's activities have attended these workshops and every one has identified a direct or indirect link between their work and nuclear safety.

5.2.5. Identifying and Removing Barriers to a Strong Nuclear Safety Culture

Once individuals were able to understand their own impact on nuclear safety and the principles for a strong nuclear safety culture they were then able to identify two levels of improvement that could be made to progress and improve: How could leaders/managers remove organisational barriers? How could they as individuals commit to improve in line with the principles for a strong nuclear safety culture?

5.2.6. Support for Changing Behaviours and Improving Culture

The workshops were rounded off by sharing individual commitments to improve, but also by outlining the management and leader commitments to supporting the changes already agreed for the Encourage Phase. As we already explained, seeing managers and leaders 'walking the talk' gives a good visible demonstration of their commitment to which the organisation should align.

5.3 The Encourage Phase

The workshops were designed to enhance awareness and understanding of the concepts of nuclear safety culture and how these concepts apply to the individuals day to day work activities. It was recognised as part of the development of the enhancement plan that what actually happens day to day would drive the culture. Hence, a parallel Encourage Phase was developed to take forward the learning from the Educate phase to help change behaviours and the overall culture.

A number of enabling activities for this phase were developed from which each business unit developed their own Encourage Phase Plan.

Key to this phase was the behaviours of the leaders throughout the organisation: the visible actions and decisions by the management/leaders. These set the tone for what the real expectations were believed to be by the organization. Hence it was essential that the management, down through all levels of leaders and supervisors, were aligned on these expectations. This aligns with the Pond Model used to understand culture outlined in 5.2.2.1 above.

Visibility of leaders starts with them 'walking the talk'. This needs to be enhanced at the start of the phase and less so once they are satisfied that there is good alignment. This is achieved by spending time in the field observing, coaching, mentoring, correcting inappropriate behaviours and reinforcing the desired behaviours. Management cannot be seen to be tolerant or unaware of inappropriate behaviours and it is only by making themselves visible to the workforce, observing activities in the field that this can be achieved. These requirements align with the task observation and coaching activities expected of leaders and developed as part of the parallel Human Performance enhancement programme.

Another companion activity applicable to this phase of change was for management to have periodic small group meetings with the general workforce. These provided the opportunity to discuss what is working well across the organisation as well as discussing management expectations, difficulties encountered in meeting those expectations and possible solutions. This is described as a 'Compliments and Concerns programme'.

Leaders and supervisors also need to take time to talk to their team members, collectively and individually to understand what activities or actions they would like to pursue or what changes they would like to make as a result of the learning from the workshops. This enables them to support their team in making changes until these become embedded.

Following on from these management and leader activities, there is a need to ensure that their actions and decisions demonstrate and reinforce the desired culture. This leads to the requirement for communication to the workforce on the bases for safety significant decisions to ensure that they are not perceived to be contrary to the stated values, principles and expectations.

5.4 The Embed Phase

The final and ongoing phase of the enhancement plan is the Embed phase. The objective of this phase is to achieve Nuclear Safety excellence as a part of the way we do business. Within this phase it is recognised that there will be a requirement for continuous improvement, a culture in which organisational learning is part of the way we do our work.

The main activities for this phase will bė:

1. Ongoing mechanisms for assessing and benchmarking nuclear safety culture and determining further local or generic improvement activities
2. Continued refinement of processes and procedures to ensure they explicitly highlight and address nuclear safety
3. Mechanisms to align the behaviours of new people and new leaders to the changed culture of the organisation
4. Ongoing removal of barriers to a strong nuclear safety culture
5. Communicating nuclear safety successes.

6 Conclusions

The creation of an appropriate safety culture within high impact failure organisations is paramount in ensuring a safe, productive and profitable organisation. British Energy's approach to enhancing safety culture by enabling all staff and major contractors to learn from significant worldwide events caused by shortfalls in safety culture is outlined as a key element of aligning the whole organisation on the need for continuous improvement in safety culture as an ongoing Performance Improvement activity.

Reference List

International Atomic Energy Agency (IAEA) Nuclear Safety Advisory Group Reports:
INSAG-4 Safety Culture (1991)
INSAG-15 Key Practical Issues in Strengthening Safety Culture (2002).

Institute of Nuclear Power Operators (INPO), World Association of Nuclear Power Operators (WANO) Performance Objectives and Criteria (1999)

INPO/WANO and Principles for a Strong Nuclear Safety Culture (2004, 2006)

Reason J (1997). Managing the Risks of Organisational Accidents. Ashgate .

Schein, E.H. (2004). Organizational culture and leadership, 3rd edition. Jossey-Bass, San Francisco

The Pond Model, reprinted from View From The Lighthouse (2005) CoastWise Consulting, Inc. www.coastwiseconsulting.com

Comparing and Contrasting some of the Approaches in UK and USA Safety Assessment Processes.

Richard Maguire, B.Eng., M.Sc., C.Eng., MIMechE MSaRS
SE Validation Limited
rlm@sevalidation.com

Abstract

Humanity is thinking very hard about how accidents initiate, develop and propagate into disasters, such that they can be prevented or interrupted before they have opportunity to cause harm, injury or loss. Many industries and countries have authorities and inspector organisations that research and police hazardous areas of work and judge safety performance. Evidence is often called for in demonstration of safety performance and this has many beneficial features, from identifying areas for improvement to providing defence evidence in legal cases. The focus of the approaches to compile the evidence is always concerned with understanding the safety status of a system with the familiar goal of avoiding future accidents. However, there are differences in these approaches across the many industries and nations of the world, and interestingly, differences in national and industrial fatal accident statistics.

This paper seeks to review some of the approaches to safety assessment and evidence collection in just two nations – the USA and the UK. Further, this paper seeks to evaluate whether any differences in approach could be considered as contributory causes to the differences in fatal accident rates between these nations.

Keywords
Safety Assessment Comparison, Safety Cases, Safety Reports

1 Introduction

There are multiple requirements throughout the world for risk and safety analysis in a wide variety of industries. It is unfortunate that different phrases and terms are used to identify them. The main interest of comparing and contrasting the definitions and some of the approaches used is to demonstrate that even though there are hugely significant overlaps in processes and goals, there are still significant differences in national fatal accident statistics. As a by-product of this paper, the reader will also gain an insight into some of the key words, descriptions and processes used in safety assessment in the USA and the UK. For international co-operation and interoperability, this insight can be an essential pre-requisite.

The rest of the paper will initially review the stated definitions of key safety terms in each country, compare the published figures for fatalities in various industries and contrast some of the approaches used to assess and record safety performance. The respective use of monetary values as a measure of accident impact and the methods for safety planning in both nations will be examined, followed by a discussion on the concepts of safety reporting. The paper will also seek to suggest reasons for the differences in national fatality statistics based on the differences in the approaches reviewed.

2 Common Definitions or not?

Recent research for the Health and Safety Executive (Williamson & Weyman 2005) and carried out by others (Maguire & Brain 2006), has reviewed the public perception of risk and the meanings of key words in the safety domain. However, these research studies were not focussed on performing a comparison of the meanings of words defined for use in comparable industries between the USA and the UK. A brief review of the defence industry definitions in both countries will serve to demonstrate the differences of the common language.

In the USA military the System Safety Approach is approved for use by all departments and agencies within the USA Department of Defense (DoD 2000). Its objectives are to protect private and public personnel from accidental death, injury or occupational illness; also to protect public property, equipment, weapon systems, material and facilities from accidental destruction or damage while executing missions of national defence. Its definition of safety is as follows;

> "Safety: Freedom from those conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment." (DoD 2000)

In the UK military the exact term 'safety' is not defined in the latest interim version of Defence Standard 00:56 on Safety Management Requirements for Defence Systems, it only defines 'safe'. However, the earlier issue 2 of the standard does have a definition as follows;

> "Safety: The expectation that a system does not, under defined conditions, lead to a state in which human life is endangered. (The scope of "safety" may be expanded by adding to this definition in the Programme Safety Plan.)" (MoD 1996)

In more recent years the definitions used have indeed been expanded in the UK to allow for the inclusion of property, equipment and even public and political image. However, additional comparisons are difficult, as in the USA there is reluctance to derive and publish definitions that can have legal connotations. For example, former Inspector General of the United States Department of Transportation, Mary Schiavo, is attributed (Motley Rice 2006) as saying that;

> "The Federal Aviation Administration (FAA) is the government agency responsible for ensuring our country's safe air travel. Most people do not realize that government agencies have failed to agree on a definition of safety or uniform guidelines."

The use of the word 'accident' is also an area of contrast – again perhaps due to the legal connotations that it could have in the USA. The UK Health and Safety Executive has the following definition for a 'Major Accident' from its COMAH guidance;

> "Major Accident: An occurrence (including in particular, a major emission, fire or explosion) resulting from uncontrolled developments in the course of the operation of any establishment and leading to serious danger to human health or the environment, immediate or delayed, inside or outside the establishment, and involving one or more dangerous substances." (HMSO 1999)

In the USA the term 'mishap' is preferred, this definition is again from the USA Department of Defense;

> "Mishap: An unplanned event or series of events resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment." (DoD 2000)

Aside from the obvious differences in phrase choice, the underlying content and meaning of these words does have a similarity. Both accident and mishap specifically include aspects of human health and damage to the environment. The definitions of safety both relate to a freedom from human harm, with the potential option to consider additional aspects.

A further area to highlight is that of the health and safety plan. In the USA and UK, this plan, known as a HASP, has a dedicated use in specific, but different industries. However, there are striking similarities in approach and content.

In the USA, the Health and Safety Plan or HASP specifically addresses hazardous waste. This includes decontamination and clean up of a hazardous waste site and investigating the potential presence of hazardous substances. The key elements of a HASP, whilst having the specific objectives described above, would be useful in many other safety related planning programmes. They are as follows:

- Site characterisation and system description
- Identifying the safety and health risks
- Specifying requirements for personal protective equipment
- Specifying requirements for health surveillance
- Site control, monitoring and decontamination
- Production of an emergency response plan
- Procedures for confined entry and spill containment (DoE 1994).

The construction industry is the focus for the UK HASP, it is the subject of The Construction (Design and Management) Regulations. As part of tendering for a construction contract a Health and Safety Plan must be submitted. The pre-tender plan must be developed for the construction phase to include:

- A full description of the project
- Arrangements for managing the project
- Arrangements for monitoring compliance with health and safety requirements
- The identified risks to health and safety
- Arrangements for welfare of people associated with the project (HMSO 1994)

Upon inspection there is a good comparison between the international uses of the HASP. Even though the plans are used for different industries, the objectives and contents are remarkably similar. Both plans require a description of the system, and identification of the risks to health and safety. Both also specifically call for attention to the welfare or health of the people associated with the respective systems under the plans. Site control in the USA plan can be seen as analogous to the UK management of the project, and a project-monitoring task also appears in both. There is a difference in fidelity between the requirements for compliance with regulations. The UK plan speaks generically of monitoring compliance with (national) health and safety requirements, without being specific about which requirements are meant. The USA plan cites some specific requirements for protective equipment and an emergency response plan, which appear to be related to the local site and project under consideration. However, these differences are considered as minor when compared to the overall similarity of the plans.

The use of the HASP does not necessarily need to be industry specific. The approaches set down would be equally applicable to any industry, any project, and any system (Maguire 2006). The other interesting point to note is that both these plan definitions were published in the same year, 1994.

3 Comparison of Industry Data

Consider the following data from the USA Bureau of Labor Statistics, the UK Health and Safety Commission, the UK Defence Analytical Services Agency and the USA Army military fatality statistics, as shown in tables 1 to 3 below. These tables quote actual recorded data for workplace fatalities in a number of industry types over a 12-month period;

USA Industry	Number of fatalities (Y2004)	Workplace fatality rate per 100,000 workers (Y2004)
Manufacturing	459	2.8
Construction	1224	11.9
Mining	152	28.3
Government	526	2.5
Agriculture	659	30.1
Transport & Warehouse	829	17.8
Retail Trade	372	2.3
Leisure	245	2.1
Finance	115	1.2
	ALL INDUSTRY	4.1 (4.1 x 10-5)

Table 1. USA Industry fatality statistics (USA DoL 2005)

UK Industry	Number of fatalities (FY2004/05)	Workplace fatality rate per 100,000 workers (FY2004/05)
Manufacturing	41	1.2
Construction	72	3.5
Extraction and utility	2	1.1
Service industries	63	0.3
Agriculture	42	10.4
	ALL INDUSTRY	0.6 (6 x 10-6)

Table 2. UK Industry fatality statistics (HSC 2005)

Statistic	Fatalities	Rate
UK Accidental and violent deaths in the Army (including road traffic accidents and suicides). 2004.	75	65 per 100,000 strength (6.5 x 10-4)
USA Accidental Army fatalities (Ground based and including reservists). Financial year 2004 to 2005.	272	40 per 100,000 soldiers (4.0 x 10-4)

Table 3. UK and USA Military fatality statistics (DASA 2005, USA Army 2005)

It should also be noted that each country does not necessarily define the industry sectors and military personnel in the same way, and the time periods whilst being 12 months, do not necessarily cover the same 12 months. However, the definitions are considered close enough to be useful for general comparison – particularly the all-industry figure.

Whilst the military statistics show a similarity at values of the order 10-4, an initial point to note is that according to the whole set of figures, it does appear to be much less safe to be employed in the military.

The difference in the all-industry fatality rate is quite shocking. For every UK industry fatality, there are nearly 7 USA industry fatalities. Comparing the data entries, it is interesting to observe that in both countries the agricultural fatality rates are the highest, and that service industries (leisure, retail and finance in the USA figures) have the lowest fatality rates. Where the definitions of industry types are close enough, a direct comparison indicates that the USA has fatality rates at least twice the UK rates (e.g. manufacturing : USA 2.8, UK 1.2 per 100,000). In the case of mining and extraction, the USA fatality rate is 25 times the UK rate.

There must be significant reasons behind the contrasting nature of some of the specific industry values and the all-industry values that go beyond the contrariety in definition of industry sectors and the calendar for data collection. The next three sections of this paper consider a variety of aspects that may be seen as contributory factors.

4 Value of a Prevented Fatality (VPF)

In the USA, the National Safety Council makes an estimate of the average costs of fatal and non-fatal injuries from unintentional sources as a way to illustrate the impact of these events on the USA national economy. The costs are a measure of the money spent and the income not received due to accidents, injuries and fatalities. It is regarded as a further way to understand the importance of preventative measures. The determined values can be used to estimate the financial impact on a State or local community, and can be used for cost-benefit analysis.

> "Costs of Motor Vehicle Injuries
>
> These costs are calculated from an assessment of wage and productivity losses including the total of wages and benefits; medical expenses including doctor fees, hospital fees, costs of medicines and emergency medical vehicle services; administrative costs including public and private insurance costs – i.e. the costs of doing business; property damage including vehicle and equipment loss and repair. Average economic cost per death, injury or crash, 2004:

Death	$1,130,000
Non-fatal disabling injury	$ 49,700
Property damage / other non-disabling injury	$ 7,400

> In addition to the direct economic cost components relating to a motor vehicle accident, a 'comprehensive cost' is also determined. This includes a measure of the value of lost quality of life, this has been obtained through empirical studies of what people actually pay to reduce their safety and health risks. These cost values do not represent real income or expenses incurred, therefore they can only be used for cost-benefit analysis calculations.

Death	$3,760,000
Incapacitating injury	$ 188,000
Non-incapacitating injury	$ 48,200
Possible injury	$ 22,900
No injury	$ 2,100" (NSC 2005)

In the UK, the government Department for Transport (DfT) has carried out a very similar type of calculation for assessing the economic viability of road improvement schemes. The calculated values are used to make judgements about total value to the community of the benefits of preventing road traffic accidents. The UK cost includes direct factors such as lost economic output, medical and emergency service costs. It also includes a significant contribution from 'human costs', which reflects the USA concept of 'lost quality of life' when you die.

The 2004 figures for the construct of the DfT value for the prevention of a fatal accident is as shown in Table 4 below.

Element of Cost	Value (£)
Lost Output	522,639
Medical Costs	5,469
Human Costs	1,033,783
Police Costs	1,607
Insurance and Admin	254
Property Damage	9,465
Total	1,573,217
	($ 2,831,800)

Table 4. Construct of the DfT VPF (DfT 2005)

The VPF has subsequently been adopted widely throughout the safety domains of UK industry, but sometimes with the additional use of a proportion factor. The proportion factor is usually in the form of an integer multiplier and is based on a societal anticipation that industry should pay more to prevent a fatality from a source that is publicly perceived to have a particular horror or dread factor associated with it, for example radiation poisoning. The Health and Safety Executive has published the following comments and guidance on the use of the DfT value, particularly in risk ALARP cost-benefit analyses for the control of major accident hazards (COMAH).

> "Le Guen, Hallett and Golob produced a paper in 2000 [Le Guen, Hallett & Golob, 2000] on the "Value of preventing a Fatality" which was circulated to HSE Board members and presented to the Risk Assessment Liaison Group (RALG) (RALG/Sep00/03). That paper discussed the ratio of the cost of preventing a fatality (CPF) to the value of preventing a fatality (VPF). The starting point for VPF was taken to be the DETR [now known as DfT] figure of approx. £1m [now £1.5m] used in new road schemes. Other values of VPF were then described [with a proportion factor introduced]. These were 2 x DETR for deaths from cancer and 3 x DETR for some aspects of railway safety." [HSE 2004]

Whilst the immediate composition between the UK and USA values are not identical, they are numerically comparable at the $3million to $4million mark, based on an exchange rate of around $1.8 to £1(typical of the period 2004 to 2005). The main contrast is in the UK's use of the proportion factors where there is a public dread component. As these are cited as being integer numbers, they do have a dramatic multiplication effect on the final total value.

The sets of calculations from the USA and the UK for a value of a prevented fatality both include largely similar cost factors, particularly including a human allowance for lost quality of life. Rather unsurprisingly they arrive at broadly similar values. In light of the similarities it appears to be the case that the value assigned to the prevention of a fatality is not a significant driver for the difference in fatality statistics.

5 Safety Planning

From the UK Ministry of Defence the following advice is given for a project safety management plan. Whilst this is an older text, it may be taken as a useful generic example of the plan concept. Where the text is perhaps topic focussed, I have provided a more generic interpretation (MoD 1996).

> "Structure. The following structure [and content description] may be adopted as a basis for the safety programme plan:
>
> Part 1 Introduction. This part should describe the system of interest, the project scope and objectives and a brief overview of the way safety will have an effect on the project.
>
> Part 2 Safety requirements. This part should list out all the safety requirements for the system of interest. These requirements will come from legislation, standards and codes of practice. The main purpose of this section is to provide a reference for all the project staff and to act as a record of all the requirements that are intended to be satisfied. This section should also record any interpretation of the requirements or any tailoring [selective adoption or rejection of specific requirements] that has occurred.
>
> Part 3 Management and Control. This part should contain a description of the 'who', 'when' and 'how' parts of implementing the safety plan. It should specifically include the timing of various assessments and reviews; the human resource structure for the safety programme – including the identification of the main safety personnel and their training requirements; how any sub-contractors are going to be managed; and how records are going to be kept specifically including the records of hazards and safety decisions taken.
>
> Part 4 Analysis and Assessment. This part should describe the safety analyses and assessments that are going to be conducted on the system. It needs to explain what safety information is going to be obtained from each particular analysis, and how that will fit into the whole system safety programme."

From the USA Occupational Health and Safety Administration, the following advice is given in the Hazardous Waste Operations and Emergency Response Standard (HAZWOPER). Again, where the text has become specific, a more generic interpretation is given (OSHA 2006).

> Employers shall develop and implement a written safety and health plan for their employees involved in hazardous waste operations. The program shall be designed to identify, evaluate and control safety and health hazards, and provide for emergency response during hazardous waste operations. The written program shall incorporate the following.
>
> (A) An organizational structure. This part of the program shall establish the specific chain of command and identify the responsibilities of supervisors and employees.
>
> (B) A comprehensive workplan. This part of the program shall address the tasks and objectives of the system's operations including the processes, logistics and resources required. This shall also include the anticipated clean-up activities as well as normal operations.

(C) A site specific safety and health plan. This part of the program shall address the safety and health hazards specific to each phase of system operation. It shall consist of a hazard and risk analysis for each task; details of personal protective equipment to be used; frequency of contamination monitoring; any site control measures; details of the emergency response plan; and accident control measures.

(D) The safety and health training program. This part of the program shall record the required training, which should thoroughly cover the following areas: Identification of the personnel responsible for safety and health; educating the employees about all hazards within the site or system; training in the use of the protective equipment; specifying the work practices by which an employee can minimize the risks from hazards; training in the safe use of equipment and controls; and having a medical surveillance program in place.

[NOTE: This particular standard specifies that general site workers should receive a minimum of 40 hours of off-site instruction and three days field experience on-site, under the direct supervision of a trained and experienced supervisor. In the author's mind, it remains debatable that training time alone is a reasonable factor in deciding on personnel's suitability for safety work.]

(E) The medical surveillance program. This part of the program shall specify the coverage [personnel to be included], nature and frequency of any medical surveillance program. This should relate to the nature and type of hazard present on the site or within the system e.g. chemical poisons, high noise levels and rotating equipment. The scope and role of the medical assessor should also be confirmed.

(F) The employer's standard operating procedures for safety and health. This part of the program shall address the engineering controls, work practices and protective equipment procedures for the hazards on the site or within the system. Specific attention should be given to selection of protective equipment; manual handling (especially drums and containers); transportation of hazardous materials; confined space entry; and decontamination procedures.

In reviewing these two approaches, a number of key aspects are notably common within them. The specification of a chain of responsibility in both is particularly welcome. The safety, risk and hazard analysis sections are equally comparable, as are the descriptions of the process, site or system of interest. The acknowledgement of safety training in both is excellent, as is the specification of a medical surveillance program in the USA plan. A more generic step noted in both approaches is the health monitoring of the actual system of interest – the safety review process and the description of how this is to be conducted.

The main difference between the two approaches is the way requirements are handled. In the UK safety programme plan, part 2 specifically focuses attention onto safety requirements from legislation, codes of practice and standards. In the US plan, part F focuses on standard procedural requirements from the source of the employer's standards, not a universal national standard. This could be a significant difference between USA and UK safety planning culture – a similar difference was noted in the review of the HASP. However as a whole, there are far more similarities than differences.

In light of this review, it does appear that in the approach to planning safety aspect of engineering, there are not enough differences to explain the variance in national fatality rates.

6 Safety Reporting

There are a multitude of recommended layouts for a safety report depending on where you are and what industry you are working in. In the UK, one approved methodology utilises the safety case report. There are many offerings on the content of a safety case report (See Maguire 2006), but typical is that from the MoD as follows (MoD 2004);

- Executive summary. This should enable the Duty Holder to provide assurance to stakeholders that he/she is content with the progression of work and that safety requirements have been, or will be met.
- Summary of system description. A brief description should be given noting that a full description is contained within the safety case [as a document suite].
- Assumptions. The assumptions that underpin the scope of the safety case, or the safety requirements, arguments or evidence should be stated.
- Progress against the programme. An indication of the current status relative to expectations within the programme, and progress on safety management since the previous safety report.
- Meeting safety requirements. This section should include a description of the principal, agreed safety requirements (e.g. ALARP), and a summary of the argument and evidence that demonstrates how the safety requirements have been, or will be met. A statement about the contemporary residual risk should be made.
- Emergency/Contingency arrangements. A statement confirming that appropriate arrangements have been or will be put in place and identification of any areas where such arrangements are likely to be inadequate.
- Operational information. This section should contain the output from the safety case that is relevant to the management of operational safety, including, the main risk areas and any limitations of use or operational capability.
- Independent safety auditor's report. Where an ISA is engaged, they should prepare a formal report for inclusion in the safety case report.
- Conclusions and recommendations. This should include an overall assessment of the safety of the system and any recommendations to enable any issues to be resolved.
- References. A list of key reference documents should be provided [including key test and process evidence, hazard logs, software and human integration reports, and any other evidence being used to support the safety case].

The USA doesn't explicitly use the 'safety case' phrase, however the Occupational Safety and Health administration does give advice on the completion of a 'safety and health program report' (OSHA 1989). The main section headings and brief contents of this advice, which is still extant today, has been used to develop the following report section list;

Part 1: Management Leadership and Employee Involvement

- Worksite safety policy
- Current safety goals and objectives
- Task and system description

- Orientation outline for staff, visitors and contractors
- Evaluation of safety and health responsibilities
- Budget showing money allocated to safety and health
- Evidence of employee involvement.

Part 2: Worksite Analysis

- Results of baseline site hazard survey with notation for hazard correction
- Employee reports of hazards
- Historical mishap investigation reports
- Trend analysis results
- Procedures for change analysis, which include hazard considerations.

Part 3: Hazard Prevention and Control

- Maintenance records
- Preventative maintenance procedures
- Site safety and health rules
- Emergency drill procedures
- Health surveillance and monitoring procedures
- Reports, investigations and corrective actions taken for near misses
- Specific OSHA mandated procedures for specific hazards.

Part 4: Training

- Program of yearly training topics
- Employee training recording procedures and data.

The MoD and OSHA recommendations both have explicit sections on emergency procedures and have an acknowledgement of the idea that there is an on-going process in place – the MoD explicitly asks for progress against the programme, the OSHA asks for trend analysis and a current safety goal ('current' implying that this changes over time). The OSHA list stands out by specifically asking for a declaration of the budget allocated to safety and health. It is difficult to see this happening in the UK, although as part of ALARP, there is a cost-benefit aspect to the accompanying analysis, so perhaps that is not so far away.

There are also definite differences between them. The training and health surveillance aspects are made more explicit in the USA list, this is due to the combined scope of safety *and health* that is not present in the UK contents list. The MoD list explicitly asks for the definition of operational limitations and any safety issues that are still outstanding. The UK reporting procedure also allows (and some say, mandates) for the use of an independent advisor or auditor – the ISA. This is not present in the USA reporting description. An external review can add significant rigour to any process, particularly where the reviewer is a required signatory to the report in question.

Further, there appears to be an important difference in the foci of the safety reports. The USA safety report pays a great deal of attention to the health of the human in the system, "the employee" i.e. where any hazardous impact is felt and measured. The UK safety report does acknowledge this area, but does have a wider-based focus on the whole system or equipment of interest i.e. where the hazard impact initiates. The views of safety from both ends of an accident sequence are equally valid approaches, but the earlier the assessment starts, the more options for prevention and mitigation there are. I suggest that the wider view with additional assessment and prevention opportunities will capture and halt more accident sequences and could prevent more fatalities.

Both the UK and USA approaches have their merits and both have their shortcomings. However, perhaps the differences discussed above in context, rigour and focus of the safety reporting approaches, are indications of where the different fatality rates might start to originate?

7 Summary

There are many social, industrial and cultural aspects that contribute to any national statistic. In the safety domain, the national fatality statistics and a few of the possible domain-specific contributory approaches have been compared and contrasted. It is surprising, at least to this author, that the differences in the approaches and aspects are really quite small. Engineering judgement and experience might suggest that these approaches should contain significant differences – many engineers on both sides of the Atlantic will have horror stories of dealing with the other side. However, a slightly deeper look into the description of safety plans, fatality values and safety reporting has indicated that these processes and aspects are actually much more comparable than each country might think. Both sets of national approaches should be equally respected.

The reference to UK national (and therefore potentially more consistent) safety requirements in planning, and the rigour and systematic approach to reporting in the UK through a safety case, have been the main differences noted. These could be the significant factors that explain some of the differences in the fatality statistics. However, cultural, historical and doctrinal aspects have not been assessed – these should be areas of future work.

8 References

DASA 2005, "Deaths in the UK Regular Armed Forces 2004", Defence Analytical Services Agency.

DfT 2005, "Highways economic note No.1 : 2004", Department for Transport, London.

DoD 2000, "Standard Practice for System Safety" Military Standard 882D, United States Department of Defence, February 2000.

DoE 1994, "Hazard Baseline Documentation" DOE-EM-STD-5502-94, Section 5.5. United States Department of Energy, August 1994.

J. Le Guen, N. Hallett, and L. Golob, 2000, "Value of Preventing a Fatality" A paper by the Risk Assessment Policy Unit and Economic and Statistical Advisory

Unit, SASD. Internal paper for the HSE Risk Assessment Liaison Group.[Quoted in HSE 2004]

HMSO 1994, “The Construction (Design and Management) Regulations” Statutory Instrument 1994 No. 3140. Her Majesty’s Stationary Office, London.

HMSO 1999, “The Control of Major Accident Hazards (COMAH) Regulations” Statutory Instrument 1999 No. 743. Her Majesty’s Stationary Office, London.

HSC 2005, “Statistics of Fatal Injuries 2004/05”, The Health and Safety Commission, Bootle, UK.

HSE 2004, “Guidance on ‘as low as reasonably practicable’ (ALARP) Decisions in Control Of Major Accident Hazards (COMAH)”, The Health and Safety Executive, July 2004.

R Maguire, 2006, “Safety Cases and Safety Reports”, Ashgate Publishing, 2006.

R Maguire & C Brain 2006, “History and Perception of the Language Used in the Safety Domain”, The IET first international conference on System Safety, June 2006, London.

MoD 1996: “Safety Management Requirements for Defence Systems Part 1” Defence Standard 00:56, Issue 2. Ministry of Defence, December 1996.

MoD 2004: “Safety Management Requirements for Defence Systems Part 1” Interim Defence Standard 00:56, Issue 3. Ministry of Defence, December 2004.

Motley Rice, 2006: “Industry Accountability”, Company information available at:- http://motleyrice.com/transportation/aviationsafety/IndustryAccountability.asp

NSC 2005, “Estimating the costs of unintentional injuries, 2004”, National Safety Council, Itasca, USA.

OSHA 1989: “Safety and Health Program Management Guidelines 1926 Subpart C”, Occupational Safety and Health Administration, US Department of Labor, USA.

OSHA 2006, “Hazardous Waste Operation and Emergency Response (HAZWOPER)” Standard, 29 CFR 1910.120, OSHA April 2006.

USA Army 2005, Combat Readiness Centre, “Army Safety Statistics – Ground Accident Statistics” available at:- https://rmis.army.mil/stats/prc_fy_ground_stats

USA DoL [2005], “Census of Fatal Occupational Injuries (CFOI) – Current and Revised Data”, USA Department of Labor, Bureau of Labor Statistics, USA.

J Williamson & A Weyman 2005, “Review of the Public Perception of Risk, and Stakeholder Engagement HSL/2005/16”, Health and Safety Laboratory, 2005.

Trends in Safety Case Development

Safety Case Composition Using Contracts - Refinements based on Feedback from an Industrial Case Study

Jane Fenn and Richard Hawkins
BAE SYSTEMS, Brough, UK
Phil Williams
General Dynamics (United Kingdom) Ltd, Hastings, UK
(representing the Industrial Avionics Working Group)

Tim Kelly
University of York, York, UK

Abstract

Modular safety cases provide a means of organising large and/or complex safety cases into separate but interrelated component modules of argument and evidence. Safety case 'contracts' can be used to record the interdependencies that exist between safety case modules – e.g. to show how the claims of one module support the arguments of another. A number of techniques for structuring and describing modular safety cases using the Goal Structuring Notation were defined by Kelly in (Kelly 2001). The Industrial Avionics Working Group, (IAWG) has been using these techniques as part of a substantial industrial case study being funded by the UK Ministry of Defence. Based on this experience, and a number of issues encountered, modifications to the original approach have been defined. This paper presents some of these experiences of the IAWG in using 'modular' GSN – in particular, those relating to capturing and recording safety case contracts – and proposes an enhanced approach.

1 Introduction

The Industrial Avionics Working Group, (IAWG), which was formed in 1979, is an industrial consortium of companies working in the aerospace sector, namely, BAE SYSTEMS, General Dynamics (United Kingdom) Ltd, Westland Helicopters, Smiths Aerospace and SELEX S&AS. During 2006, the Ministry of Defence has funded a programme of research, building on a feasibility study carried out by IAWG, and developing a modular safety argument for an aircraft system. This has entailed the use of the modular Goal Structuring Notation (GSN) extensions defined by Kelly (2001). This activity has highlighted some issues for which IAWG, in conjunction with Kelly, have proposed some modification and enhancements to the definition of the modular GSN extensions, with accompanying guidance on implementation issues.

2 Modular GSN Definition

GSN has been widely adopted by safety-critical industries for the presentation of safety arguments within safety cases. However, to date GSN has largely been used for arguments that can be defined 'stand-alone' as a single artefact rather than as a series of modularised interconnected arguments. In order to make the GSN support the concepts of modular safety case construction it has been necessary to make a number of extensions to the core notation.

The first extension to GSN is an explicit representation of modules themselves. This is required, for example, in order to be able to represent a module as providing the solution for a goal. For this purpose, the package notation from the Unified Modelling Language (UML) standard has been adopted. The GSN symbol for a safety case module is shown in Figure 1.

In presenting a modularised argument it is necessary to be able to refer to goals (claims) defined within other modules. Figure 1 introduces an element to the GSN for this purpose – the "Away Goal". An away goal is a goal that is not defined (and supported) within the module where it is presented but is instead defined (and supported) in another module. The Module Identifier (shown at the bottom of the away goal next to the module symbol) should show the unique reference to the module where the goal can be found.

Away goals can be used to provide *support* for the argument within a module, e.g. supporting a goal or supporting an argument strategy. Away goals can also be used to provide contextual backing for goals, strategies and solutions.

Representation of away goals and modules within a safety argument is illustrated within Figure 1. The annotation of the top goal within this figure "SysAccSafe" with a module icon in the top right corner of the goal box denotes that this is a 'public' goal that would be visible as part of the published interface for the entire argument shown in 1 as one of the "objectives addressed".

The use of some of these notational extensions by the IAWG in developing the modular safety argument has highlighted issues which are discussed in Section 3.

The strategy presented within Figure 1 to address the top goal "SysAccSafe" is to argue the safety of each individual safety-related function in turn, as shown in the decomposed goals "FnASafe", "FnBSafe" and "FnCSafe". Underlying the viability of this strategy is the assumed claim that all the system functions are independent. However, this argument is not expanded within this "module" of argument. Instead, the strategy makes reference to this claim being addressed within another module called "IndependenceArg" – as shown at the bottom of the away goal symbol. The claim "FnASafe" is similarly not expanded within this module of argument. Instead, the structure shows the goal being supported by another argument module called "FnAArgument", indicated by the 'module reference' symbol. The "FnBSafe" claim is similarly shown to be supported by means of an Away Goal reference to the "FnBArgument" module. The final claim, "FnCSafe", remains undeveloped (and therefore requiring support) – as denoted by the diamond attached to the bottom of the goal.

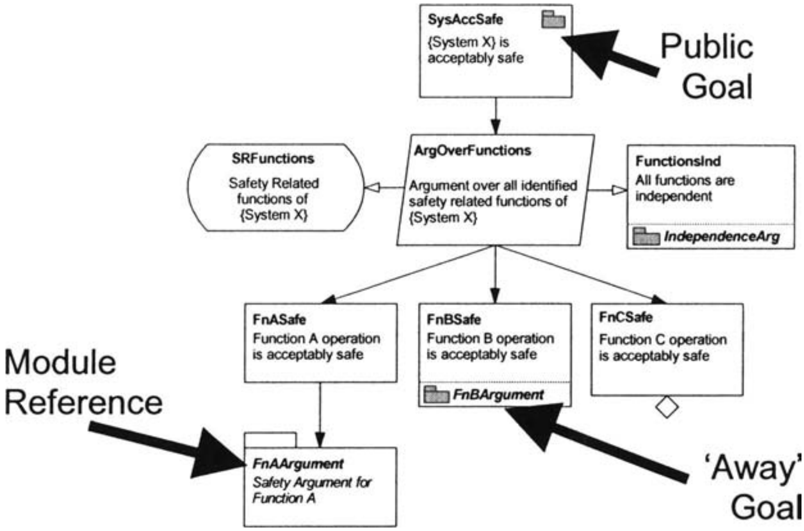

SysAccSafe
{System X} is acceptably safe
Public Goal
SRFunctions
Safety Related functions of {System X}
ArgOverFunctions
Argument over all identified safety related functions of {System X}
FunctionsInd
All functions are independent
IndependenceArg
FnASafe
Function A operation is acceptably safe
FnBSafe
Function B operation is acceptably safe
FnBArgument
FnCSafe
Function C operation is acceptably safe
Module Reference
FnAArgument
Safety Argument for Function A
'Away' Goal

Figure 1 GSN Extension

In the same way that it can be useful to represent the aggregated dependencies between software modules in order to gain an appreciation of how modules interrelate 'in-the-large' (e.g. as described in the 'Module View' of Software Architecture proposed by Hofmeister et al. in (Hofmeister et al. 1999) it can also be useful to express a module view between safety case modules.

If the argument presented within Figure 1 was packaged as the "TopLevelArg" Module, Figure 2 represents the module view that can be used to summarise the dependencies that exist between modules. Because the "FnAArgument" and "FnBArgument" modules are used to support claims within the "TopLevelArg" module a supporting role is communicated. Because the "IndependenceArg" module supports a claim assumed as conte≻. to the arguments presented in "TopLevelArg" a contextual link between thesε modules is shown.

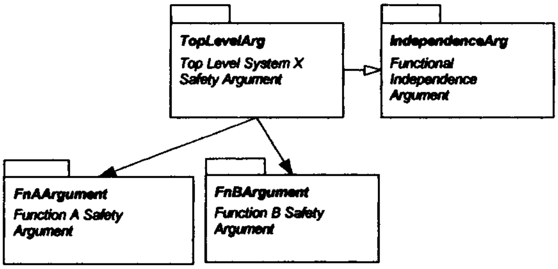

TopLevelArg
Top Level System X Safety Argument
IndependenceArg
Functional Independence Argument
FnAArgument
Function A Safety Argument
FnBArgument
Function B Safety Argument

Figure 2 – Example Safety Argument Module View

In a safety case module view, such as that illustrated in Figure 2, it is important to recognise that the presence of SolvedBy relationship between the module TopLevelArg and FnAArgument implies that there exists at least one goal within TopLevelArg that is supported by one or more arguments within FnAArgument. Similarly, the existence of an InContextOf relationship between TopLevelArg and IndependenceArg implies that there exists at least one contextual reference within TopLevelArg to one or more elements of the argument within IndependenceArg.

Alongside the extensions to the graphical notation of GSN, the following supporting documentation is required:

Interface declaration for each safety case module – the external visible properties of any safety case module must be recorded – e.g. the goals it supports, the evidence (solutions) it presents, the cross-references ('Away Goal' references) made to / dependencies upon other modules of argument. Figure 3 depicts the items to be defined on the boundary of a safety case module expressed using the GSN.

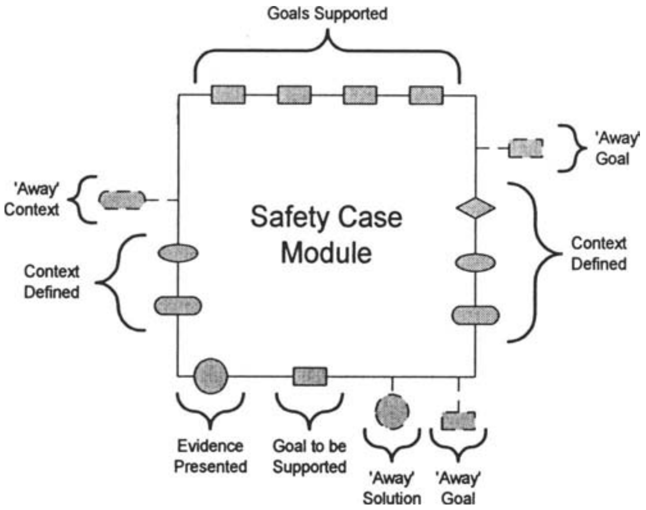


Figure 3 – The Published Interface of a GSN Safety

The overall safety case is composed from the safety case modules by linking elements in different modules in a 'safety case contract', such as goals requiring support from one safety case module are solved by public goals in a second safety case module. Kelly proposes a table is used to record which elements are provided or resolved by the contract and context which is consistent between the modules. It is use of these tables to record safety case module contracts that IAWG found to be difficult in practice and so have proposed an alternative strategy, as described in section 4.

3 Issues of Using Modular GSN Notation

Initial concerns arose when using the 'Module Reference' notation within a safety case module. An example below represents where the computing architecture provides functions that prevent applications running on it from communicating other than by pre-defined mechanisms. The safety case module discussing the need to prevent unintended communication between applications doesn't need to know how the architecture provides that capability, but does need to know that the architecture safety case module will provide that argument, so the following GSN fragment represents this situation using the 'module reference' symbol.

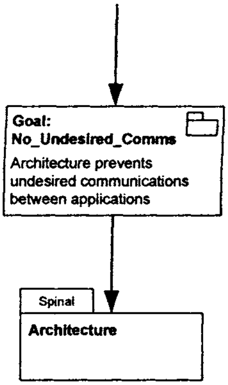

Goal:
No_Undesired_Comms
Architecture prevents undesired communications between applications
Spinal
Architecture

Figure 4 - Safety Argument Fragment in the Application Safety Case Module

This 'module reference' symbol had been used as a way of indicating that the goal would be solved using some goal (or goals) in the named module, (with the link explicitly defined in a separate safety case contract). This is distinct from an 'away goal', which references a specific goal in another module, such that it is essentially *hard-wired,* thus not requiring a separate safety case contract. In the example in Figure 4, the claim 'Goal: No_Undesired_Comms' is to be solved using a goal (or goals) contained within the Architecture module. The specific goal (or goals) from the Architecture safety case module that are to be used would be specified in the safety case contract.

Where the module containing support for a goal is not known in advance, Kelly proposes the use of the 'undeveloped' annotation. Using this approach in the example in figure 4, the goal requiring support, 'Goal: No_Undesired_Comms' would simply have been left as an undeveloped goal and the module reference element would not have been used.

Neither of the two approaches to representing a goal requiring support discussed above was found to be ideal when applied to the case study. Below, the issues and challenges are highlighted.

3.1 Undeveloped goal approach

The use of the 'undeveloped goal' notation where a goal is supported by argument in another module raised some concerns:

- It is not possible to distinguish between goals requiring support from other modules (i.e. those requiring a safety case contract), and those that require further development, i.e. the method of development is unclear
- Even once the contract is in place, there is no way of identifying where the contract is made, or the modules that are linked, as the goal remains represented as an undeveloped goal, i.e the GSN provides no visibility of the contractual inter-module links

These drawbacks are balanced by providing a representation which does not 'hard-wire' the argument into any safety case architecture constraints, i.e. changing the supporting argument module does not require the calling safety case module to change, only the safety case contract linking the two modules needs to change.

3.2 Module reference approach

The module reference approach provides greater visibility that a goal is supported by another safety case module, but the argument becomes 'hard-wired' to an extent. The argument module developer is forced to identify up-front the module that is going to provide support for the goal. This doesn't permit the desired flexibility to allow changes to the way a goal is solved using other modules; it also puts the requirement upon the developer of the module to identify the way in which the goal will be discharged by other modules.

3.3 Summary of GSN Notational Issues

Clearly a 'trade-off' has been identified between the visibility of the definition of links between safety case modules and maximising the reusability and modifiability of modules by minimising the 'hard-wiring' between safety case modules.

4 Issues of Using Safety Case Contract Tables

Kelly (2001) describes the use of Safety Case Contracts as a matching between 'goals requiring support' (expressed as un ،eveloped goals or module references) and 'goals providing support' (expressed a، public goals) across safety case module boundaries. Defining the goals that are public (and hence available to provide support to other modules) and those that are private (and not available to provide support) has raised further issues, discussed below.

4.1 Public and Private Goals

Once the goals requiring support from other modules have been identified, it is necessary to record the goals defined in other modules that are to be used to provide this support, by means of a safety case contract. These goals providing support are often referred to as the *public goals* of the safety case module. Kelly (2001) notes that the interface should not necessarily contain all of the goals supported by the module, owing to the fact that some will be considered internal detail whilst others will not.

It is possible for any goal to be declared public, but this may not necessarily be desirable, particularly if modules are being developed independently, in which case it would need to be negotiated explicitly as to which goals are required to be public. It is desirable that the number of public goals should be as restricted as possible. Using only the minimum necessary public goals eases assessment of the impact of changes on other modules. There is however a trade-off between easier assessment of change (which requires a small number of public goals) and reusability (which is easier when more goals have been declared public).

In order to develop a modular safety case, the argument integrator may need visibility of private goals in modules, and then request the goal 'owner' to make the required goals public in that module. It should not be possible for anyone other than the 'owner' of the module to change the public/private status of a goal. It may be useful to try to enforce this through tool support.

If goals which have been declared public are not used to discharge a goal requiring support from another module in a given safety case architecture configuration, then it should be made clear that this is the case, as they are not then of concern when considering the impact of changes on other modules. Therefore it may be necessary to indicate in some way which public goals are unused for a particular safety case, such that it is clear that whilst the goals are 'visible' to other modules, they are not required.

4.2 Capturing Safety Case Contracts

Whenever a successful match can be made between goals requiring support in one module, and goals provided in another module, a contract is made to capture the agreed relationship between the modules. Kelly (2001) proposes a table to be used for capturing the contractual relationship as shown in Table 1.

<table>
<tr><th colspan="6">Safety Case Module Contract</th></tr>
<tr><th colspan="6">Participant Modules</th></tr>
<tr><td colspan="6">(e.g. Module A, Module B and Module C)</td></tr>
<tr><th colspan="6">Goals Matched Between Participant Modules</th></tr>
<tr><td><i>Goal</i></td><td colspan="2"><i>Required by</i></td><td colspan="2"><i>Addressed by</i></td><td><i>Goal</i></td></tr>
<tr><td>(e.g. Goal G1)</td><td colspan="2">(e.g. Module A)</td><td colspan="2">(e.g. Module B)</td><td>(e.g. Goal G2)</td></tr>
<tr><td></td><td colspan="2"></td><td colspan="2"></td><td></td></tr>
<tr><td></td><td colspan="2"></td><td colspan="2"></td><td></td></tr>
<tr><th colspan="6">Collective Context and Evidence of Participant Modules held to be consistent</th></tr>
<tr><td colspan="3"><i>Context</i></td><td colspan="3"><i>Evidence</i></td></tr>
<tr><td colspan="3">(e.g. Context C9, Assumption A2)</td><td colspan="3">(e.g. Solutions Sn3, Sn8)</td></tr>
<tr><th colspan="6">Resolved Away Goal, context and Solution References between Participant Modules</th></tr>
<tr><td colspan="2"><i>Cross Referenced Item</i></td><td colspan="2"><i>Source Module</i></td><td colspan="2"><i>Sink Module</i></td></tr>
<tr><td colspan="2">(e.g. away Goal AG3)</td><td colspan="2">(e.g. Module B)</td><td colspan="2">(e.g. Module C)</td></tr>
</table>

Table 1 - Safety Case Contract Table

In trying to apply this tabular approach to an example case study modular safety case a number of problems were encountered, including:

- It was unclear without more explicit examples, exactly what the safety case contract table was meant to cover, and how it was to be applied. In practice it was found to be difficult to capture all the necessary information in such a tabular form.
- There is no mechanism for capturing the strategy used in addressing one goal with another. This strategy could in many cases be fairly complex. In the same way that strategy (potentially with its own context and assumptions) may be needed to show how a goal *within* a module solves another, this may also be required where the solution is made across modules via the contract.
- The tables exist as completely separate entities from the GSN argument itself. This means that there is no visibility within the GSN structure of contractual links.

To address these concerns, the IAWG team have proposed an alternate approach to capturing safety case contracts. This approach is currently being trialled on an industrial case study.

5 IAWG Proposed Implementation of Safety Case Contracts

Based on the challenges identified above, the following solution has been proposed as a way of capturing the safety case contracts between safety case modules in the IAWG case study modular safety argument.

IAWG propose that the contract should be captured using GSN, as this provides an expressiveness and clarity which is not p' ovided by the use of a tabular approach. This also allows the contract to be integra.ed with, and viewed as part of, the total safety case argument.

5.1 GSN Contract Reference

The contract will be constructed as a GSN safety case module which can be referenced by the goal requiring support. This means that the module and goal providing the solution to the goal requiring support is not identified directly by that goal, but is instead specified in the GSN contract module. This allows the solution in the contract to be changed without the module containing the goal requiring support being changed. Figure 5 illustrates the notation that is proposed to indicate that a goal is to be solved using a goal or goals provided by other modules, using a safety case contract.

In the example shown in Figure 5, the goal 'Goal: No_Undesired_Comms' is to be solved via the safety argument contract 'Contract {Z}'. It can be seen that a new GSN symbol has been introduced to represent the contract module. This new symbol has been introduced here specifically to distinguish a safety case contract module from a 'normal' safety argument module; this is necessary as there are certain properties of safety case contract modules which do not apply to safety argument in general. These unique properties of safety case contract modules are discussed later.

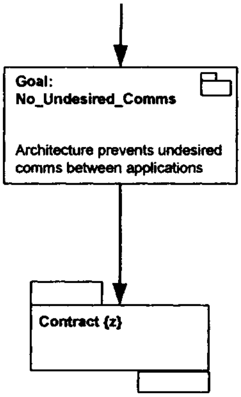


Figure 5 – Safety Case Contract Reference in GSN

It may be desirable to express the fact that a goal requiring support will be solved through use of a contract without specifically making reference to a particular safety case contract module. This may be desirable if, for example, the solution to the goal has not yet been defined in a particular contract module. In such a situation, the intention to support a goal requiring support through use of a contract can be indicated through using the GSN goal annotation proposed in Figure 6.

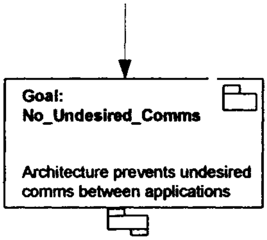


Figure 6 - Indicating a Goal is to be solved using a contract

Once a contract is developed to provide a solution to the goal, the contract can be referenced explicitly as in Figure 5.

5.2 GSN Contract Module

The contract itself is represented as a GSN module. This shows how the goal requiring support from one module is solved using a goal, or goals, provided by other modules. An example 'Contract {Z}' module is shown in Figure 7.

This contract shows how the unresolved goal 'Goal: No_Undesired_Comms' from the Applications module (identified using an away goal reference) is resolved using a goal 'Goal: Partitioning' from the Architecture module. A highly simplified version of the Architecture module, provided for illustrative purposes, is shown in

Figure 8. This goal is similarly identified using an away goal to 'Goal: Partitioning' in the Architecture module. A strategy is also provided.

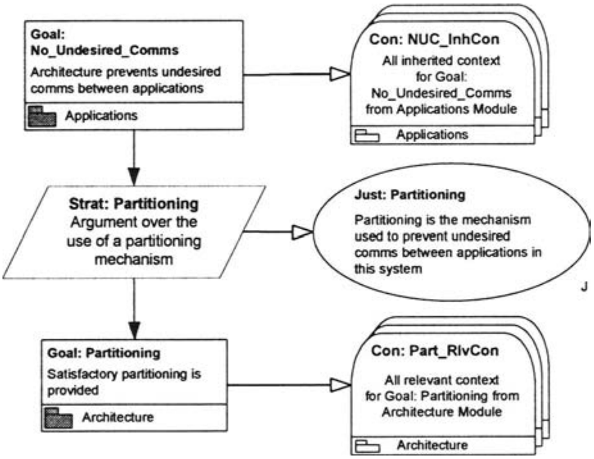

Goal: No_Undesired_Comms
Architecture prevents undesired comms between applications
Applications
Con: NUC_InhCon
All inherited context for Goal: No_Undesired_Comms from Applications Module
Applications
Strat: Partitioning
Argument over the use of a partitioning mechanism
Just: Partitioning
Partitioning is the mechanism used to prevent undesired comms between applications in this system
J
Goal: Partitioning
Satisfactory partitioning is provided
Architecture
Con: Part_RlvCon
All relevant context for Goal: Partitioning from Architecture Module
Architecture

Figure 7 - Contract {Z}

It can be seen in Figure 7 that the context relevant to each away goal in the contract must also be included. W ; discuss later the issue of identifying relevant context. Another new GSN symbol is required at this point in order to indicate that the context on the goal is a *collection* of existing contexts (in this case from other modules). The *away* context 'collection' symbol is illustrated in Figure 7. It should be noted that this symbol is equivalent to including many 'away context' references to each element of existing context in the other module, however the new 'context collection' symbol allows the presentation to remain less cluttered. The contextual elements that are covered by the 'context collection' must be stated.

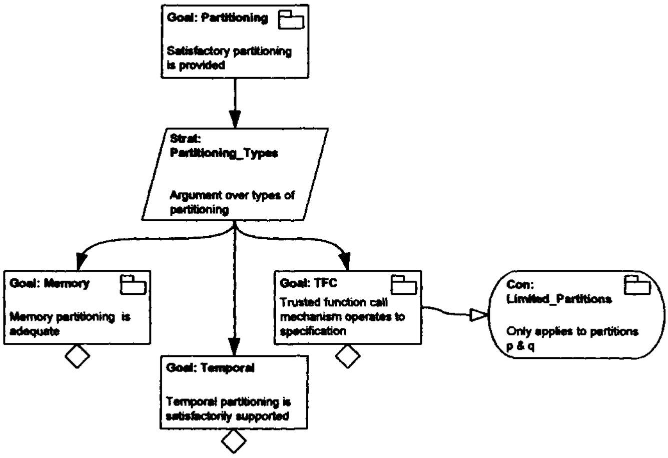

Goal: Partitioning
Satisfactory partitioning is provided
Strat: Partitioning_Types
Argument over types of partitioning
Goal: Memory
Memory partitioning is adequate
Goal: TFC
Trusted function call mechanism operates to specification
Con: Limited_Partitions
Only applies to partitions p & q
Goal: Temporal
Temporal partitioning is satisfactorily supported

Figure 8 - Architecture Safety Case Module (simplified)

The context relevant to 'Goal: No_Undesired_Comms' in Figure 7 is not only context directly connected to this goal in the Applications safety case module, but also 'inherited' context from any parent goals higher up the argument structure. Similarly, the context for goal 'Goal: Partitioning' must include not only all inherited context and context directly connected to the goal but also must take into account where lower level goals are reduced in scope by the use of contexts, assumptions or justifications. The reason for this is illustrated in Figure 8 where the context 'Con; Limited_Partitions' reduces the applicability of the solution offered from all partitions to only partitions p and q. It is therefore necessary for 'Con; Limited_Partitions' to be included as part of the collective context to 'Goal: Partitioning' in the Contract {z} module.

A justification is provided in the contract module through 'Just: Partitioning' which justifies why 'Goal: No_Undesired_Comms' is supported by 'Goal: Partitioning' within the scope defined by the inherited context of 'Goal: No_Undesired_Comms'. As seen, it is also possible to include a strategy between the goals matched in a contract module, if this is required.

5.3 Generic Pattern for GSN Safety Argument Contracts

The specific example above has been used to illustrate how a GSN safety argument contract module approach may be applied. It is possible to define a contract module in more generic terms as a pattern which can be ˈised in constructing a contract for any goal requiring support from other modules. A contract pattern is proposed in Figure 9.

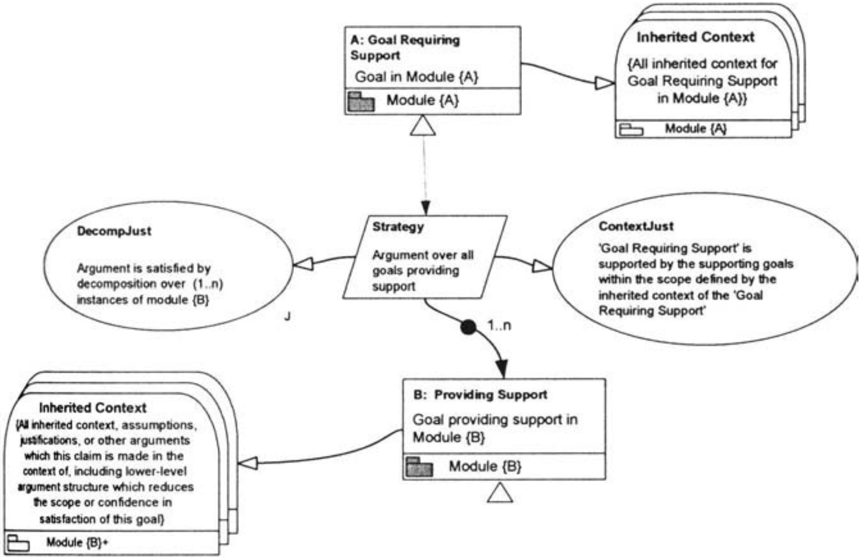

A: Goal Requiring Support
Goal in Module {A}
Module {A}
Inherited Context
{All inherited context for Goal Requiring Support in Module {A}}
Module {A}
DecompJust
Argument is satisfied by decomposition over (1..n) instances of module {B}
J
Strategy
Argument over all goals providing support
ContextJust
'Goal Requiring Support' is supported by the supporting goals within the scope defined by the inherited context of the 'Goal Requiring Support'
1..n
Inherited Context
{All inherited context, assumptions, justifications, or other arguments which this claim is made in the context of, including lower-level argument structure which reduces the scope or confidence in satisfaction of this goal}
Module {B}+
B: Providing Support
Goal providing support in Module {B}
Module {B}

Figure 9 - Generic pattern for safety case contract modules

Figure 9 illustrates how more than one goal from more than one module may, if required, be used to resolve the top goal. Strategy and justification elements may be used as necessary to make the argument clear.

5.4 Dealing with Context

When dealing with context in making safety argument contracts between modules, Kelly (2001) talks about agreeing the consistency between the collective context of the participating modules. In practice, this can be extremely problematic. The simplest way of showing consistency between collective context is if there is a direct match between the contexts. In reality this is never, as Kelly asserts, likely to be the case. It is unrealistic to expect that context which is inherited from modules which have been developed independently and in significantly different domains might match. Anything other than such a direct match is likely to make compatibility extremely complex to argue. For example, consider contract {Z} in Figure 7. It is possible that context defined in the Architecture m‹ dule which is inherited by goal 'Goal: Partitioning' may, for example, refer to mod :s of the operating system. Such information is unlikely to appear as context in the Applications module, as this module may have no knowledge of the modes of the operating system. Although the context of 'Goal: No_Undesired_Comms' and 'Goal: Partitioning' would, in this case not directly 'match', it doesn't necessarily mean that 'Goal: Partitioning' is not a valid solution of 'Goal: No_Undesired_Comms'. Instead, what is required to be shown in making the contract is that 'Goal: Partitioning' satisfies 'Goal: No_Undesired_Comms' within the inherited context of 'Goal: No_Undesired_Comms'. It must be possible for the context of participating modules to be simultaneously "true", otherwise the composed argument becomes unsound.

5.5 Incorporating Contract Modules into the Safety Argument Architecture

It is possible to consider the safety argument contract module as part of the safety argument architecture as indicated by the module view in Figure 10.

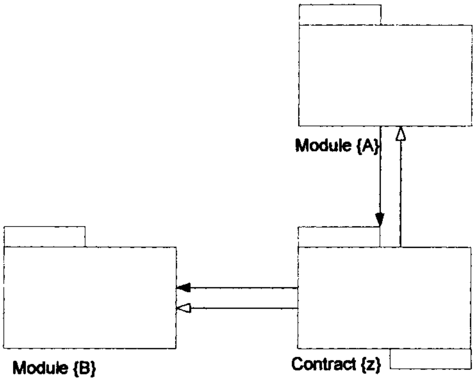


Figure 10 - Including a contract module in the safety argument architecture

Whether the contract modules are represented as an integral part of the architecture or instead, one contract module is produced, with the individual contracts being represented as separate *views* of this module is yet to be resolved, but should not affect the way in which the contracts are developed.

5.6 Notes on Away Goal Decomposition

It should be noted that normally when using GSN it is considered invalid to decompose an away goal. This is because an away goal is merely a reference to a 'real' goal defined elsewhere that may or may not be supported. Therefore, to provide a solution to an away goal is to 'support' a reference to a goal rather than to support the goal itself. In the safety case contract pattern shown in Figure 9, it can clearly be seen that a solution is provided for the away goal referencing the goal requiring support. The purpose of a safety case contract module is specifically to show how a goal *in one module* is supported by a goal *from another module*. In such cases (and only in such cases) we contend that it *is* valid to provide a solution for an away goal. It is important to note that even within a ccntract module, it is only permissible to decompose an away goal that refers to a goa^l requiring support. It remains invalid to provide a solution to an away goal that is already supported within its own module, such as goal 'B: Providing Support' in Figure 9. The following property of a safety case module can thus be defined:

- Within safety case contract modules it is valid to decompose away goals which refer to a goal *requiring support* from another module.
- Conversely, the goal requiring support, which is addressed via a contract, must not be decomposed in its host module.

6 Summary

The Ministry of Defence has funded the IAWG to develop a modular safety case and, in doing so, some issues were identified in the definition and use of safety case contracts for modular GSN as defined in Kelly (2001). In discussion with Kelly, IAWG have developed alternate solutions, which IAWG hope will be encompassed within an updated definition of modular GSN in the future. The main points of the alternative solution, as presented in this paper, are summarised below

- A GSN safety case contract module has been proposed as a method of capturing the contract between safety argument modules
- Implementation guidelines for GSN safety contract modules have been developed and are recorded below
- A pattern for GSN safety case contracts is presented in Figure 9
- Assessing and arguing context compatibility is a known and complex issue which requires further work

6.1 Implementation Guidelines

In defining the solution proposed in this paper, a number of process constraints were identified which should be followed when implementing safety case contract modules in GSN. They are summarised below:

- It should not be possible for anyone other than the 'owner' of the safety case module to change the public/private status of a goal.
- It may be necessary to indicate in some way the public goals that are unused for a particular safety case, such that it is clear that the goals are available to other modules, but in this particular case are not required., i.e. provide traceability of public goal usage

- Within safety case contract modules (but *only* within safety case contract modules) it is valid to decompose away goals which refer to a goal requiring support from another module.
- Where a goal requiring support is addressed via a contract, the goal must not be decomposed in its host module.

7 Acknowledgements

The work described here has been undertaken by the Industrial Avionics Working Group, funded by the Ministry of Defence, with support and advice provided by QinetiQ, acting as Independent Safety Advisor, and Tim Kelly of the University of York.

8 References

Kelly, TP (2001). Concepts and Principles of Compositional Safety Cases - (COMSA/2001/1/1) - Research Report commissioned by QinetiQ

Hofmeister, C., Nord, R., Soni, D (1999). Applied Software Architecture, Addison-Wesley

THE SUM OF ITS PARTS

John Spriggs
NATS
Whiteley UK

Abstract

A safety case is a collation of the arguments and supporting evidence needed to demonstrate that an item is safe to use. Although there are regulatory and legislative requirements in some industries, and in some parts of the world, for the production of a safety case, there are few standard definitions of what should go into one. Indeed, there is no single, corre :t, way to prepare a safety case; but this should not be surprising, ɩ iven the wide variety of applications to which they are put.

Even after consigning large quantities of evidence to referenced documents, the safety case for a complex item is often large and unmanageable. It can be made more manageable by structuring the arguments using one of the graphical methods that are now available. The result can still be unwieldy, however. This paper presents a practical way to partition safety cases that facilitates through-life maintenance of the documentation, but is also suitable for the development phase of a system.

Introduction

In general, a safety case is a set of arguments, supported by evidence and explicitly stated assumptions, intended to show that an item is safe for use in a given environment. Clearly, this requires augmentation with a definition of what the item is, and also what constitutes "safe" in that context.

The safety case of an item therefore contains:

- A description of the item in question, including its purpose and operational concept;
- A definition of safety, of what is tolerable in this context, and of any additional behaviour required of the item to achieve the target level of safety;
- An argument showing that the item as designed will have the required safe behaviour, with supporting evidence and assumptions;
- An argument showing that the implementation of the item has preserved the safety features, with supporting evidence and assumptions;
- An argument showing that the item can safely enter service and that it will remain safe when in service, with supporting evidence and assumptions; and,
- In a regulatory environment, arguments, with supporting evidence and assumptions, addressing the additional requirements that the Regulator expects the safety case to address explicitly.

This list implicitly requires additional things, such as a description of the configuration of the item and how that configuration is to be managed throughout the lifecycle. In particular how proposed changes to the item are assessed as to their impact upon the safety arguments. Also, the safety in service argument may depend upon certain information being provided to the users and maintainers of the item. This information could be contained, for example, in handbooks and/or training material. Whatever means of promulgating the information is chosen; its effectiveness also needs to be considered as part of the argument.

In NATS we provide Air Traffic Services to aircraft, and have many safety cases for different aspects of those services and the infrastructure used to provide them. They tend to fall into three categories:

- Operational Unit Safety Cases – these provide assurance that operational Air Traffic Control units, for example those based at airports, are safe for continued use.
- Facility Safety Cases – these provide a: surance that the infrastructure provided to support operational Air Traffic Control, for example radio stations, are safe for continued use.
- System Safety Cases – these present an overall justification for the claim that a system is safe to be introduced into operational service. The safety case is then maintained throughout the life of the system to demonstrate that it is suitable for continued service.

This paper concentrates on the last of these, as it is of more general applicability, and the principles are more readily read-across to other industries. A system in this context can be something localised like radar sensor or a telephone exchange, or it can be something much larger, such as a set of geographically separated sensors, communications networks, actuators and displays used to collect data and present it to an end user for a safety-related application.

The First Cut

Many, if not all, safety arguments are based upon the presentation of evidence that safety requirements have been fulfilled; this will usually be in the form of results from testing, numerical analyses, demonstrations and simulations. Backing evidence will also be presented, such as quality records showing that a robust development process has been followed. If this is taken literally, and all this information is held within one document, it will be very large, difficult to use and practically impossible to maintain.

However well structured, a full safety case can be unwieldy; it clearly needs to be partitioned. There is one obvious first cut to be made; most of the supporting and backing evidence will also be used for other purposes and can reside in their own documents, to which the safety case and other users may refer. For example, fault tree analyses may be used in maintenance planning for an item, whereas test results can help define its operational envelope. It is, of course, common practice to refer out to other documents that contain the required information, rather than record that information in the safety case itself. What is sometimes forgotten, however, is that a process requirement arises from such a strategy.

The organisation needs to put in place a configuration management system, and a records retention policy, that ensures all the references will be maintained, and will survive as long as the safety case itself. Archiving all the information together physically at each configuration baseline is good practice.

Despite having consigned large quantities of evidence to external documents, a real safety case is still likely to be large, unwieldy and difficult to manage. Papers at previous Safety-critical Systems Symposia have observed that safety cases can read like rambling Victorian novels if not structured appropriately. Graphical methods for structuring safety arguments have been developed and presented. There is, however, still the danger of producing the overall safety case of an item in the form of an illustrated Victorian novel. What is needed is a further partition; but upon what basis?

Referring to the safety case of an "item" glosses over a lot of complexity. In most practical examples, the item will be a system comprising multiple platforms, people and their procedures. The platforms are likely to be made up of equipment items procured from different sources; indeed several companies may well develop the hardware and software for one equipment item. Can we achieve partition just by requiring each to provide a safety case for their bit? Yes, we can, but it is not a very structured approach; in particular, the interfaces between the various suppliers' equipment may not be dealt with properly – things can get missed as each assumes it is the others' responsibility. There is also the problem of balance and scale; one supplier may be required to do a safety case for a large installation, another for an antenna or a modem, for example. Perhaps such devices do not need their own safety cases, but who should incorporate them into their bit? It can get very complicated procuring equipment from one supplier for free issue to another, who will integrate it with their equipment and do a safety case for the combination.

There are ways of managing such a situation; one approach is to have an overall system safety case that references out to evidence, which may include subordinate safety cases, obtained from all the suppliers. But that does bring us back to the question of how sensibly to partition the overall safety case.

Partition by Persistence & Purview

Two parameters that you may not have considered as the basis for a partition of a safety case are persistence and purview. They would often be inter-related and so will be discussed together.

The purview being considered here is not that of the system, but rather that of those who have to review, approve and accept the safety case, i.e. the signatories and their advisors. They will have different competences and scopes of authority. A complete safety case for a large system would, in principle, need to be signed-off by all those affected: the service owner and operator; the sub-system owners; operators and maintainers; those concerned with monitoring and control; et alia. We can arrange a cohesive partition of the safety case based on rôles and responsibilities of the signatories.

Persistence in this context is not exactly a lack of, or resistance to, change, but a lack of need for frequent change. Look at the content of a safety case for a large service provision system. It contains everything from the overall objectives and requirements of the service, through to procedures for day-to-day monitoring and control. A real system will be made up of new items and legacy items. Spare parts become increasingly difficult to obtain for the legacy hardware items, whilst random failures increase as they approach the end of their design lives.

If we were to replace such an item with a new equivalent, the overall design and implementation would change, but the service objectives and requirements would remain the same. So why not split off the objectives and requirements part of the safety case from the rest? It will become the top level document, only changing as the service needs to change. It will be reviewed by experts in the service domain and signed-off by the service owner, the nominated future operator and the corporate duty holders.

Similarly, an item may be refurbished. The design remains the same, but the implementation assurance has changed; there will be new test results, performance trials, etc. The design part of the safety case can be split off from the rest. This part would be reviewed by systems engineers and logistics engineers before being signed off by the system owners and operators who they advise.

An early step in the design process is the apportionment of the system requirements to the various sub-svstems. In a large, geographically distributed, service provision system this appo tionment may be done on a regional basis to take into account the different operational and environmental mitigations that apply. For example in the air traffic services, different apportionments may be done for en-route airspace and the more complex terminal area airspace. The apportionment parts can be hived off from the rest of the safety case, reviewed by local domain experts and signed-off by the regional service owners and operators.

A key feature of many of the new systems introduced into service by NATS is that they replace existing in-service systems. The part of the safety case that covers the installation, testing and commissioning therefore also needs to address the transition activities during which one system is brought "on-line" and the old one is removed from service, whilst preserving continuity of service for the end-users. This part of the safety case can also be published separately. It would be reviewed and signed-off by representatives of those who will own the system and those who will operate it during the transition and beyond.

Consider all that has been removed from our overall safety case throughout this discussion, to be published as separate volumes. All that is now left is the in-service (and eventual removal from service) part. Depending upon the nature of the overall system, this could be subsumed into one of the Operational Unit, or Facility Safety Cases mentioned earlier. Alternatively, it could be a logistics safety case specific to the system – it can even be in separate parts itself, addressing separately managed sub-systems.

That last remark applies to all parts of the safety case, the sub-system or supplier-based partitions may be used as well as that based on persistence and purview. Multiple safety requirements documents would be unusual; there is, in general, no advantage in splitting the safety requirements into separate volumes. But consider the evolutionary nature of systems. There may be an existing system that needs to be replaced; the opportunity can be taken to add new functions. In many cases the original document would be up-issued to include the new requirements, but there are occasions when the new function is far reaching, requiring updates to several major sub-systems – sensors, data transmission means and user equipment. Just on purview grounds, it can be more effective to keep the new function's safety requirements separate. The bringing together of the two sets of requirements can be done in an update to the apportionment documents or as part of a new design.

An example of such a wide-ranging new function is the datalink for use in exchanging Air Traffic Control related information with aircraft. It uses as its communications channel extensions to existing VHF communications facilities, or to Mode S secondary surveillance radar. It can also use satellite communications systems, High Frequency radio and other legacy communications systems. Its safety requirements would not naturally fit with any of these systems in preference to any of the others; the requirements can be captured separately and then feed into each of the other safety cases.

In summary, the foregoing discussion has identified four major parts for a safety case, the content of each will be considered in more detail in a later section. The four parts identified are:

- Part One ~ Safety Objectives and Requirements
- Part Two ~ Design and Apportionment
- Part Three ~ Implementation and Transition
- Part Four ~ Operation and Maintenance

A Note on Review and Agreement

A further advantage of partition by purview is that the reviewers and signatories are looking at a document that remains largely within their domain of influence. They should not have a huge tome to wade through, looking for the bits of interest. Not being an onerous task, the review is more likely to be done when requested, rather than being put off in favour of more attractive tasks. Similarly, when the document is up-issued, there is less extraneous material deflecting attention from the change that has been made.

The partition of a safety case by purview and/or persistence is, in effect, a version of the principle of incremental safety case development. Incremental development is when you involve the reviewers as the arguments, etc., develop. They first see and agree an outline structure, effectively a plan, of the safety case. In this version the basic arguments are outlined, identifying what evidence will be required and from where it will come. As more detail is added, they agree the changes, e.g. at monthly meetings, and so when it comes to final sign-off, they will be confident in what they are agreeing.

Unfortunately, it is not always possible to secure the services of the reviewers to that extent, although it is highly recommended as a way of managing the interface with suppliers or Customers. Enforcing a partition, with separately reviewed parts, achieves the same objectives, albeit in a coarser manner. For something completely new, plan to produce several drafts of each part for review; for something more familiar, fewer review cycles would be required.

In practice, it will be necessary to obtain agreement of, denoted by signatures on, a safety case (or a part thereof) at significant stages in the development of the item it describes, e.g. before it enters the test and evaluation phase, or before it goes into service. These will be major programme milestones; we do not want it always to be Safety Assurance who is seen to be holding them up. But that is what will happen if the review and agreement of the safety case does not start until the last bit of evidence has been recorded therein.

It is not necessary to wait; the arguments can be agreed in advance and it is known what evidence is needed to demonstrate their validity. It is only necessary to manage the documentation of that evidence such that it is established in advance the identity (and issue state) of the documents in which the evidence will be recorded. These documents can then be referred out to by the safety case. When the safety case is otherwise ready for review, prepare a list of those items of evidence that are not yet available. Turn it into an action list identifying who is to prepare each item and by when; publish it as a formal document, possibly identified as another part of the safety case. The safety case can then be sent out for review and agreement, with the caveat that it is only valid for use when all the actions on the list are complete.

The last test is successfully completed, the system is deemed ready for operation; it is only necessary to get the responsible manager's signature to acknowledge closure of the action list and it can go into service without delay.

A Bit More Detail

The Safety Case Part One ~ Safety Objectives and Requirements

Before a system can be designed, it must be specified: what are its objectives, what is the concept of operations? Part One of the safety case is developed whilst those aspects are being formulated. The content is restricted to the first two bullet points in the Introduction to this paper. The first calls for a description of the item in question. At the beginning of a development programme, however, one does not have a detailed description of that which is being developed. What is required at this stage is a description of the intended rôle and the functions required to fulfil it. Diagrams are useful here to show the interrelationships between functions. Any regulatory requirements arising from this intended rôle will need to be identified.

The second bullet point, in effect, calls for a risk classification scheme and a specification of any additional behaviour that is required to achieve tolerable risk in that context. That specification constitutes the functional safety requirements. If the risk classification scheme has not already been agreed, e.g. as part of an extant safety management system, it should be derived and justified in the safety case as part of the definition of safety and what is tolerable in the particular context.

It is not sufficient just to state the requirements. The process by which they were derived should also be described, although this could be by reference to an external standard, or the company's safety management system. A practical approach to deriving the safety requirements is to consider the unwanted outcomes that could arise from the use of the system with reference to the risk classification scheme.

Undesirable outcomes are often expressed in terms of hazards, but some systems, although complex, may have a simple set of such outcomes that are not of themselves hazardous. Consider a data communications system for example. Its rôle is to provide, in a timely manner, data at point B that are the same as those input to point A. The unwanted outcomes are:

- Loss: in which no data arrive at point B;
- Corruption: in which the data arriving at point B are different from those sent from point A; and
- Delay: in which the data arrive, but later than required.

It is the application to which the system is to be put that determines the possible hazardous effects of these outcomes. One application-specific factor that needs to be taken into account is time. The system may be such that a loss of data transmission for a minute, say, may be acceptable, but any longer would cause a problem. The time from the onset of a failure condition until the onset of a safety effect is known as the Significant Period of Time (SPoT) and it should be explicitly included in statement of safety requirements.

Part One of the safety case captures the functions to be performed by the system along with the Safety Objectives, Integrity, Functional, Performance and Regulatory Safety Requirements that apply. The distinction between Safety Objectives and Safety Requirements is made because certain operational and environmental mitigations can be taken into account in formulating the requirements.

The process starts with the identification of a particular functional failure condition or unwanted outcome. Domain specialists are involved in the process and are able to assess the severity of the consequences should it occur. The risk classification scheme is then used to identify what level of risk has been deemed tolerable for such an outcome. Risk is a two dimensional quantity, encompassing both the likelihood of a particular outcome and the severity of its consequences should it arise, but by this point in the process the severity has been defined, so the likelihood is the only variable. We can represent it as a line; see Figure 1, in which the lowest risk is to the right.

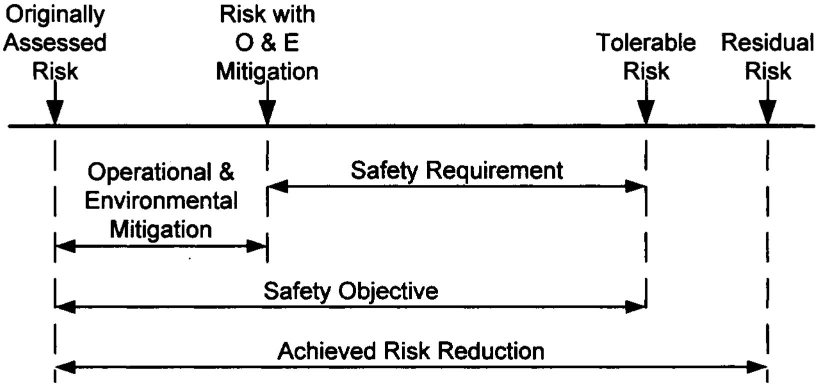


Figure 1 ~ The Relationship between Safety Objectives and Safety Requirements

The safety objective states that the likelihood must be the "tolerable" level (or less). The amount of risk reduction needed to achieve this objective is shown by the horizontal arrow labelled Safety Objective in the figure. Some of the notional risk has already been reduced by other factors known to the domain experts, such as operational profiles, seasonal differences and the like. The risk reduction actually needed is thus that required by the safety objective, minus the operational and environmental mitigation. This is labelled Safety Requirement in the figure.

One is striving to reduce the risk as much as is reasonably practicable, so the final achievement should exceed the objective, as shown in Figure 1. In practice, of course, the planned activities can result in a residual risk that is in excess of what is tolerable; more work must then be done, leading to additional mitigations.

In a large, geographically distributed, system there may be different requirements in different regions addressing the same objective. Whether the regional apportionment belongs in the Part One, or in a different set of regional documents depends on the nature of the system under consideration, on the organisation developing it, and on the eventual operating organisation. The important thing is that the partition be planned in advance.

Further apportionment of safety requirements to sub-systems would be part of the design activities, as it is, in general, during design that the decomposition into sub-systems is first defined.

The Safety Case Part Two ~ Design and Apportionment

Part Two of the safety case addresses the design, showing that the item as designed will have the required safe behaviour, not only in operation, but also during installation, commissioning and transition to service.

The description of the system can be more detailed than in the Part One and will contain the decomposition into sub-systems, with apportionment of safety requirements thereto. It must also a clearly identify the boundary of the system under consideration, i.e. the scope of the safety case, and its interfaces with other systems and facilities.

Apportionment to sub-systems may be done arbitrarily for a new system; for example, if five items serially contribute to a particular requirement, they may be each assigned twenty per cent of the target. However, many large systems are designed to include items that have been deployed successfully in similar systems elsewhere; knowledge of their performance in those contexts may therefore be used to apportion the requirements in the new system.

All sub-systems need to be identified, not just the "glamorous" ones like a surveillance data processor or a satellite communications transceiver. For example, if continuity of operation is a concern, you will need to consider whether it will be interrupted by the fire alarm going off, a failure of air conditioning, a power cut, and so on. Figure 2, overleaf, shows just some of the items that may be included in a Safety Case Part Two.

It is not just the internal sub-systems and interfaces that need to be considered; are there dependencies on other, external, systems? These need to be identified; if those systems were to fail, how would the system under consideration continue to fulfil its safety requirements?

Are there any limitations on the use or maintenance of the system as designed? These also need to be clearly identified.

For each safety requirement, either identify the evidence that confirms (or otherwise) that it will be met, or identify from whence such evidence will be obtained, and when. For example, there may be a requirement on the performance of a sensor that cannot be verified until it has been installed and tested in situ.

Where it is predicted that an apportioned safety requirement will not be met, the apportionment can be adjusted accordingly, tightening the targets for all the other portions. If the apportionment is not defined in the Safety Case Part Two, the document in which it resides will need a formal update to reflect the change.

A complex system does not just go into service; there will be a period of evaluation and testing. Part Two of the safety case needs to detail the assurance that these activities are tolerably safe, including identification of any additional safety measures required during installation, commissioning and integration. For an Air Traffic Management system, we also have to demonstrate that none of these activities will have an adverse effect on the safety of the existing services to air traffic.

Much of the assurance at this stage will be based upon design documentation. It is necessary to show that this is controlled and reflects what will be implemented. A summary description of the design process used and how it is effective in exposing and correcting design errors is also needed, especially for software designs.

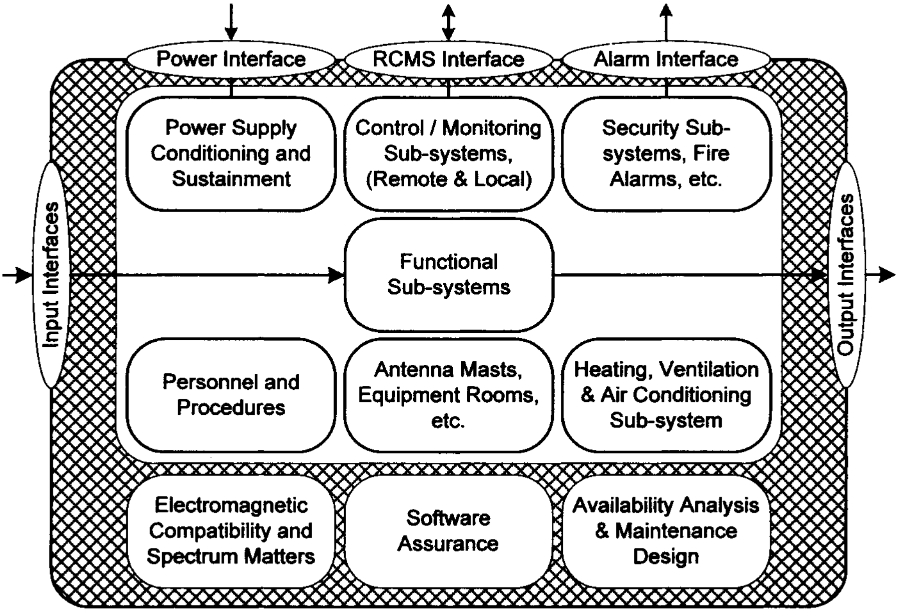

Power Interface
RCMS Interface
Alarm Interface
Power Supply Conditioning and Sustainment
Control / Monitoring Sub-systems, (Remote & Local)
Security Sub-systems, Fire Alarms, etc.
Input Interfaces
Functional Sub-systems
Output Interfaces
Personnel and Procedures
Antenna Masts, Equipment Rooms, etc.
Heating, Ventilation & Air Conditioning Sub-system
Electromagnetic Compatibility and Spectrum Matters
Software Assurance
Availability Analysis & Maintenance Design

Figure 2 ~ A System is Not Just the Functional Sub-systems

Not all aspects of the design will necessarily survive implementation unscathed; we also need to assure the safety of the design as implemented.

The Safety Case Part Three ~ Implementation and Transition

It is the Safety Case Part Three that demonstrates that the system as implemented meets all its safety requirements, and hence its safety objectives. It also describes any specific requirements (and their fulfilment) to assure safety during operation and maintenance. For a system that is taking over from, or augmenting the operation of, another system, the Safety Case Part Three also needs to provide an argument, with evidence and explicit assumptions, that it will not affect the safety of the service during transition into service.

Experience has shown that plans do not come to fruition quite as intended. If there are any shortcomings in the system as implemented, these are to be highlighted, and any special measures needed to compensate for them identified. Any limitations on use or maintenance of the system must also be made explicit, as must any dependencies on external systems. The lists of limitations and dependencies may just be an endorsement of what was stated in the Safety Case Part Two, but they may have additions arising from the implementation.

For each of the safety requirements, either provide confirmation of the fulfilment statement made in the Safety Case Part Two, or present the evidence that it is now met. This could be test results or analyses, or both. Again it may be necessary to adjust the apportionment between sub-systems (remembering to up-issue the apportionment document formally). Confirm that each of the original high-level objectives in the Safety Case Part One is now achieved.

If an objective has not been achieved, use the risk classification scheme to assess the resulting risk class. Either more work will be necessary, or the elevated risk will be deemed acceptable due to external mitigation not previously taken into account. Whichever course of action is taken, the decision needs to be recorded and justified in the safety case. If an external mitigation is to be used, it will need to be added to the list of dependencies.

If there are explicit regulatory requirements, they may be addressed either by mapping to other system requirements and evidence of their fulfilment, or by reporting on the additional verification activities carried out.

It is, of course, necessary for the safety requirements to continue to be met when the system is in service. If any aspects of system performance need to be monitored in service to ensure this, they should be identified in the Safety Case Part Three, with the mechanism that will be used. If procedures for such monitoring are not already extant for other systems, they need to be designed and put in place at this stage.

The system should now be ready for transition into service.

The Safety Case Part Four ~ Operation and Maintenance

This part of the safety case is required before handover of the system into operation and its acceptance should be a criterion of the handover decision. This is because it provides the safety assurance for sustained operational service. In practice, this part may be an update to an existing Facility or Unit Safety Case, rather than a new document. If it is a separate document, other safety cases in the wider operational system are likely to require update anyway, if only to refer out to the new system.

The purpose of the system is to be stated, along with any limitations on its use and maintenance, and any shortcomings in the design as implemented. Dependencies on external systems also need to be stated. Operators' procedures, for both normal and abnormal system operation, monitoring and control need to be identified. These will tend to be in manuals or operator cards, which can be externally referenced.

Similarly, there will be engineering procedures for what needs to be done to sustain the system in operation and to restore it to service in a timely manner in case of breakdown. If the system provides a critical service, whether that be safety criticality or financial criticality, there should be realistic service level agreements for maintenance set up and referenced from the safety case. The response times to repair given therein could be used as assumptions in the safety argument, and will need to be validated.

It is all very well to have procedures in place for operators, maintainers and those who are expected to carry out the performance monitoring specified in Part Three of the safety case - but do these people exist? Rôles and responsibilities need to be defined, or existing ones augmented to encompass the new system. The presence of adequate numbers of personnel needs to be confirmed. This is not just a counting exercise; each staff member needs to be adequately trained for their assigned tasks. Refresher training n.ay be required for some tasks.

The Safety Case Part Four is the final link in the chain from when the safety objectives and requirements were derived. Its first issue should conclude with a clear statement that the system can be put into operational service. It should, but if insufficient evidence has been brought to support the arguments presented, it will not. If the decision has been made not to proceed into service, due to insufficient safety assurance, the reasons must be stated in the first issue of the Part Four. An action plan can then be put into place to resolve the problems. Alternatively, if the decision is a "Go", as it should be following this phased agreement of the safety case, the system enters service and the overall safety case enters its maintenance phase.

Each part of the safety case, once it is agreed and formally issued, is subject to maintenance. By the time the Part Four is issued, for example, the Part Two may be at Issue Level Three due to adjustments of apportionment and an originally specified item being upgraded, or even unavailable. The Part Three may be at Issue Level Two to reflect the regression testing following the change of item. Part One is likely to be unchanged.

Conclusion

It is a foregone conclusion that the safety case of a large and complex system will itself be large and complex. It is already standard practice to reference out to build state lists, test results, analysis reports and such like, rather than to include all the information in one huge safety case document. This paper has addressed the problem of making what remains more manageable and maintainable, by splitting it with regard to purview of the responsible parties and the likely persistence of the various parts.

Others have described solutions derived from different principles. For example, the railway standards partition a system's safety assurance such that those subsystems that are "generic products", i.e. COTS items, get to have their own safety cases, whereas the rest, and the overall system, have Design and Implementation Safety Cases. These latter documents are similar in scope to the Part Two, described above, for Design and an amalgam of Parts Three and Four for Implementation. (Nordland 2003)

Another partition that has been made public depends on the acquisition lifecycle of the systems authority. Development of the system safety case is such that baselines are ready for the "gateways" at which the decisions to proceed on to the next stage of development are made. The first part of the safety case in this scheme is similar in scope to our Part One described above; the second part is a combination of the Parts Two and Three, whereas the third part is similar to the Part Four. (Howlett 2003)

The scheme described here encompasses a lot of the same principles as these two examples, but it is more focussed on the persistence of the documents and the maintenance of them as a cohesive whole. Ideally those bits that need to change frequently will be small and self contained, whereas those that are unlikely to change much can be large and exquisitely detailed. Once the safety objectives for a system or service are developed, validated and agreed, one could expect them to remain stable for a considerable proportion of the system lifecycle.

Where operational and environmental mitigation has been used extensively to derive the associated safety requirements, there is the possibility of more frequent changes, as conditions change, so this aspect of the Safety Case Part One could be split out to another document, or documents if there is the expectation that some aspects will change at different times, for example due to operation in different geographical regions. Such a split can also be done taking into account the purview of the domain experts who provide the information that is the basis of the transformation from safety objectives to apportioned safety requirements.

The design of the system is more likely to change throughout the lifecycle than its requirements. During development this may be due to changes in the apportionment of safety requirements to subsystems. During operation, which is (intended to be) the longest lifecycle phase, the Safety Case Part Two will change as components are replaced due to obsolescence, or upgraded after investigation of anomalous behaviour, and due to the occasional changes in the Part One and/or the operational and environmental mitigations.

The Safety Case Part Three will change for most, if not all, changes of the Part Two. Furthermore, there will be additional events that leave the design unchanged but require changes to the Part Three, for example refurbishment of some equipment item would not change the design, but would require regression testing and transition back into service.

The persistence of the Part Four is not really a concern if it has been subsumed into an existing Unit or Facility Safety Case; they will continue to change as other systems change as well as the one under consideration. If the Part Four is a stand-alone document, the main drivers for change will be improvements in operating procedures, or organisational changes, in addition to the changes coming through from the other Safety Case Parts. For example, a software design change may be made to overcome a previously declared shortcoming, and so the procedural mitigation for that shortcoming can be removed when the safety argument, as modified, is agreed.

Whatever partition is decided upon, in order to be effective it must be planned in advance. Define the scope of each part and their interdependences. Specify who should review and agree each part; preferably by name, but certainly by rôle. Gain their agreement on the partition and establish their expectations as to the eventual content of each part.

Different rôles are involved in the detail of the different parts of the safety case, so they can review and sign-off the documents for which they are stakeholders without having to plough through a lot of other material to do it. In practice this facilitates document maintenance and saves time. An even bigger time saver is the principle of conditional approvals – the documents are reviewed and agreed such that the safety case is signed-off, but does not become valid until all the actions required to collect test evidence, for example, are complete. When the decision is made to proceed into service on the basis of a completed action list, you can so proceed, without having to wait for another round of reviews and document agreement.

References

Nordland, O (2003) Safety Case Categories – Which One When? In: Redmill, F & Anderson, T (eds.) Current Issues in Safety-critical Systems. Springer-Verlag, London, 2003, pp 163-172

Howlett, R F (2003) Processes for Successful Safety Management in Acquisition. In: Redmill, F & Anderson, T (eds.) Current Issues in Safety-critical Systems. Springer-Verlag, London, 2003, pp 189-200

Lessons in Safety Assessment

Independently Assessing Legacy Safety Systems

Paul Edwards, Andrew Furse and Andrew Vickers
Praxis High Integrity Systems Limited
Bath, England

Abstract

It is not uncommon for large-scale safety-related engineering projects to make use of legacy systems that are either reused as is, or reused with modifications. Projects of this type can require the use of independent assessment as part of the regulatory approval process. Providing an independent assessment of such a project brings with it particular challenges related to issues of cross-acceptance, of the value of historical operational claims, and of supply chains involving international organisations sometimes with their own independent assessors. In this paper the various characteristics of large-scale engineering projects are outlined, together with their impact on an independent assessor. Lessons for independent assessors are identified.

1 Introduction

Large-scale safety-related engineering projects are typically complex and challenging. Many projects of this type are either required to use (for example because of regulation), or voluntarily make use of, independent safety assessment (ISA). Independent safety assessment is there to provide a second, ideally commercially un-pressurised, view of the extent to which safety risk has been reduced for a particular engineering project. The relationship between an independent safety assessor (ISA) and a project is often a key relationship: at best it can be a valuable 'second pair of eyes' spotting problems before they occur and underwriting key engineering decisions, at worst it can be a programme distraction that adds delay and uncertainty. The relationship can be influenced by key engineering decisions that are taken by the project. One such area of decision making is concerned with the use of legacy equipment designs.

Significant benefit can often be gained from reusing designs or equipment from similar projects – benefit in terms of both reduced development risk, reduced safety risk, and potentially reduced ISA cost/risk. The use of such legacy equipment, however, is not problem free – different projects are rarely exactly the same, either in context or in application, and these differences require management. Specifically, gaining an ISA's buy-in to the particular legacy strategy can often be key if full benefit is to be gained from the reuse of safety information.

The authors of this paper have played key roles in the independent safety assessment of a number of large-scale safety-related engineering projects that have adopted a significant reuse strategy. During this work a number of themes have been identified that effect how successful the reduction of ISA cost/risk can be for the use of legacy systems. The purpose of this paper is to share these themes. Although the discussion is based on experience mainly within the rail market, the authors believe that these themes are also applicable in other domains.

Sections 2, 3 and 4 are by way of background. They respectively outline the example characteristics of the large-scale safety-related engineering projects to which we refer, outline the key features of the role of an ISA, and summarise the benefits that can be gained from a legacy strategy.

Section 5 then summarises a set of issues that we believe should be considered by an ISA when considering how to cost-effectively assess this class of system.

Section 6 collects together the lessons before Section 7 concludes.

2 Large-scale and Legacy Engineering Projects

To put into context the challenges that ISAs face in assessing legacy systems within large-scale engineering projects, it is useful to identify some of the characteristics that these large-scale projects present.

These characteristics fall under two main headings. Firstly there is diversity and breadth: large-scale rail projects involve a wide range of participants, stakeholders and disciplines and, frequently, diversity across international boundaries. Secondly there is the complication of legacy, both legacy in the environment into which the new project must fit, and legacy in the use of pre-existing equipment or equipment design. Figure 1 summarises some of these characteristics. based on the authors' experiences with rail projects.

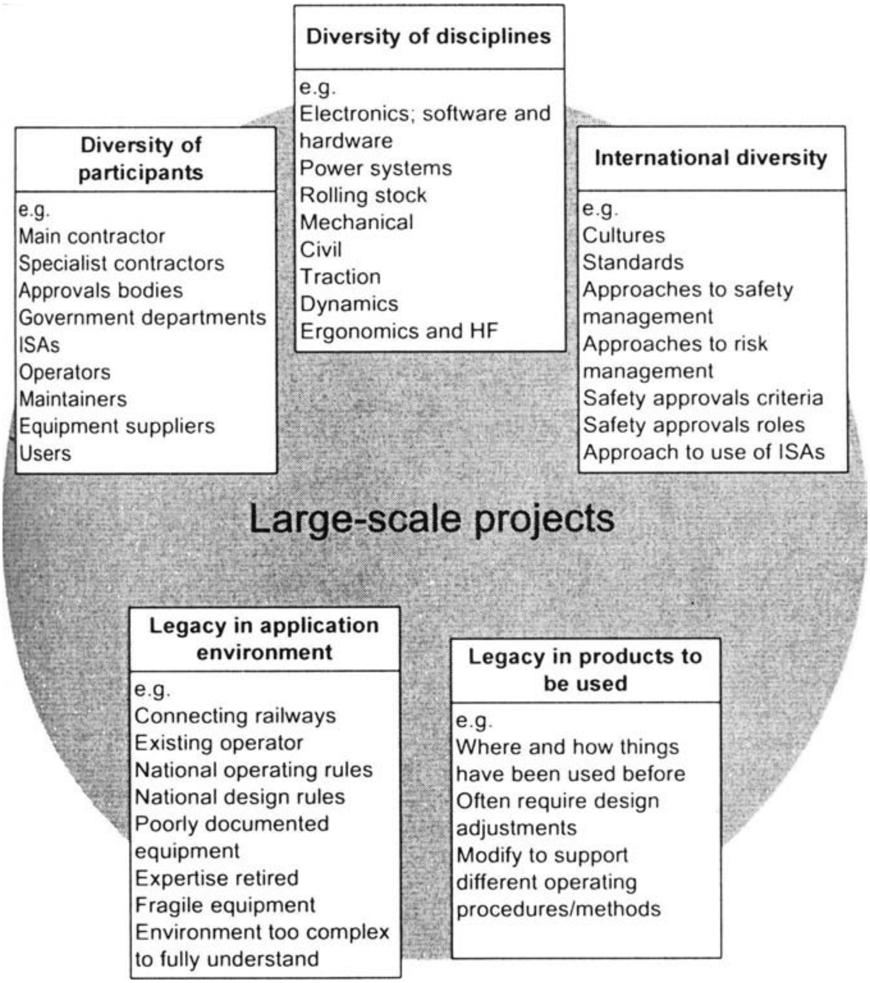


Figure 1: Characteristics of large-scale rail projects

3 The Role of the Independent Assessor

This section gives an overview of why ISAs are used, and the role that they have in the safety management and safety approval of projects. The main focus is on the UK and European rail industry, but there is also a brief mention of the use of ISAs in other industries.

Within the European Union, safety management of railway safety-related control systems is covered by a set of three main process standards: EN50126

(1999) (safety processes), EN50128 (2001) (software) and EN50129 (2003) (safety evidence). These standards, often referred to as the 'CENELEC standards' broadly align with IEC61508 (1998, 2000), although the application of EN50126 and EN50129 is wider than just E/E/PES. EN50129 requires that '*before an application for Safety approval can be considered, an independent safety assessment of the system/subsystem/equipment and its Safety Case shall be carried out, to provide additional assurance that the necessary level of safety has been achieved*'. This states a clear objective for Independent Safety Assessment, but the standards give relatively little guidance on how this should be achieved.

In the UK these standards are supported with the 'best-practice' guidance contained in the Yellow Book (2000) and its associated Application Notes. Yellow Book Application Note 4 (2005) addresses Independent Safety Assessment. While the Yellow Book originates in the UK, it is also used as guidance on good practice in a number of other countries and organisations.

In the UK main-line railway, the support of ISAs for safety submissions has been a mandatory part of the approvals process for many years. Approaches to the use of ISAs in other countries, and on other railways, varies. Increasing use of EN50129 means that independent checking of some sort is becoming more common on projects, but there is still a lot of variation in the scope assigned to ISAs, the depth of assessment expected of ISAs, the methods used by ISAs, and the relationship between ISAs and safety approvals bodies.

It should also be noted that the normal practice for employing ISAs, at least where the use of a specialist ISA organisation is concerned, is for the organisation that is submitting the safety case to contract, manage and pay for the ISA.

Rail safety approvals in Europe are currently undergoing a shake-up with the introduction of the High Speed Rail Interoperability directive (EU Council Directives 1996/48/EC and 2004/50/EC) and the Conventional Rail Interoperability directive (EU Council Directives 2001/16/EC and 2004/50/EC). As these are enacted in law in member states they are resulting in new structures for safety approvals, in which the role of appointed Notified Body (NoBo) assessment organisations is much greater. The topic of NoBo is not discussed further in this paper, but note that independent safety assessment still has a role in this, as required by EN50129, but in some cases is becoming more closely linked to the work of the NoBo.

What is important to understand in reading this paper is that there is no single well-defined ISA approach. ISAs are widely used in supporting safety approvals in the rail industry, but there is considerable variety in how this is approached and what is expected of the ISA by the parties involved in developing and approving systems.

Finally, it is worth stating that Independent Safety Assessment is not just the preserve of the rail industry. The underlying principle of independent expert check of safety work is recognised as good practice in many safety-related industries, including nuclear, automotive and defence, and is rapidly becoming recognised as necessary across the safety domain.

4 The Benefits of Legacy Systems

To minimise programme and cost risk, projects understandably want to re-use existing off-the-shelf (OTS) solutions. OTS systems are generally better understood than bespoke ones, their operation and maintenance is understood and their limitations determined. There is, on the surface, reduced development risk and reduced certification risk if the OTS system has already been through some form of regulation. Indeed, many forms of regulation explicitly allow for cross acceptance, i.e. allowing a developer to take credit for accreditation already gained. Cross-acceptance of a proven in-use (legacy) system should mitigate against new system development risks. Projects can often therefore consider that using a proven in-use system will enable tight delivery timescales to be delivered, thus meeting financial constraints on a project.

Whilst proven systems can fit easily into the target safety case, the legacy system may not be CENELEC compliant. Back-fitting CENELEC compliance may not be possible since details of original requirements, design regime and safety justification may be sketchy.

It is also worth considering that 'legacy systems' actually cover a wide variety of scopes from, for example, a whole interlocking or network application, through specific equipment such as relays, to something as small as an individual module of software. However, it is likely that it will be used for the first time on the intended railway system or perhaps in a changed context with a different interface arrangement. Such change of use needs to be considered in the cross-acceptance analysis; we explore this issue in more depth in the first of our assessment challenges.

5 Assessment Challenges

5.1 Cross-Acceptance

The basis of cross-acceptance is to justify the safe re-use of the legacy system or equipment. A major challenge in achieving this is to define the baseline system that forms the legacy – the baseline is a specific configuration. Understanding the baseline system and component subsystems is the basis on which the cross-acceptance argument is built. From the baseline system the degree of legacy claim can be assessed. However, beware that the operational baseline system may have had upgrades undertaken during its service life, thus assessors need to be careful that the amount of in-service history that is being presented for a particular sub-system may not relate to the very latest version of that sub-system currently in use. Legacy reliability data will often need to be relied upon and within this evidence it needs to be clear what system configuration was used to gain it and the extent and date of modification from when the reliability is claimed.

Having defined the original baseline (or native) system as the legacy system, experience shows that manufacturers will take a new application as an opportunity to alter product elements due to equipment and component updates, possibly due to obsolescence of components. If, say, a microprocessor is changed, this may lead to wholesale software language changes. Safety for these changes will need to be argued in addition to the cross-acceptance.

In addition to the sub-system or product level changes there will inevitably be functional changes at the Generic (or Specific) application level. These too have to be defined.

Overall system confidence is built up from the legacy of the sub-systems, the understanding of changes at the product or sub-system level and the combination of these for the generic application. It is common for consideration to be given to historical applications worldwide; for example the legacy argument used for the signalling system for a major re-signalling project being carried out in the UK with which we are familiar is based on an application baseline from an installation in the far East, although there is additional historical confidence based upon a North American and UK metro application.

A number of key elements are required to satisfy an independent safety assessor of a cross acceptance argument. Is the system valid? Is the system being used in the same way? Are there any safety issues open?

Although the concept is to target a generic system, when cross-accepting to another railway organisation it is not straightforward. The physical environment, interface environment, interlocking rules and operational rules are all likely to be different. It could be questioned whether there is such a thing as a generic system.

In the following paragraphs we discuss a number of the obstacles to safe reuse.

The assurance regime of the baseline system

Confidence is required of the product acceptance regime in the native environment and its rigour and depth of original assurance.

- The system supplier will wish to argue that the system has a strong proven in-use record. The assurance regime that was in place for the legacy system is likely to be 'owned' by another railway organisation. Thus there may be a lack of visibility as to the scope and rigour of the legacy assurance process.
- From the infrastructure perspective, full assurance in detail will need to be supported by the manufacturer. Thus where evidence is not forthcoming additional target regime assurance will be required to demonstrate rigour. In addition there is likely to be assurance overlay to bring the confidence 'up to' the target railway environment assurance.
- Making an approach to the native environment railway operator to question the safety of the signalling system may not be greeted with enthusiasm. However, understanding the breadth and depth of the assurance regime is essential and building a relationship with the native railway operator is necessary.

Safety argument for the baseline system

Assessment is required of the safety argument for the baseline system with adaptation for the target railway. Clarity on compliance to standards and any derogations also needs to be considered.

- The manufacturer will want to reuse an existing safety case. There is likely to be resistance to adapt the system to the target railway because of the need to revisit the safety case.
- From the infrastructure perspective any adaptation will need to be undertaken to new technical standards and the need for the system to be supported long term could force upgraded components onto the systems.
- To meet the target railway environment the target engineering standards need to be met. To determine the difference between the native and target standards may be a complex process and thus compliance to the target standards may be needed from first principles.

Context of use

Full understanding of the application in the target environment compared to the native environment is required. Experience shows that application rules (for example interlocking rule differences or different braking characteristics due to train type) need to be addressed.

- The Railway context is likely to be different. So from the suppliers' perspective it is likely that changes to design rules for items of application data are required.
- From the infrastructure perspective, new equipment brings new or changed functionality. This may require, for example, change of data type and/or format, hardware/software updates to ensure long-term support of the system and changes to communications protocols and security updates. Inevitably there will also be a change in electromagnetic compatibility and environmental conditions.
- Defining the full context of system changes of hardware, software and application data is necessary and the impact upon the target environment needs to be understood. Changes due to interfacing equipment (e.g. trackside systems, control centre interfaces, adjacent interlockings and communications connections) are inevitable and to manage them it is necessary to ensure knowledge of the interfacing equipment.

Application rules

Usually every railway system brings with it a set of operational rules and regulations. A full understanding of the operation in the target environment compared to the native environment is required.

- Changes due to railway operating rules must be considered; i.e. the safe use of the system may be consistent with the operating railway but it may not necessarily support the operating railway. The manufacturer will want to minimise deviations from previous application rules to limit impact on re-engineering the system and consequential impact on the safety case.
- The infrastructure owner will want to undertake minimum change to existing practices which will have been built up historically and will be part of the operating staff's 'natural' behaviour.

- The type of railway is a dominant factor on the degree of change in this context. A captive railway, for example a metro system operated by one set of rolling stock and staff dedicated to that railway, has a greater chance of shift of operating regime than a line connected to a national rail network operated by a number of train companies or types of rolling stock.
- Even where revised signalling technology has minimum impact on the user interfaces it is very likely that there will still be a need for a revised set of operating rules to meet the requirements of the existing operating philosophy, for example the need to operate in degraded modes for both the signalling system and train systems.

Maintaining system safety

To ensure that system safety is maintained after commissioning, the use, limitations and constraints for the target system need to be clear. Assumptions, dependencies and caveats for the native environment need to be understood so as to safely pass to the target environment.

- The manufacturer is required to hand over a set of maintenance requirements with the system. This is likely to be the maintenance requirements from the previous application and may not be the optimum for the target railway environment.
- The infrastructure owner will want to reduce overall maintenance compared to the 'previous system' and ensure that the environmental constraints are met.
- Understanding the application conditions and reasons for particular maintenance regimes in the native environment and implications of any adaptation for the target system is critical to the long term safety of the target railway. Applicability of legacy performance history and DRACAS (Data Recording, Analysis and Corrective Action System) information to the target performance criteria need to be considered.

5.2 Operational Evidence

There is often a significant operational change to the railway when new equipment is introduced. Taking the installation of a new computer-based interlocking system onto a railway as an example, the solution is nearly always a mixture of differences in the types of equipment change and differences in the amount of operational change required to the target railway to limit redesign of the equipment.

Put another way, are changes to be made to the railway or should we make changes to the legacy system? Either solution can be used but more realistically it is usually a combination of both. A specific example has recently been experienced in one of our projects.

In our example the OTS, interlocking will not work with the latest version of the train detection axle counter system. The problem is whether we should change the railway, for example use an older model of axle counter rather than a more modern equivalent (however only the up to date version is likely to be fully supported by the manufacturer) or change the legacy system, for example by making technical changes to the interlocking commands that interface to the axle counter system.

The two solutions to this type of problem are:

- Changes to the re-used hardware/software component are usually possible. However the component safety case will need to be revisited. Increasing the system integrity may be difficult and there are inherent risks in modifying existing equipment (the proven in-use argument can become reduced).
- Changes to the railway can be made but in most cases there may be no existing safety argument to change. Establishing a safety argument for the whole railway could have numerous knock-on affects. In addition there are risks associated in changing established practices.

In making the trade-off the temptation is often to change the procedures of the railway, but this may not always be the most robust long-term solution. However, making adaptations to the new system is likely to further impact the cross-acceptance argument.

It needs to be recognised that with an acceptance of technology driven changes there is very likely to be an impact upon degraded modes and railway operating rules.

It is important not to underestimate these problems. They must be thought about early when undertaking the safety planning.

5.3 ISA Hierarchy

The safety evidence for a project of any complexity is frequently made up of a number of separate safety cases in a hierarchy. Safety cases for Generic Products may feed into higher-level safety cases for a Generic Application, which in turn may feed into larger integrated Generic Applications or into Specific Application safety cases. These different types of safety case are defined in EN50129 (2003). The safety case hierarchy may have quite a number of levels, and involve a number of different equipment suppliers, contractors, railway owners and operators. Where the safety evidence for a total system is built up in this way there is likely to be a number of different ISAs involved in assessing the different safety cases. A simple example of such a structure is shown in Figure 2. Furthermore, where parts of the hierarchy are carried over from previous projects or systems there will be a desire to reuse the associated ISA assessments. This section discusses challenges associated with managing the relationships between the ISAs in the hierarchy, and the possible reuse of ISA assessments.

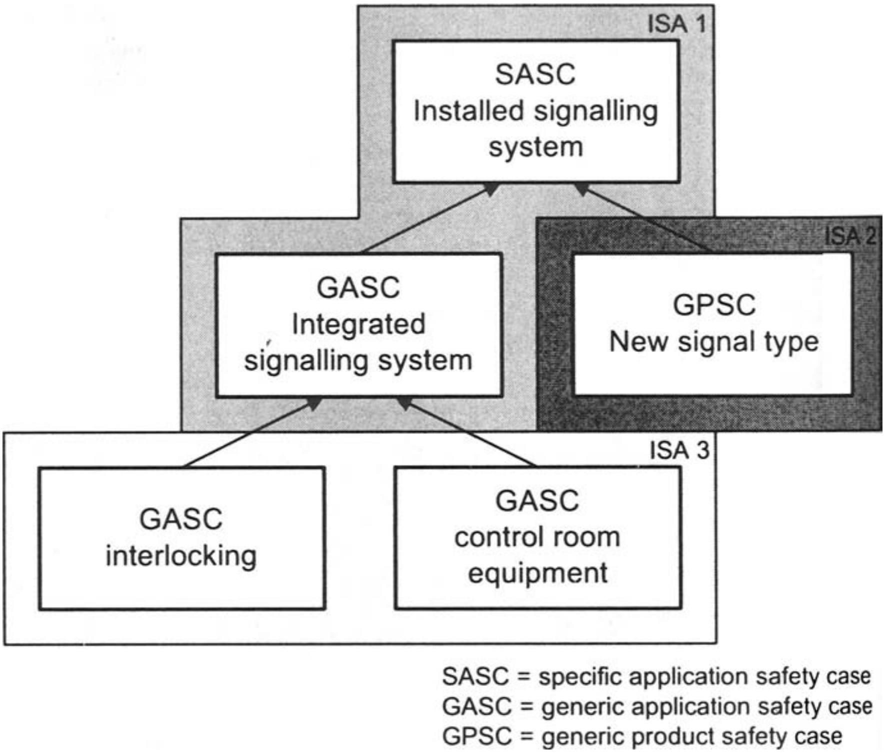

ISA 1
SASC
Installed signalling system
ISA 2
GASC
Integrated signalling system
GPSC
New signal type
ISA 3
GASC
interlocking
GASC
control room equipment
SASC = specific application safety case
GASC = generic application safety case
GPSC = generic product safety case

Figure 2: Example of ISA coverage of a safety case hierarchy

Typically the ISA for one safety case will need to know something about the assessments done by the ISAs of lower-level safety cases. In this section we will use the terms 'the ISA', 'project' and 'safety case' to refer to the ISA, the safety case authors and the safety case itself at a particular level of interest in the hierarchy, and the terms 'sub-ISA', 'sub-project' and 'sub-safety-case' to refer to the activities at the immediately subordinate level.

The safety cases in the hierarchy have relationships with each other, and safety-related information is exchanged and managed across the boundary. For example, safety requirements are passed down, safety-related application conditions (SRACs) are passed up, responsibility for managing hazards may be passed either way. The ISAs have to ensure that not only are the individual safety cases correct and complete, but also that the safety-interface between the two projects has been managed.

There are two types of interaction between ISAs that may be necessary:

- *Sharing of assessment information.* This consists of discussion and liaison between ISAs to address the safety of the interface between projects. How much effort needs to be put into this will depend on the perceived level of risk associated with the safety interface. If the project interfaces are complex, hard to define, or in some way novel then there is a greater risk of safety issues being missed at the interface, and the ISAs will need to ensure that these risks are addressed.

- *'Accepting' the work of sub-ISAs.* The ISA may need to form an opinion on the adequacy of the assessment of the sub-ISA. The extent to which this is necessary will depend on the way in which submission and acceptance of the safety cases in the hierarchy are managed. If formal acceptance is based solely on the safety case at the top of the hierarchy, then the ISA for that safety case will need to be able to confirm that all supporting safety cases have been adequately assessed and supported. If each safety case in the hierarchy is being submitted for approval separately then there may be less onus on ISAs to check the work of sub-ISAs. The expectations of the project and the approvals authority should, ideally, be stated in the remit given to the ISA.

There is also a third case which needs to be considered, where a legacy sub-system is being used and the sub-ISA's assessment is already complete and documented.

- *'Adoption' of the work of sub-ISAs.* In this case the ISA may have to form an opinion on the assessment of the sub-system, based solely on the sub-ISA's documentation of its earlier work.

If an ISA needs to form an opinion about the work of a sub-ISA, either to check that the safety of an interface has been addressed, or to 'accept' or 'adopt' the sub-ISA's work in some way, then the following issues may need to be considered (among others).

- Does the scope of the sub-ISA's work match the sub-safety case, and fit with the ISA's own scope? This includes careful consideration of the interface between the projects, to ensure that all elements of managing the interface have been assessed by one ISA or the other. This can also help to avoid duplicated work.
- Is the sub-ISA suitably competent for the scope of their assessment, and are they independent from the sub-project?
- Has the sub-ISA assessed to a sufficient depth to support the claims they make in their final conclusions, and have they covered the full scope?
- Has the sub-ISA given their opinion against the application for which the sub-safety case is now being used (e.g. trial running, full revenue service)?
- Has the sub-ISA identified the key safety risks, and concluded that they have been adequately managed?

Where liaison or checking between ISAs needs to take place, duplication of work should always be avoided. It should not normally be necessary for one ISA to repeat work carried out by another ISA. The following list gives some practical methods that are useful:

- Discussion and liaison. The authors recommend, as a minimum, that ISAs should meet to discuss their scopes, clarify their boundaries, and to discuss any technical concerns that may arise from the assessments. As well as helping to give a clear division of responsibilities these discussions can also help to reveal 'whole-system' safety risks, and can help the projects in defining their own boundaries.

- Read the sub-ISA's final report. The sub-ISA's report is their key output. The ISA should be able to determine from this report the scope and depth of the assessment, the competency of the assessors, any assumptions or restrictions on the use of the sub-system and the conclusions of the sub-ISA.
- Read sub-ISA's ISA plan. To identify any concerns about the sub-ISA's work early on, the ISA may wish to read the sub-ISA's plan, which should give a clear view of their intended scope and how they will carry out the assessments.
- More formal audit-style review. Where there is insufficient documented evidence of the sub-ISA's assessment to allow the ISA to accept it, then a more formal audit-style discussion with the sub-ISA may be necessary to build up sufficient confidence in their conclusions.

In the experience of the authors, these are the most helpful methods for accepting the work of a sub-ISA with the minimum of intrusion. However, there are a number of challenges which can arise in managing the relationship between ISAs, and the problems in the following list have all been encountered by the authors:

- Assessors who 'know' the system well may not have documented their opinions rigorously. Assessments may have been based on expert judgement without building a documented and defensible justification.
- Issues may have been agreed with the project through discussions as being closed, without formally documenting the reasons for closure, or minuting the meetings.
- ISAs may have different interpretations of what is acceptable evidence to achieve compliance with standards, especially where ISAs are from different cultural backgrounds.
- ISAs may object to being questioned or scrutinised, and consider it to be questioning their professionalism or expertise.
- ISA organisations may not be willing to issue Plans and Reports to other ISAs. ISAs are businesses in competition with each other, and may have developed proprietary ways of marketing and performing assessments.
- The sub-ISA of a legacy system may no longer be available for discussion.

There are no easy answers to most of these obstacles if they genuinely arise. However, the best way to minimise the risk of obstacles, and to maximise the benefit that can be gained from the ISAs across the hierarchy, is to involve the ISAs as early as possible in discussions with each other and with the project about the scope of the individual safety cases, the objectives for the overall project, and the role which each ISA has to play in the overall submission and safety acceptance of the system. This has been shown to work in practice, especially where ISAs are working concurrently on different parts of the system hierarchy.

The other key lesson, which applies to managing ISA interfaces, 'accepting' ISA assessments and 'adopting' legacy ISA assessments, is for ISAs to ensure they produce clear documentation. ISAs need to document clearly their scope, the methods and people used, the activities carried out, the evidence assessed or audited, the results of the assessments, the conclusions of the assessment, the reasons for these conclusions, any constraints or assumptions on the validity of the conclusions, and any outstanding open issues.

5.4 Assessment Competence

Introducing the correct competence into an assessment is obviously important – and this matters for both the process competence (e.g. human factors, safety engineering, etc.) and the product competence (e.g. the specific railway signalling scheme under assessment). Indeed it is reasonably well accepted now that assessments should often be carried out in a team manner, with a Lead ISA supported by a number of assessors who, in combination with the lead, provide sufficient coverage of the technical areas.

The concept of 'team assessment' is particularly necessary for large-scale systems, both because of the issue of scale and also because of the multi-disciplinary nature of the project. The additional challenge for large-scale legacy systems can be in gaining access to competency that could have become very scare (perhaps due to obsolescence) or due to it only belonging in other (perhaps non-local) organisations.

So, does this mean that this class of system is likely to require a large breadth of potentially very scarce competence? Well, our answer is 'no'. The issue is to match the competence with the areas of risk that you are most worried about.

We have found that a key tool in determining the competence you require to assess, is to make use of early multi-perspective risk identification workshops. These workshops can be used to both guide the assessment direction and to determine which areas of the assessment are likely to require particular competences. It is never possible, or cost-effective, to assess everything – and so this should also include the issue of which competence to bring to the assessment.

5.5 Observation Management

ISAs often work through the use of 'tracked to closure' observations. Documents are assessed by the ISA and then observations raised and shared with the Project for discussion and closure. Observations are often prioritised and must be addressed by the Project to demonstrate that the ISA's concerns have been recognised. The history of an observation's life is often maintained in order to demonstrate to third parties that the ISA/Project relationship is operating correctly.

With large-scale projects there are risks associated with the management of observations themselves. Specifically there are issues attached to:

- Volume
- Freshness, and
- Relevance

On large projects, there can be a temptation to review a lot of documentation. When a lot of documentation is reviewed, there can be a temptation to generate a lot of observations. Tracking, closure, and general management of observations can become a significant activity for both Project and ISA. Indeed, the management of large numbers of observations can become a drain on Project resources and begin to mask the underlying safety concerns. With large projects it is particularly important to be careful about the numbers of observations raised and to remember that the job of an ISA is to provide an opinion on the manner in which risk is being addressed, and not to be a manager of observations.

On a large-scale project, it is not only the number of observations which must be considered but also the freshness. Observations have a useful 'life' during which their closure can be of added value to the project. However, over time the validity of an observation can change; it can become stale because of other events having moved on or because of a change in the understanding of the assessors. It is important that the freshness of an observation is monitored, and if there is delay in its resolution consideration be given to abandonment if this is a safe option as well as to escalation.

The final issue concerning observations relates to their relevance. The technical understanding of an ISA and its Project mature at different rates and this can effect the relevance of observations that the ISA can raise. The key concerns relate to when the technical maturity of a Project is greater than that of an ISA or when they are both equally immature.

- If a Project is more mature than an ISA, then observations raised by the ISA will often be seen as irrelevant and of little value; here the relationship can degrade and the value of the ISA can be reduced.
- If the Project and the ISA are equally immature, then wasted effort can be spent identifying irrelevant issues and then closing them out satisfactorily.

Where either of the above is true, consideration should be given to the type of assessment (and amount of it) whilst technical understanding of the relevant parties matures.

The best case is where the ISA is more technically mature than the Project; in this case observations raised can be of extremely high value to the Project because the ISA has 'been there before'.

6 Lessons

The authors draw out key lessons for the assessment of large-scale legacy safety-related systems.

L1. Cross-acceptance is rarely straightforward for non-simple items. Consideration to the commercial risks should be given and detailed assessment of the argument undertaken. The use of legacy equipment is rarely 'plug and play'!

L2. Operational evidence is rarely what it seems. The evidence must be inspected 'in the large' to ensure that matters of configuration, open safety issues, railway environment and SRACs are considered.

L3. The exact configuration of the baseline system requires accurate examination to understand how best to apply previously accepted safety evidence.

L4. Maintaining long-term safety constraints in an operating environment is not as simple as it sounds.

L5. The use of a legacy system will nearly always result in both equipment and railway environment changes.

L6. Where there is a hierarchy of safety cases, ISA coordination needs careful consideration. It can be beneficial to involve ISAs as early as possible in discussions with each other and with the project about the scope of the individual safety cases, the objectives for the overall project, and the role which each ISA has to play in the overall submission and safety acceptance of the system.

L7. ISAs need to produce clear documentation, and projects need to provide sufficient budget to facilitate this. ISAs need to document clearly their scope, the methods and people used, the activities carried out, the evidence assessed or audited, the results of the assessments, the conclusions of the assessment, the reasons for these conclusions, any constraints or assumptions on the validity of the conclusions, and any outstanding open issues.

L8. ISA competence needs to reflect the key areas of risk; a sampling approach can be successfully applied such that the relevant (legacy) competencies are identified and not necessarily all of those represented by the Project.

L9. ISAs and Projects mature at different rates. If either the Project or the ISA is too immature, little formal assessment should be carried out until both parties progress.

7 Conclusions

In this paper we have tried to distil a number of the lessons we have learnt from assessing large-scale complex safety-related systems that have included a significant legacy equipment strategy. The use of such equipment is often seen as attractive because it offers the potential to offset some flexibility against known developmental risks. However the strategy raises a particular set of issues from both a project's perspective and from an assessor's perspective as reuse of equipment is not as simple as first it may seem. The lessons we have presented are drawn from our experiences and whilst largely from the rail domain, we do believe they are relevant to other sectors as well.

8 References

EN 50126 (1999) *Railway Applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)*, CENELEC

EN 50128 (2001) *Railway applications – communication, signalling and processing systems – Software for railway control and protection systems*, CENELEC

EN 50129 (2003) *Railway applications – communication, signalling and processing systems – Safety-Related Electronic Systems for signalling*, CENELEC

IEC 61508 (1998, 2000) *Functional safety of electrical/ electronic/ programmable electronic safety-related systems, issued versions*

Yellow Book (2000) *Engineering Safety Management*, issue 3, RSSB

Yellow Book Application Note 4 (2005) *Engineering Safety Management (Yellow Book 4) Application Note 4, Independent Safety Assessment*, Issue 2.0, RSSB

EU Council Directive 1996/48/EC for the *interoperability of the trans-European high-speed rail system*

EU Council Directive 2001/16/EC for the *interoperability of the trans-European conventional rail system*

EU Council Directive 2004/50/EC – amendment to 1996/48/EC and 2001/16/EC

Safety Assessments of Air Traffic Systems

Rodney May BSc PhD CEng FIET
rodmayAssociates
Glentworth, Lincolnshire, England
rodmay@iee.org

Abstract

This paper is essentially a case study of a safety assessment of an air traffic system. Key issues concerning safety management system essentials; safety requirements derivation and safety assurance provision are discussed. The study is based on many safety assessments of air traffic systems recently undertaken for medium-sized UK airports. The impact of the EU directive on interoperability is also reviewed.

1 Introduction

This paper is a case study of a safety assessment of an air traffic system, and covers three main topics: safety management system essentials; safety requirements; and safety assurance.

No attempt has been made to give a treatise on each topic, for which standards and guidance already exist, for example, guidance for Air Traffic Services, [Ref. 1, 2, 3 & 4]. Instead, a small number of key issues are addressed.

Confidentiality precludes discussion of a specific airport operator and assessment. Instead, recent experience gained on several assessments of air traffic systems for medium-sized airports, regulated by the UK CAA Air Traffic Services Safety Regulation Group (UK CAA ATS SRG), is used. The experience of air traffic system assessments includes:

i) radar processing and display system;
ii) meteorological system;
iii) data recording system;
iv) primary surveillance radar sensor;
v) reduction in separation standard from 5 to 3 nautical miles; and
vi) airport 500 KVA uninterruptible power supply.

In all cases a similar assessment and reporting approach was adopted.

By its very nature, air traffic control is concerned with safety. The provision of air traffic services in the UK is well regulated by the UK CAA, and indirectly (through the CAA) by EUROCONTROL and ICAO (International Civil Aviation Organisation). The issues that are addressed are as a result of achieving compliance with the current regulatory environment. As such, a preliminary view is also given on the impact of the EU directive on the interoperability of the European Air Traffic Management network.

2 Safety Management System Essentials

Two fundamental Safety Management System essentials for safety assessments are a method for classifying risk, and hence determining safety requirements, and a method of reporting the assessment results. Different industry sectors vary in their approach to meeting these two requirements, depending on their specific circumstances See for example, Part 5 of IEC 61508 [Ref. 7]. For air traffic systems in Europe, a risk matrix of severity and likelihood populated with tolerable risk is usual. In the UK the CAA SRG expects a safety assessment to be reported as a safety case.

For all the assessments listed in the introduction it has been necessary to produce safety management procedures for safety risk classification and safety cases in addition to undertaking the actual assessments themselves.

2.1 Safety Risk Classification

Most safety professionals are familiar with the concept of a safety risk-classification matrix. However, specifying a matrix for a particular air traffic environment raises two issues:

i) What hazard should be considered against which severities of risk are defined? and
ii) How should the matrix be populated for the air traffic environment?

Air traffic systems can be classified as two types: systems that can be directly interpreted by pilots; and systems that are interpreted by controllers to provide an air traffic control (ATC) service.

The purpose of ATC-interpreted systems is to maintain separation minima either between aeroplanes or between an aeroplane and a fixed object (for example, terrain). If the separation minima are infringed, then depending on circumstances, there is a chance that an accident could occur, leading to the likelihood of multiple deaths. Thus the harm could be injury or death to one or more people leading to a severity classification of multiple deaths, single death, etc.

However, a consideration of all the possible circumstances following infringement of separation minima which could result in an accident is very difficult. If attempted, it is likely to involve substantial modelling and/or computer simulation of a range of possible scenarios, making such a scheme very onerous and, in practice, unworkable. Instead, a more pragmatic approach is to define severities in terms of the ability to provide air traffic control, i.e. to maintain the specified separation minima. It side-steps the need to consider the consequences following an infringement of separation minima, as this is implicitly addressed when first constructing the risk classification matrix. In essence it is re-defining the hazard at some point in an accident sequence short of the actual point that harm occurs.

The second issue is how the risk classification matrix should be populated for a given air traffic control situation. This again presents a significant problem. In theory, for a specific aerodrome, the basis for populating a matrix could be historical records and/or modelling and computer simulation. In practice, neither has proved feasible for a medium-sized airport. Instead, a pragmatic approach is adopted using a modified form of a scheme which is in the public domain and is judged, by the regulator, to be safe for the largest aerodromes in the UK.

The rationale is that if it is safe for the largest UK aerodrome it is more than safe for a medium-sized aerodrome, unless exceptional local circumstances exist. The problem is that such a scheme is likely to be "over-safe", assuming such a concept is valid. This is addressed by not lowering the safety standards, but by introducing increased granularity in the definition of severities.

Table 1 is a complete risk matrix for ATC-interpreted systems. The likelihood or probability of failure is defined in units appropriate to the hazardous situation, that is, failures per operational hour per ATC position. Likelihood is defined both in qualitative and quantitative terms. There are approximately 10^4 hours in a year so "Frequent" is of the order of once a month.

Likelihood or probability of failure per operational hour per operational position		Severity					
Qualitative [Note 2]	Quantitative	1	2a	2b	3a	3b	4
Frequent	$> 10^{-3}$	A	A	A	A	B	C
Probable	10^{-3} to 10^{-4}	A	A	A	B	C	D
Occasional	10^{-4} to 10^{-5}	A	A	B	C	D	D
Remote	10^{-5} to 10^{-6}	A	B	C	D	D	D
Improbable	10^{-6} to 10^{-7}	B	C	D	D	D	D
Extremely improbable	$< 10^{-7}$	C	D	D	D	D	D

Table 1 - Risk classification matrix for ATC-interpreted systems

With reference to the risk classification matrix in Table 1, the increased granularity in severity by sub-dividing severities 2 and 3 can result in less onerous safety requirements according to the ALARP principle, without significantly compromising the safety required.

This is illustrated in Table 2, which is an extract from a table of severities from 1 to 4, together with "No effect".

Severity	Definition	Rationale	
		Effect on ATCO in providing a service.	Work around
1	Sudden inability to provide any degree of air traffic control or information to pilots (including contingency separation measures) for a significant period of time.	Likely to be desperate.	There is no planned or unplanned "work around" that can be implemented.
2a	The ability to maintain air traffic control or provide information to pilots is severely compromised without warning for a significant period of time.	Possibly frightening.	There is no planned "work around".
2b		Likely to be difficult.	It is likely that "work around" exists and can be implemented.
No effect.	The analysis may show that some failures have no safety significance and these shall be categorised accordingly.	None.	Not applicable

Table 2 - Definition of severity categories

Three aspects are illustrated:

i) The definition of severity is related to the hazard of not being able to maintain a separation standard.

ii) Severity 2 has been divided into two, severity 2a and 2b through the inclusion of a rationale.

iii) The rationale provides guidance to enable a user to make sensible judgements.

The population of the matrix is aligned with the ALARP (As Low as is Reasonably Practicable) principle, where risk is classified from A to D:

A	Unacceptable
B	Tolerable only if risk reduction is impracticable or if its cost is grossly disproportionate to the improvements gained.
C	Tolerable if cost of reduction would exceed the improvement gained.
D	Acceptable, necessary to maintain assurance that risk remains at this level

Table 1 – Application of the ALARP principle

Class C and B are in the ALARP region. A judgement is necessary on the cost of reducing the risk against the risk accepted. Effectively, this is a cost/benefit analysis. In practice increasing seniority is expected in the sign-off of risk as acceptable when moving from risk class C to B.

2.2 System Safety Cases

A System Safety Case can be viewed as documenting system safety assurance, where assurance is a claim, supported by an argument, substantiated by evidence.

In the UK it is usual to document safety assurance by reporting safety assessments and associated information in a safety case. The air traffic sector is no exception. However, differences in the format and content of safety cases exist between industrial sectors. For example, the MoD typically adopts a one-part safety report, progressively released during a project life-cycle, referencing further information (typically evidence) in support of a safety argument, the whole being called a safety case. The National Air Traffic Services (NATS) typically use a four-part structure, released at key project milestones.

It is proposed that there is a strong theoretical and practical argument for a two-part safety case approach, supported by document templates with embedded guidance on for example how a safety argument should be constructed.

Excessive multiple parts has the potential to lead to considerable duplication of material in each part. Adopting a one-part structure has the potential to not enforce a key discipline, separating the specification of safety requirements from the provision of safety assurance. For these reasons a safety structure comprising two parts; Part1 (Safety Requirements) and Part 2 (Safety Realisation) is judged best. Such a structure aligns and enforces the discipline of using risk assessment both to determine safety requirements and to provide safety assurance.

In practice, another strong practical argument for the use of document templates in association with the two-part structure, provided the templates includes embedded guidance when innovative input is required. Thus the templates comprise both standard text and guidance to be overwritten by project-specific information. This is acknowledged as a "double-edge sword". The downside is that such a pre-determined approach might result in some new unforeseen aspect being missed.

The benefits are:

i) a good document template will increase significantly the likelihood that all legitimate issues needing to be addressed will be addressed;
ii) it is more cost-effective as time is not wasted in re-inventing a document structure and its contents on each occasion that a safety case is required;
iii) it results in considerable simplification of the procedure for system safety case; and, most importantly;
iv) it demystifies the process and facilitates the involvement of local experts who know the application in depth and the hazards that can arise; yet
v) still facilitates (through overwriting embedded guidance) the development of a strong safety argument.

As usual, there are always exceptions. With reference to Section 3 below, a safety case for the reduction in separation standard from 5 to 3 nautical miles did not involve the systematic derivation of safety requirements with a tolerable hazard occurrence rate using the risk classification scheme. Instead, the requirements were stated as a regulatory requirement for the Primary Surveillance Radar Sensor [Ref. 6]. In this case, it was sensible to produce a combined Part 1 and Part 2 Safety Case.

3 Safety Requirements

Safety requirements can either be derived or are a given. The derivation of safety requirements involves a systematic process. Some derived safety requirements specify integrity, often in the form of a tolerable hazard occurrence rate. Other derived safety requirements could be functional or performance-related, for example a radar aerial might have to rotate once every 15 seconds.

In addition, in the UK air traffic sector there are usually (given) regulatory requirements to be met. These regulatory requirements often specify functional and performance requirements.

3.1 Requirements with a Tolerable Hazard Occurrence Rate

In the UK air traffic sector the accepted systematic approach for determining the derived safety integrity requirements is a Hazard and Operability (HAZOP) type study by a team, of a functional model of the system, with the results documented in the form of a Failure Mode Effect and Criticality Analysis (FMECA). This is illustrated in Figure 1. The figure shows a flow diagram of the steps required when performing such an analysis. The flow diagram is also supported by notes which give further guidance for some of the key steps.

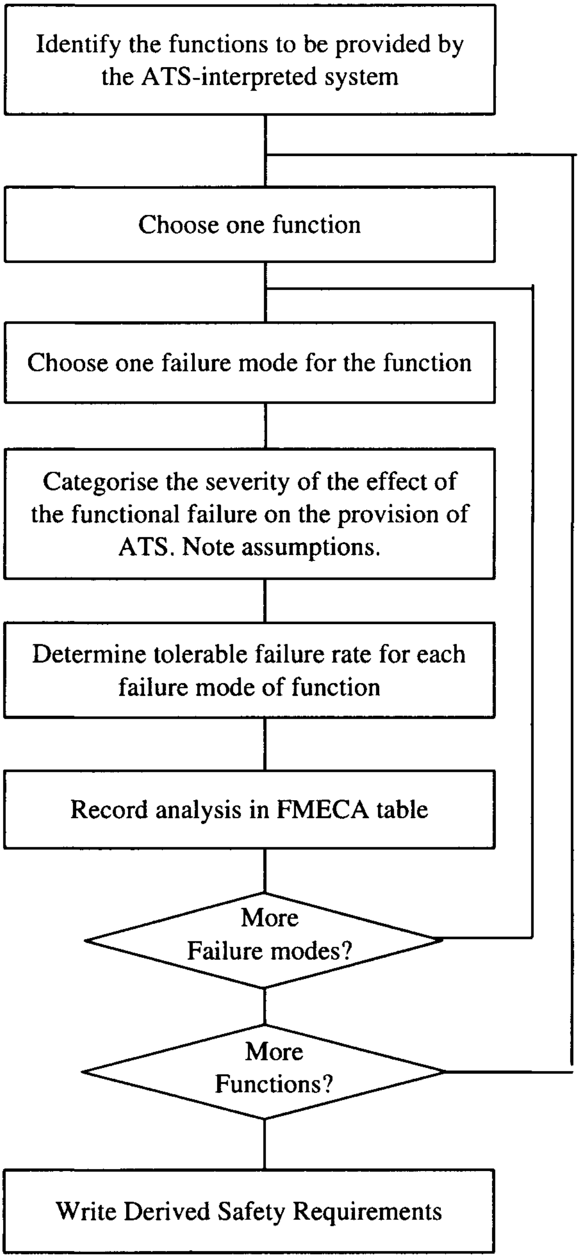

Identify the functions to be provided by the ATS-interpreted system
Choose one function
Choose one failure mode for the function
Categorise the severity of the effect of the functional failure on the provision of ATS. Note assumptions.
Determine tolerable failure rate for each failure mode of function
Record analysis in FMECA table
More Failure modes?
More Functions?
Write Derived Safety Requirements

Figure 1 - Derivation of safety requirements for ATS system

Notes:

An example of a function is a flight information system.

Examples of failure modes for a function are loss, partial loss, detected and undetected corruption

Use Table 2 to classify the severity of the particular failure mode of the function

Use Table 1 to determine the tolerable failure rate for the particular functional failure mode.

An FMECA table is included in the Requirement Safety Case MS Word template.

The basic structure of a derived safety requirement is “the probability that [*the failure mode of the function*] shall be no greater than [*the tolerable failure*

This is a standard approach familiar to functional-safety practitioners. From experience there are three key issues:

i) the choice of functional model;
ii) the choice of failure modes, including aggregate failure modes; and
iii) the importance of accurately capturing and documenting assumptions to be validated.

Ideally the functional model should be as simple as possible provided there is sufficient detail to derive a comprehensive set of safety requirements. There is a tendency to overcomplicate the functional model, either by considering low-level functional failures away from the system boundary (including treating causal effects as functional failures) and/or by including too much implementation detail.

Implementation details should be avoided. Exceptions regarding details on implementation could include planned redundancy or diversity which can significantly reduce the integrity required of a safety requirement. For example, in a control tower, provision for a back-up controller position might be part of an initial specification.

The choice of failure modes often presents a problem for inexperienced practitioners. Failure modes must be plausible and comprehensive. The most dangerous failure modes are usually undetected failures, for example, the probability of a radar display showing incorrect information which is not detected.

From experience, by considering high-level failure modes of a functional model at the system boundary the issue of aggregated risk, when more than one hazardous situation is plausible, does not present a significant problem. By specifying plausible aggregate failure modes the rates in Table 1 are still applicable.

Often the effect of a failure of a function is far less than the uninformed observer would expect. This is because mitigations already exist which reduce significantly the impact of a failure. In the case of a total failure of electrical power at one airport, this was heavily mitigated by a documented procedure to transfer control to an adjacent air traffic control provider. Clearly, as a legitimate regulatory process, it would be negligent not to include it in the requirements determination process. Equally clearly, it was an assumption upon which the safety requirement relied, and if it was not documented, the specification of the safety requirement would be incomplete.

3.2 Functional, Performance & Regulatory Safety Requirements

The functional and performance requirements have been grouped with regulatory requirements because in the UK air traffic sector, experience shows they are often closely coupled.

The UK CAA SRG has a standard for air traffic services safety requirements, CAP670 [Ref. 6]. For example, there are nine safety requirements for radar.

RAD01 to RAD09. RAD05 specifies "Radar Display Engineering Requirements". Topics covered include; display characteristics, symbology, engineering design and functional parameters. An examination of the requirements indicates that the functional and performance requirements are often already comprehensively addressed.

Another topic addressed under Systems Engineering is SW01 "Regulatory Objectives for Software Safety Assurance in ATS Equipment". This effectively gives guidance on how the attributes required for software should be assured. As most ATC systems contain software, this is an important regulatory safety requirement to state and meet.

It should be noted that the above comments are based on the experience of the case studies within the UK environment. Usually for ATC-interpreted systems, all the functional and performance requirements are not addressed by a regulatory safety standard. In all cases a rigorous structured approach should be adopted to ensure that all necessary safety requirements are captured. The derivation of functional and performance safety requirements is often the most difficult aspect of an air traffic management safety assessment and, in the past, has often been avoided, by requiring new systems to have at least the same functionality and performance as the systems they are replacing .

3.3 The Importance of Assumptions

The importance of assumptions has already been mentioned in Section 3.1. If too much mitigation is assumed, the safety requirements will be deficient and hence unsafe. If too little mitigation is assumed, resources will be inefficiently deployed when they could be more effectively employed elsewhere.

The question is often asked "When should mitigation be accounted for, during or after the derivation of safety requirements?" In the air traffic sector mitigation is often a regulatory requirement, for example, the use of procedural control. In such situations it is most sensible to account for such mitigation during the requirements derivation. As such, the derived requirements are likely to be considerably less onerous, resulting in fewer burdens on the system realisation process.

Mitigations are assumptions that have to be validated. If a procedure is assumed that transfers control to another air traffic control provider in the event of a total system failure, it must be "fit for purpose" if and when required. Clearly, if the validation of assumptions is not kept current the safety requirements will not be met and a dangerous situation could arise.

Summarising, including mitigation in the requirements determination process, effectively equates their safety significance with safety requirements. As such, it is very important that both safety requirements and assumptions are treated the same as regards the provision of assurance during the realisation process.

3.4 Packaging of a Requirement Safety Case

The template with guidance for a Part 1 Requirement Safety Case is quite simple:

1 Introduction
   1.1 Purpose of System
   1.2 Functional Description of System
   1.3 Scope of System
2 Safety Requirements Specification
   2.1 Derived Safety Requirements
   2.2 Functional, Performance and Regulatory Requirements
   2.3 Assumptions
Definitions
References
Appendix A – FMECA Table
Appendix B – Derived Safety Requirements
Appendix C - Functional, Performance and Regulatory Requirements
Appendix D - Assumptions

Table 3 – Contents of Requirement Safety Case

The template should have the attributes expected of an ISO 9001 compliant document.

A personal preference exists to include the detail where possible in appendices and to keep the main body of the document as "standard" as possible. This explains the apparent duplication of Section 2.1 and Appendix B, Section 2.2 and Appendix C and Section 2.3 and Appendix D. The material in the main sections 2.1 to 2.3 describes the process followed whilst the associated appendices contain the actual results.

Using this template approach the main activities when drafting a Requirement Safety Case are the completion of Section 1, particularly the choice of functional description, and the Appendices. Appendix A is the FMECA table populated with the functions and their failure modes, from which Appendix B logically follows. Appendix C specifies the Functional, Performance and Regulatory Requirements.

The assumptions in Appendix D can arise during:

i) the drafting of Section 1 in particular the scope;
ii) the derivation of the safety requirements including completing the FMECA table; and
iii) the derivation of the functional, performance and regulatory safety requirements.

All the assumptions are listed in Appendix D.

4 Safety Assurance

Assurance is required that the safety requirements have been met during its realisation and transition into operational service.

Figure 2 shows a typical top-level argument to support a claim that the system, as realised, satisfies its safety requirements.

All aspects of the argument are important. However from experience the selection of a system architecture and the assessment of the design have presented the greatest challenges.

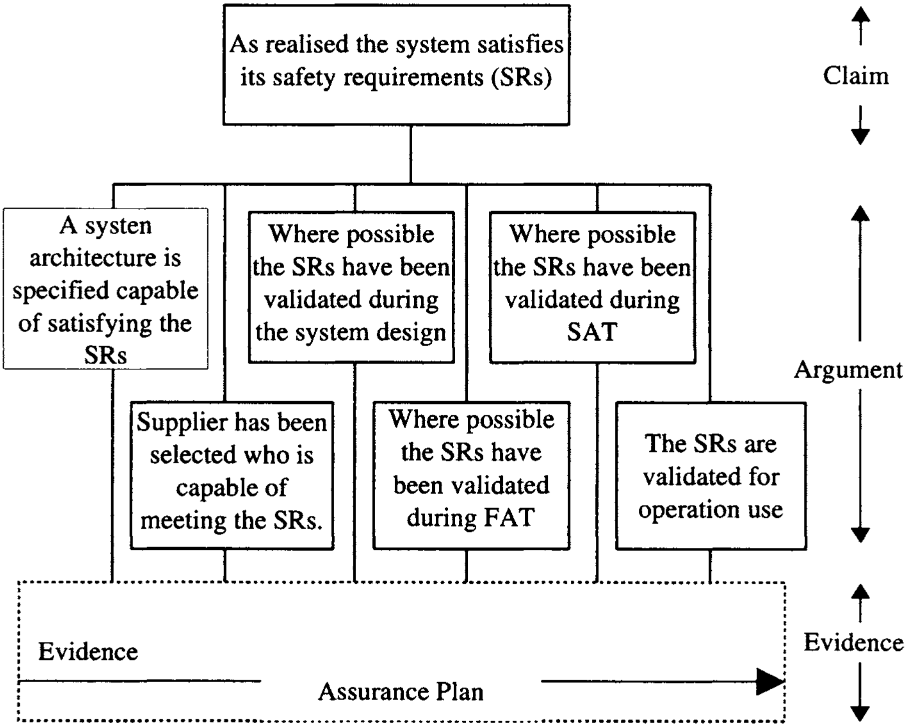

As realised the system satisfies its safety requirements (SRs)
Claim
A systen architecture is specified capable of satisfying the SRs
Where possible the SRs have been validated during the system design
Where possible the SRs have been validated during SAT
Argument
Supplier has been selected who is capable of meeting the SRs.
Where possible the SRs have been validated during FAT
The SRs are validated for operation use
Evidence
Assurance Plan
Evidence

Figure 2 – Simplified schematic of safety assurance

4.1 System Architecture

The term “System Architecture” as used here is equivalent to the safety requirements allocation process as described in IEC61508 Part 1 requirement 7.6.1 [Ref. 7]. It precedes the system design architectural phase undertaken by the system supplier.

The choice of system architecture can greatly affect the burden placed on the system supplier in respect of meeting the safety requirements and the residual risk when the system is in operational use.

The approach adopted for the safety requirements allocation is to take the most onerous safety requirement(s) to be met and propose a system architecture which has the potential to meet the requirement. A Fault Tree Analysis (FTA) is performed, apportioning target safety integrity levels (SILs) to the system elements. The proposed architecture, associated fault tree and apportionment are iterated until an acceptable architecture is found.

Of particular concern is a system element containing software. It all the cases considered the suppliers used Microsoft Windows or Linux for the software operating environment. It is also usual for a lack of segregation between safety and non-safety related software. Thus a claim of a Safety Integrity Level (SIL) greater than 1 is problematic. Having a diverse/redundant architecture, for example a main and back-up air traffic control position, results in system elements with target SILs which have the potential for being assured.

To date, for all the assessments which form this case study, this work has been performed on behalf of the aerodrome operator without the formal involvement of the supplier. There is a strong case for a supplier when proposing a system solution to include assurance that the architecture is "fit-for-purpose" and that the safety requirements can be met.

4.2 Assessment of Design

For this case study, the suppliers of a system has always been contracted to provide all necessary assurance, which in practice means safety assurance of their equipment.

Ideally, a comprehensive assessment of the system design should be made which makes a compelling argument supported by evidence that the system safety requirements will be met by the design implementation.

The performance of suppliers is mixed, from good to indifferent. In one case extensive safety assurance was provided. In reality an assessment of a design by a supplier is often restricted to hardware reliability and proven in use. Occasionally suppliers have claimed compliance with IEC61508 for software and there is growing evidence of suppliers performing more detailed system analyses, for example failure mode, effects and criticality analysis (FMECA) of system assemblies.

In all cases, because of inadequate involvement by a supplier at the system architecture stage (Section 4.1), there is additional work required to relate even a good assessment of a design to the safety requirements derived from a functional model of the system requirements.

CAA [Ref. 1] and European standards [Ref. 2, 3 and 4], have been published for safety assessments, but from experience do not provide the detail contained in the international standard for the functional safety IEC61508 [Ref. 7]. Adherence by suppliers to a standard like IEC61508 would simplify considerably the provision of a safety case by the aerodrome operator.

4.3 Other issues

Concluding this section on safety assurance, it is expected, where possible, that safety requirements will be validated in a factory acceptance test (FAT) and site acceptance test (SAT). These validations should provide increasing assurance that the safety requirements have been met. From experience, at this stage, the emphasis is moving from providing assurance on the safety requirements with a tolerable hazard occurrence rate to assurance of the functional, performance and regulatory safety requirements.

In addition, the transition into service of a new or changed system, when an air traffic service is being continuously provided, can be hazardous, so a hazard analysis of a transition and reversion strategy is required.

4.4 Packaging of a Realisation Safety Case

The template with guidance for a Part 2 Realisation Safety Case is more complex than a Part 1 Safety Case, but still quite straightforward:

1 Introduction
2 System Design
3 System Operation and Maintenance Arrangements
4 System Assurance
Definitions
References
Appendix A – Status of Safety Requirements
Appendix B –Safety Requirements Resolution Summary
Appendix C – Assumptions Validation Summary

Table 4 – Contents of a Realisation Safety Case

As with the Requirement Safety Case, the template should have the attributes expected of an ISO 9001 compliant document.

Again, a personal preference exists to include key detail on the safety requirements and assumptions where possible in appendices. Now however, significant more guidance is provided for detail to be included in Sections 1 to 4 of the main body of the document.

Section 1 Introduction is essentially a "cut and paste" of the Requirement Safety Case Introduction. For the system it details its purpose, provides a functional description and defines its scope.

Section 2 details the actual system design. A complete system description should include details on: environmental specification, system configuration (software as well as hardware); dependencies on which the system design relies; and design and engineering authorities.

Section 3 details the operational and maintenance arrangements. Also included are any requirements for performance monitoring and limitations on use, principally due to any short fall on the assurance provided that the safety requirements and assumptions are met.

Section 4 details the actual system assurance provided. The assurance should address: safety requirements with a tolerable hazard occurrence rate; functional, performance and regulatory requirements; and assumptions. Clear conclusions should be reached and proposals for any outstanding issues made.

Clearly Section 4 on System Assurance is key. The structure for the high-level assurance argument is illustrated by Figure 2 with the claims, arguments and evidence decomposed further as necessary. Also, as the Realisation Safety Case is issued at key milestones, any proposals for outstanding issues should reduce as increasing assurance is provided.

5 The European Interoperability Requirement

The Single European Sky (SES) regulations, effective from 20 April 2004, introduced new requirements and procedures that impact on airports which provide air traffic services. A key aspect of the SES regulations is the Interoperability Regulation [Ref. 5] whose objective is to achieve interoperability between the systems, sub-systems and associated procedures within the European Air Traffic Management Network.

5.1 Interoperability Requirements

The Interoperability Regulation has a main body containing twelve Articles and five Annexes.

Annex I of the Interoperability Regulations list eight systems for air navigation services.

The Interoperability Regulation has Essential Requirements, listed in Annex II of the Regulation, which the systems have to meet. The Regulation calls for the creation of Implementing Rules necessary to complement and further refine the Essential Requirements. The Regulation also calls for the establishment of Community Specifications. Compliance with a Community Specification creates a presumption of conformity with the Essential Requirements and any relevant Implementing Rule.

The provider of an air navigation service must provide an EC Declaration of Verification, confirming compliance, and shall submit the declaration to the National Supervisory Authority together with a Technical file. The contents of the EC Declaration of Verification are listed in Annex IV. The verification process has four elements: the contents of an EC declaration; verification procedure for systems; technical file; and submission.

The supplier of air navigation equipment shall provide an EC Declaration of Conformity or Suitability for Use, where either:

i) an assessment has been made regarding the intrinsic compliance of the system in isolation with the relevant community specification; or
ii) an assessment has been made regarding the suitability of the system within its air traffic environment.

The contents of a Declaration of Conformity or Suitability for use are listed in Annex III.

There is a requirement for a Notified Body appointed to carry out tasks associated with the assessment of an EC Declaration of Verification by an air navigation service provider and an EC Declaration of Conformity or Suitability for use by a supplier of equipment. The requirements of a Notified Body are listed in Annex V.

5.2 Impact of Interoperability Requirements

All the cases listed earlier in 1 Introduction are included within the scope of the Interoperability Regulations.

At present, there are no Implementing Rules or Community Specifications, so compliance is with the Essential Requirements. Also, there are no Notified Bodies.

Summarising, the "deliverables" to meet the Interoperability Regulations are:

i) a Technical File from the air traffic services provider;
ii) a Declaration of Verification (DoV) from the air traffic services provider; and
iii) a Declaration of Suitability of Use (DSU) from the suppliers of the systems constituents.

The Technical File would appear to be similar to a well-constructed, high-level safety case, including or referencing further detailed information.

Currently, it is judged that the UK CAA SRG will accept a safety case approach (as a de-facto technical file) together with an EC Declaration of Verification, as a submission in satisfaction of the requirements of Annex IV. The proviso is the safety case must address all the requirements listed in the Interoperability

Regulations, including copies of all Certificates of Conformity or Declaration of Suitability of Use.

The Declaration of Verification, Certificates of Conformity and Declaration of Suitability of Use are typically one page documents signed by an authorised representative of the operator and supplier respectively, stating that from their understanding the interoperability requirements have been complied with. Clearly, as such, the authorised representative should ensure that the requirements have been met before signing any document.

6 Conclusions

This paper is a case study based on experience gained from a range of current safety assessments of air traffic systems as used at medium-sized UK airports.

It has reviewed the following three stages of safety assessments for air traffic systems, the issues that have arisen and how they have been addressed:

i) how to establish the tolerable level of risk for a particular air traffic services context;
ii) how to derive safety requirements expressed as the integrity required of a safety function, through the allocation of a risk reduction burden; and
iii) how to provide assurance regarding the degree to which safety requirements have be met in practice.

The key issues that presented themselves, for which solutions have been found, which both meet quality objectives and are pragmatic in their execution, are:

i) constructing a safety classification scheme for the air traffic sector for a specific context;
ii) recognising the importance of an accurate functional description together with the associated failure modes;
iii) recognising the importance of assumptions and their validation;
iv) deciding on an appropriate structure for a Safety Case;
v) recognising the importance of the system architectural stage in apportioning integrity requirements in a realisable solution;
vi) noting the difficulty in eliciting from air traffic system suppliers a system assessment which provides convincingly the assurance required; and
vii) the recent impact of the EU directive on interoperability and how it should be met in practice.

Finally, it is acknowledged that a range of safety assessment techniques and measures exist, and that each industrial or business sector has developed approaches appropriate for their situation. However, the basic philosophies of the approach described here and the issues that arise are judged to apply generally.

References

1 CAP 760 Guidance on the conduct of Hazard Identification, Risk Assessment and the production of Safety Cases. For Aerodrome Operators and Air Traffic Service Providers, UK CAA 2006.

2 ESARR 3 Use of Safety Management Systems by ATM Service Providers, EUROCONTROL, 2000.

3 ESARR 4 Risk Assessment and Mitigation in ATM, EUROCONTROL, 2001.

4 ESARR 6 Software in ATM Systems, EUROCONTROL 2006.

5 Regulation (EC) no 552/2004 of the European Parliament and of the Council on the interoperability of the European Air Traffic Management network, European Union 2004.

6 CAP670 Air Traffic Services Safety Requirements, UK CAA 2003.

7 IEC 61508 Functional Safety of electrical/electronic/programmable electronic safety-related systems, International Electrotechnical Commission. 2002.

CARA: A Human Reliability Assessment Tool for Air Traffic Safety Management – Technical Basis and Preliminary Architecture

Barry Kirwan
Eurocontrol Experimental Centre
Bretigny sur Orge, F-91222 CEDEX, France

Huw Gibson
School of Electronic, Electrical and Computer Engineering, The University of Birmingham, Edgbaston, Birmingham, West Midlands.

1 Introduction

This paper aims to serve as the basis for development of a sound Human Reliability Assessment (HRA) capability for Air Traffic Management (ATM) applications in safety case and Human Factors assurance work. ATM is considered a 'high reliability' industry, although recent ATM-related accident occurrences have shown that such a status can never be assumed, and there is a continual need to look for safety vulnerabilities and mitigate them or their effects. Clearly, however, ATM is very human-centred, and will remain so at least in the mid-term (e.g. up to 2025). The air traffic controller has shown great capacity for safety over the years, and this must be maintained against a background of continually increasing traffic levels (currently running at 4 – 18% per year in Europe) and automation support aimed largely at enhancing capacity. Other industries have for several decades made use of HRA approaches. Such approaches aim to predict what can go wrong, and how often things can go wrong, from the human perspective. Such a capability is useful to ensure that safety cases of current and future systems are not ignoring the key component in the ATM system, the human controller.

However, it is not simply a matter of taking a HRA method off-the-shelf from another industry – ATM performance is very different from, say, nuclear power operation, rail transport, petrochemical or medical domain performance (domains where HRA has matured or is evolving). There is therefore a need to consider what approaches have been tried in such industries, and to learn from what has, and has not worked, and then fit and adapt a method that will serve ATM's needs. Additionally, whilst error types (what we do wrong) are relatively well-understood in ATM through incident experience, the likelihoods or probabilities of such errors, which are the cornerstone of any HRA method, are far less known. This is particularly so because error recovery in ATM is very strong.

Although other industries have such probabilistic human error 'data', ATM has almost none, and so it will take some time to develop an approach for ATM (since data from other industries may not be relevant). Nevertheless, preliminary studies have occurred using incident information from an Air Traffic Control Centre, error recordings from a real time simulation, and expert judgement protocols for an ATM safety case. Such initial studies do suggest that development of a HRA capability for ATM is feasible.

This paper therefore sets out to review HRA in other industries and to determine the overall architecture and style of HRA approach or approaches that are needed for ATM. It will then go on to give a vision of what such approaches would look like. Later companion reports will then focus on the development of these approaches, and their demonstration in safety case contexts. In summary therefore, the aims are as follows:

- Review HRA experience in other industries
- Determine the architecture of a HRA approach needed for ATM
- Give preliminary examples of the intended approach

2 Background to Human Reliability Assessment

Since in essence ATM wishes to learn from other industries to achieve human reliability assurance, it is useful to consider briefly the origins and evolutionary pathway of HRA in those other industries, so that ATM can learn and adapt its own methods more wisely. A précis of the evolution of HRA is therefore given below before focusing on ATM direct needs and prototype tools. Following this review and a summary of the key lessons to learn from other industries, a generic HRA Process is outlined, and examples where Eurocontrol has already made some advances are cited.

2.1 The Origins of Human Reliability Assessment

Human Reliability Assessment (HRA) is concerned with the prediction of human errors and recoveries in critical human-system situations. HRA started out in the early '60s in the domain of missile and defence applications in the USA. Early work focused on development of human error databases for simple operations (e.g. activate pushbutton), which could be compiled into task reliabilities (e.g. start reactor). However, this 'micro' attempt failed, because humans are primarily goal-driven, and their actions cannot be broken down into such minute sub-actions without losing something of the 'goal' that binds them together. Work nevertheless continued on HRA development, more or less as a research endeavour, until the end of the seventies (see Kirwan, 1994, for a review of early HRA developments).

2.2 The First Major HRA Technique

In 1979 the Three Mile Island nuclear power plant accident occurred, and was a fundamental shock to the industry, halting nuclear power advancement in the USA permanently, and bringing a huge focus on human error, Human Factors and the need for a better way of managing human reliability. The first true HRA technique was the Technique for Human Error Rate Prediction (THERP; Swain and Guttmann, 1983), which became available as a draft in 1981, published formally in 1983, and has been in use ever since. This technique contained a database of human reliabilities which were not too 'microscopic', and assessors found they could use the approach to deal with various human errors that could occur. The human error probabilities (HEPs) associated with each error type, typically stated as an error rate per demand, were based on the principal author's (Alan Swain's) experiences and data available from early US studies in the defence domain (e.g. the manufacture of missiles). Any HEP is a probability value between zero and unity, and a typical range for human error, from very likely to fail to highly unikely, is between 1.0 and 10^{-5}. In principle, HEPs are derived from observation of human performance:

HEP = No. of errors observed / No. of opportunities for error

The assessor needing to quantify a human error probability, say for a fault or event tree (THERP favoured event trees because they maintained the sequence of operations as seen by human operators, thereby maintaining the 'goal-orientation'), would find the most appropriate human task description in THERP (e.g. reading an analogue display; operating a pushbutton; etc.), and would obtain a 'nominal' HEP (e.g. 1 error in 1,000 operations or demands). This HEP could then be modified by the assessor within a range specified (e.g. a factor of ten) by the technique based on factors evident in the situation: for example, if there was significant time pressure on the operators, the assessor might modify the nominal HEP by a factor of ten, yielding a value of one in a hundred (10^{-2}) for the HEP being assessed.

These modification factors were called Performance Shaping Factors (PSF), and THERP specified many, from task-based factors such as time pressure, to psychological and physiological states such as emotional disturbances and fatigue. Although little guidance was given in the THERP documentation on how exactly to apply these PSF and modification factors, THERP assessors received a one to two week course to become accredited, and would be given instruction in application of modification factors (PSF).

THERP also recognized that a fault tree or event tree minimal cutset[1] could contain several errors that had to happen together for an accident scenario or hazard

[1] A minimal cutest is the minimum set of events required to happen to lead to an accidental outcome, or a hazard. In a fault tree it is a set of events multiplied together connected by one or more 'AND' gates to lead to the top event, and in an event tree it represents the combination of events on an unbroken pathway through the tree ending in a failure state.

to arise or progress. There would be cases however where one human error might lead to another, or at least increase its likelihood of occurring. This had in fact occurred in the Three Mile Island accident where, due to a misdiagnosis of the conditions inside the reactor vessel, several human 'recovery' tasks failed. This effect is known as dependence. An example in air traffic would be where the controller calls the wrong aircraft (e.g. due to call sign confusion) and attempts to give it the wrong instruction (e.g. climb to FL350). When the pilot reads back the instruction, in theory the controller should hear both the call sign and the instruction and realize that (s)he has made a mistake. But if the controller has truly confused the call signs, then the read-back would sound perfectly correct, because it matches the (mistaken) intention. In such a case, the recovery is completely dependent on the original error, and so will fail.

THERP recognized several levels of dependence, from zero to low, moderate, high and complete dependence, and developed simple equations with which to modify HEPs. THERP remains one of the only HRA techniques to explicitly tackle the issue of human dependence. Dependence remains a critical concern for ATM Concept changes, because such changes (e.g. changing from voice communication to electronic data transfer) can alter the dependence between certain controller and pilot tasks, and dependence effects on the total calculated risk for a new concept could be significant.

2.3 Other 'First Generation' HRA Techniques

Although THERP was immediately successful, there were a number of criticisms of the approach, in terms of its 'task decompositional' approach (it was still seen by some as too 'micro-task' focused), its database origins (which have never been published), its broad-brush treatment of psychological and Human Factors aspects, and its high resource requirements. By 1984 therefore, there was a swing to a new range of methods based on expert judgement. The world of Probabilistic Risk Assessment (PRA, later known as PSA, Probabilistic Safety Assessment), whose champion was (and still is) the nuclear power industry, was used to applying formal expert judgement techniques to deal with highly unlikely events (e.g. earthquakes, and other external events considered in PRAs). These techniques were therefore adapted to HRA.

In particular two techniques emerged: Absolute Probability Judgement, in which experts directly assessed HEPs on a logarithmic scale from 1.0 to 10^{-6}; and Paired Comparisons (Seaver and Stillwell, 1983; Hunns, 1982), where experts had to compare each human error to each other and decide simply which one was more likely. A psychological scaling process and logarithmic transformation were then used to derive actual HEPs. The latter approach required calibration: at least two known human error data points to 'calibrate' the scale. A third expert judgement technique, still in limited use today, was also developed, called SLIM (the Success Likelihood Index Method; Embrey et al, 1984). The main difference with this technique was that it allowed detailed consideration of key performance shaping factors (PSF) in the calculation process; the experts identified typically 4 – 8 critical PSF, weighted their relative importance, and then rated the presence of each PSF in each task whose reliability was required. This produced a scale, as for Paired Comparisons, which could then be calibrated to yield absolute HEPs.

One further technique worthy of note was developed is 1985, the Human Error Assessment and Reduction Technique (HEART: Williams, 1986; 1988). This technique had a much smaller and more generic database than THERP, which made it more flexible, and had PSF called Error Producing Conditions (EPCs), each of which had a maximum effect (e.g. from a factor of 19 to a factor of 1.2). HEART was based on a review of the human performance literature (field studies and experiments) and so the relative strengths of the different factors that can affect human performance had credibility with the Human Factors and Reliability Engineering domains. At the time of its initial development, HEART was seen as quicker compared to the more demanding THERP approach, but its industry-generic nature meant that it was sometimes not always clear how to use it in a specific industrial application. This tended to lead to inconsistencies in its usage. Later on however, such problems were addressed, firstly within HEART itself, and secondly by developing tailored versions for particular industry sectors, notably nuclear power and, very recently, rail transport (Gilroy and Grimes, 2005).

In the mid-late 80s, a number of further accidents in human-critical systems occurred: Bhopal in India, the world's worst chemical disaster; Chernobyl in the Ukraine, the world's worst nuclear power plant disaster; the Space Shuttle Challenger disaster; and the offshore oil and gas Piper Alpha disaster. All had strong human error connotations, but they also shifted concern to the design and management aspects, and the wake of Chernobyl in particular led to the notion of Safety Culture, as an essential aspect of system (and human) risk assurance as well as the impact of goal-driven behaviour on system safety, and in particular 'errors of intention'. These latter types of errors referred to the case wherein a human operator or team of operators might believe (mistakenly) that they were acting correctly, and in so doing might cause problems and prevent automatic safety systems from stopping the accident progression. Such errors of intention are obviously highly dangerous for any industry, since they act as both an initiating event (triggering an accident sequence) and a common mode failure of protective systems.

2.4 Validation[2] of HRA technique

Near the end of the '80s, a number of HRA techniques therefore existed, and so assessors in several industries (mainly at that time nuclear power, chemical and process, and petrochemical industries) were asking which ones 'worked' and were 'best'. This led to a series of evaluations and validations. A notable evaluation was by Swain (1989), the author of THERP, who reviewed more than a dozen techniques, but found THERP to be the best. A major comparative validation was carried out (Kirwan, 1988) in the UK nuclear industry involving many UK

[2] Validation means that the technique is used to predict HEPs for a set of tasks whose actual HEPs are known (but not to the assessors). Ideally the estimates are accurate to within a factor of three, but at least a factor of ten. Validations can also detect if a technique tends towards optimism/pessimism. See [18, 19].

practitioners, and using six HRA methods and 'pre-modelled' scenarios (using mainly single tasks and event trees). This validation, using assessors from industry, and using some 'real' data collected from incident reports and unknown to the assessors involved, led to the validation of four techniques, and the 'invalidation' of two. The empirically validated techniques were THERP, APJ (Absolute Probability Judgement; direct expert judgement), HEART (which had border-line validity), and a proprietary technique used by the nuclear reprocessing industry and not in the public domain. The two techniques that were 'invalidated' (i.e. they produced wrong estimates, typically wrong by a factor of ten or more), were Paired Comparisons and SLIM. Both of these techniques' results suffered because of poor calibration during the validation exercise.

2.5 A Wrong Path

In the late '80s an approach called Time Reliability Curves (TRC: Hannaman et al, 1984) was developed in several versions. Fundamentally this approach stated that as time available increases over time required for a task, human reliability increases towards an asymptotic value. Various curves were developed of time versus performance. However, while such curves had strong engineering appeal, they were later invalidated twice by two independent studies (Dolby, 1990; Kantowitz and Fujita, 1990) and were largely dropped from usage[3].

2.6 Human Error Identification & Task Analysis

In the mid-late 80's a trend also emerged with a significant focus on human error identification, and more detailed understanding of the human's task via methods of task analysis (several of which have already been applied in ATM)[4]. The need for this focus was simple and logical: the accuracy of the numbers would be inconsequential if key errors or recoveries had been omitted from the risk analysis in the first place. If the risk analysis treated the human tasks superficially, it was unlikely to fully model all the risks and recoveries in the real situation. This led to a number of approaches (Kirwan, 2002; Kirwan and Ainsworth, 1992) and theories and tools. In particular one of these was known as Systematic Human Error Reduction and Prediction Approach (SHERPA: Embrey, 1986), and was based on the works of key theoreticians such as *James Reason* and *Jens Rasmussen*, and was the ancestor of the later ATM error identification approaches TRACER (Shorrock and Kirwan, 2002), the incident Eurocontrol error classification approach HERA

[3] It is interesting to note that currently there has been some resurgence of interest in this approach in Bulgaria and Hungary, but elsewhere largely it is no longer looked upon favourably.

[4] The most useful task analysis techniques are Hierarchical Task Analysis (HTA), which develops a top down description of goals, tasks and operations; Operational Sequence Diagrams (OSDs) which consider interaction of different personnel (e.g. tactical and planner controller; controller and pilot; etc.) and Timeline Analysis which plots actions along a temporal event-driven axis.

(Isaac et al, 2002) and its error identification counterpart HERA-Predict. HRA came to be seen not merely as a means of human error quantification, but also as the whole approach of understanding and modelling the task failures and recoveries, and making recommendations for error mitigation as well. Thus HRA became concerned with the complete assessment of human reliability, and this broadening of its remit persists until today, though at its core HRA remains a quantitative approach.

In the UK and some parts of Europe (though not France or Germany) HEART gained some dominance mainly due to its flexibility, its addressing of key Human Factors aspects, and its low resource and training requirements. In 1992 it was adopted by the UK nuclear power industry as the main technique for use in probabilistic safety assessments. Since it had been border-line in the first main validation, it was improved and successfully re-validated twice in 1995 and 1998 (Kirwan et al, 1997; Kirwan 1997a; 1997b; Kennedy et al, 2000), along with THERP and another technique known as JHEDI (Kirwan, 1997c), the latter remaining proprietary to the nuclear processing industry. JHEDI is of interest however, since it was based entirely on the detailed analysis of incident data from its parent industry. The argument was simple: the more relevant the source data was for the HRA technique, the more accurate, robust and relevant the technique would be. The approach of using incident data was also used in the German nuclear industry in the method called CAHR (Connectionism Assessment of Human Reliability) (Sträter, 2000), which focused on pertinent human error data and mathematical analysis of the data to represent more robust HEPs, their contextual conditions and likely human behavioural mechanisms (called cognitive tendencies).

2.7 HRA & Context – 2nd Generation HRA

In 1990 a prominent HRA expert (Dougherty, 1990) suggested, also based on the experiences of the accidents mentioned above, that most HRA approaches did not pay enough attention to context, i.e. to the detailed scenarios people found themselves in. Essentially, the argument was that considering human reliability in an abstracted format of fault and event trees was insufficient to capture the local situational factors that would actually dictate human behaviour, and lead to success or failure. This occurred at the same time as a growing concern, in the US in particular, with Errors of Commission (EOCs), wherein a human in the system does something that is erroneous and not required by procedures (e.g. shutting off emergency feed water to the reactor in Three Mile Island; disconnecting safety systems while running reactivity experiments in Chernobyl). Such errors of intention, relating to a misconception about the situation, were, as already noted earlier, severely hazardous to industries such as nuclear power. Incident experience in the US was suggesting that it was these types of rare errors, whose very unpredictability made them difficult to defend against, that were of most real concern.

Although a number of existing techniques did consider performance shaping factors and carried out detailed task analysis, determining the roles and working practices of operators, and also considering the detailed Human Machine Interfaces (HMIs) that they would work with (what they would see and hear), therefore addressing *context*, there was a heralding call for a new generation of HRA techniques that would focus more on the *context* that could lead to such errors of intention. Therefore work on a set of so-called 'Second Generation HRA' techniques began in the early – mid 90's. The most notable of these were 'A Technique for Human Error Analysis (ATHEANA : Cooper et al, 1996; Forester et al, 2004) and the Cognitive Reliability Error Analysis Method (CREAM: Hollnagel, 1993; 1998). Actually used in various nuclear safety assessments were also MERMOS (Le Bot, 2003) and CAHR (Sträter, 2005). ATHEANA is notable because it has had more investment than almost any other HRA technique. The qualitative part of the method is used for identification of safety-critical human interventions in several instances, but its use has so far been marginal due to residual problems over quantification. CREAM is a more straightforward technique that has had mixed reviews, although currently it is in use in the nuclear power domain in the Czech Republic. Some preliminary attempts to adapt it to ATM did not work well[5].

A recent HRA expert workshop[6] provided an overview of the current level of implementation of 2nd Generation approaches. Though the methods mentioned above were all applied already in various safety assessments, there is still more work to be done regarding the evidence of empirical validation compared to 'first generation' techniques. The Level of application and work to be done can also be seen from a special issue of a key reliability journal on HRA and EOCs (Sträter, 2004). One notable exception of a 2nd Generation technique that is in regular use and which achieved regulatory acceptance is the MERMOS technique used by EdF (Electricite de France) in its nuclear programme. However, this approach appears unused outside of EdF, and appears to rely heavily on real-time simulations particular to nuclear power.

The Second Generation HRA approach is clearly still under development. Taken to one extreme, there is less focus on individual errors, and more focus on determining what factors can combine to lead to an intrinsically unsafe situation, wherein errors of intention become increasingly likely. This concept has some resonance not only with accidents such as Chernobyl and Bhopal, but also with the Überlingen accident (a mid-air collision of two aircraft). In the latter tragic event, although there were discrete controller errors that were arguably predictable by 1st Generation HRA methods, there was nevertheless an aggregation of an unusually large number of factors that predisposed the whole situation to failure. This 'confluence' of negative factors is now also being looked at not only by HRA, but also by proponents of the new field of 'Resilience Engineering'.

3 Current Approaches in Use

Recently the HEART technique has been 're-vamped' using human error data collected over a ten year period in the CORE-DATA (Taylor-Adams and Kirwan, 1995; Gibson et al, 1999) database, to develop a new nuclear-specific HRA

[5] An EEC-based student project attempted to develop the CREAM approach for air traffic operations.

[6] Halden Reactor Project Workshop on HRA, October 2005, Halden, Norway

technique called NARA (Nuclear Action Reliability Assessment: Kirwan et al, 2004). As with HEART, this approach uses only a small set of generic tasks, and a tailored set of performance shaping factors along with maximum effects, and 'anchors' to help assessors decide by how much to modify the generic task probability. NARA has not yet been validated but has been successfully peer reviewed by the nuclear regulator and industry, and independent HRA experts in a formal peer review process. In a recent review by NASA (Mosleh et al, 2006), who are considering development of a HRA method for space missions to Mars, NARA was one of five techniques highlighted for short-term interest; HEART and CREAM were also highlighted along with ATHEANA, and a technique called SPAR-H (Standardised Plant Analysis Risk HRA Method: Gertman et al, 2005), which is a quick US HRA technique for nuclear power plant evaluations taking elements from HEART, INTENT and CREAM). It is interesting to note that neither NARA, HEART nor SPAR-H are 2nd Generation techniques.

Currently methods in use are, from 1st generation, HEART, SLIM, APJ, THERP (and its quicker version ASEP), JHEDI and SPAR-H, and from 2nd generation ATHEANA, MERMOS, CREAM, and CAHR.

Whilst there is continued interest in 2nd Generation approaches in the area of HRA development, in practice '1st Generation' techniques are the ones that are mainly being applied in real risk assessment and safety assurance work. For many ATM applications, as discussed later, it is likely that a 1st Generation style HRA method for ATM would suffice. However, a more advanced method could offer advantages in terms of precision and insight for more critical human error situations. For this reason, as discussed later, a two-tiered approach for ATM may ultimately be considered.

Another avenue is that of expert judgement. Formal expert judgement is still in use today as a HRA method for many applications, including its use in developing generic human error probabilities for key industrial tasks (e.g. for the UK rail industry). However, it is not the same as 'Engineering Judgement'. The latter term refers to one or more people giving their opinion on the value of the HEP. Since it is well-known and well-researched that such unrefined expert judgement suffers from bias (Tversky and Kahneman, 1974; see also Kirwan, 1994), formal protocols are strongly advised. These entail selection criteria for defining appropriate expertise, proper preparation of materials for the exercise, expert facilitation of the expert judgement sessions, and statistical analysis of the results to detect unsound judgements. In practice this amounts to formal use of APJ to derive the raw expert judgement data, and use of Paired Comparisons (PC) to detect poor judgements (this is the part of PC which is strong and valid, and does not require calibration). This approach, together with HEART, was used recently in a preliminary Eurocontrol safety case for the GBAS (Ground-Based Augmentation System) project. These techniques (the APJ/PC partnership in particular) are recommended in preference to other expert judgement techniques such as SLIM, mentioned earlier, since the latter types of technique did not perform well in independent validation studies, whereas APJ has. However, it has to be said that the basic rule of expert judgement – ***garbage in, garbage out*** – applies. When true experts are using their many years of experience to judge how often events in their experience have occurred, that is one thing. It is quite another to assume that experts can predict behaviour with future systems that are very different from today's, wherein by definition there is no experience possible. Nevertheless, expert judgement may have a role to play in ATM HRA, possibly in terms of filling some of the gaps in quantitative data that are needed to develop ATM HRA techniques.

4 Summary of Lessons from the Evolution of HRA

The first lesson is that HRA has worked for more than two decades in several industries, enabling risk-informed decision-making, evaluation and improvement of design choices and system performance, and protection from unacceptable risks. It is also notable that it is now reaching into other industries, notably rail and air transportation, and medical and pharmaceutical domains. Wherever human performance and error are critical in an industry, HRA is often seen as useful sooner or later.

The second lesson is that in practice the simpler and more flexible approaches are more usable and sustainable. While more advanced methods are always desirable, they can take a very long time to reach fruition and deliver real results. It may be more sensible for ATM therefore to start with a practicable approach while anticipating the need for enhancing the methods in respect to 2nd Generation techniques. At a time when many safety cases are being developed, there is a need for a practical methodology now, to ensure that basic human reliability issues are dealt with in a reasonable manner.

The third lesson is that ATM has some clear advantages. Incident data can be analysed to inform HRA approaches, to help generate HEPs and to better understand the factors affecting ATM. Real Time Simulations can be used both to generate useful HEP data and to inform and verify safety case results and predictions. ATM already has at least two advanced simulation techniques (TOPAZ and Air MIDAS) that may help in the future to deliver better ways of dealing with more complex and dynamic human reliability questions. A final advantage, one we must not lose, is that the human reliability of our controllers is exemplary. This may be due to higher recovery rates or lower level of automation[7] than in other industries, but it means that we must better understand this high reliability, and more explicit modelling and measurement of this phenomenon will help us understand it better, so that we will know how to keep it.

5 ATM HRA Requirements

ATM has at least four clear application areas for HRA:

- System-wide safety cases for Next Generation ATM Systems (e.g. in Europe for SESAR, or in the US potentially for NGATS)

[7] Paradoxically, automation often makes error situations worse, because humans are less 'in the loop' and therefore do not detect problems nor correct them so easily.

- Individual concept element safety cases (e.g. a safety case for a new conflict resolution system, or for an arrival manager, etc.)
- Unit safety cases (e.g. a safety case for Maastricht Upper Airspace Centre, or another Air Traffic Control Centre or Airport)
- A Human Factors-driven HRA focusing on a specific current problem area or proposed change that may have impact on human error and recovery performance.

Given the number of safety cases that Eurocontrol needs to do in the short to medium term, and that Eurocontrol should deliver guidance on this key area to other stakeholders (Member States) for their implementation of ESARR 4 in their own safety cases, there is a pressing need for a short-term solution which also facilitates longer-term needs on HRA. This also clarifies the target audience for a HRA method – it is primarily aimed at the safety assessor or Human Factors assessor working on safety case or Human Factors Case assurance, whether for existing or future designs.

The most successful approaches in other industries have been flexible and tailored to the industry. Techniques such as HEART (and its nuclear power domain successor NARA) and SPAR-H and even CREAM, have enabled safety assessors and human reliability assessment practitioners to deal with human error without getting bogged down with the weight of the technique itself. Such 'light' tools are useful for most safety case needs. It would appear sensible therefore that ATM develop a similar approach, using generic task types relevant to the industry and safety case needs (i.e. typical tasks or errors modelled in safety cases), with appropriate modification factors (e.g. related to traffic, weather, HMI, etc.). Such an approach could be developed based initially on information from generic databases such as CORE-DATA (which includes air traffic human reliability data), shored up with formally produced and verified expert judgement data, and data collected in real-time and Human Factors laboratory simulations. The approach being developed is called CARA (Controller Action Reliability Assessment), whose testing and release are targeted to be done in 2007. The following section outlines the architecture and preliminary developments of CARA.

6 Preliminary Outline of CARA

This section shows what CARA might look like in the near future, based on preliminary work on the method. The aim is mainly to show the architecture of the approach and an overview of the technique 'mechanics' (how it will work). This will enable practitioners (e.g. safety and Human Factors assessors) to visualise the technique and its potential for practical applications.

There are three key elements of the CARA approach, using the same building blocks as were found to be successful in the HEART technique. These are outlined below in terms of their adaptation to CARA.

6.1 Generic Task Types (GTTs)

During a Human Reliability Assessment (HRA) an analyst will have specific tasks they need to quantify. A specific task is compared with Generic Tasks Types (GTTs). The GTT which best matches the specific task being assessed is selected. The selected GTT is associated with a human error probability and therefore this provides an initial quantification for the task being assessed. The GTT approach aims to have not too many GTTs, but to be able to cover all ATM-relevant tasks. CARA will have GTTs tailored to the ATM context. Initial quantification of GTTs (with uncertainty bounds where possible) will occur using data from the CORE-DATA Human Error Database. Preliminary GTTs for Air Traffic Management are as shown in Table 1.

Broad Task Type	**Generic Task Type**	**Comments**
ATCO Routine Tasks	1. Issue routine safe clearance or carry out in-sector task.	Ideally data would be collected for both GTT(1), as this level of resolution would be very useful for ATC HRA studies. GTT1 is 'high level', but should be used to represent any of the following sub-tasks: • Call aircraft on frequency (ident-assume) • Give clearance • Respond to pilot requests • Ensure aircraft at exit flight level • Hand off aircraft to next sector • Manage conflicts • Maintain smooth orderly expeditious flow • Coordinate aircraft into sector
	2. Plan aircraft into/out of sector (Planner/MSP)	This task is currently a 'place holder' as the assessment requirements, data sources and actual level of resolution for this task are currently uncertain.
ATCO Conflict Resolution	3. Detect deviation in routine scan of radar picture or strips	It should be noted that it is intended that tasks 3-6 are mutually exclusive and should always be considered/modelled separately. The current barrier models and event trees being used in ATM safety cases consider conflict detection and resolution with and without aids such as STCA (or MSAW, etc.), and responses to TCAS acknowledgement. Therefore separate GTTs will be
	4. Resolve conflict (not with STCA) when identified	
	5. Respond to STCA/alarm	
	6. Collision avoidance actions	

Broad Task Type	Generic Task Type	Comments
		required. This will also be useful where new safety nets or system safety defences may be considered.
ATCO Offline Tasks	7. ATCO offline tasks	For example, check papers, notes or notices prior to a shift.
Lower Level Tasks	8. Input/read data	These are slips in human perception and physical action, and could be used for call sign confusion or inputting wrong flight level in a Datalink transaction, for example.
	9. Communication of safety critical information	These are slips in the oral (e.g. Radio-telephony) communication of safety critical information. The GTT HEP is likely to be 0.006, based on data collected during Eurocontrol real time simulations. Data also exist on pilot error rates for communication.
	10. Evaluation/Decision Making	These are intended to be simple, individual, decision-making tasks.
Non-ATCO Tasks	11. Technical and support tasks	These are composite tasks such as are involved in the maintenance of ATC equipment.
	12. Pilot tasks.	Currently these are out of CARA's scope (except for communication tasks), but some pilot GTTs would probably need to be developed (e.g. in particular for ASAS applications, but also for selecting wrong approach, failing to level off, etc.).

Key: ASAS = Airborne Separation Assurance System; ATCO = air traffic controller; MSAW = Medium Safe Altitude Warning; MSP = Multi-Sector Planner; STCA = Short Term Conflict Alert;; TCAS = Traffic Alert and Collision Avoidance System

Table 1: Proposed Generic Task Types

6.2 Error Producing Conditions

In addition to GTTs there are also Error Producing Conditions (EPCs) which may be relevant to a task being assessed and which are captured as part of the human reliability assessment process. The EPCs are factors which are predicted to negatively influence controller performance and therefore increase the generic human error probability associated with a GTT. Examples of EPCs are 'time pressure' or 'controller inexperience'. Each EPC has a 'maximum effect' on performance, which is a numerical value which reflects the maximum impact that an EPC can have on a task. For existing CARA-like tools (HEART & NARA) the ranges of maximum effects are typically from around a factor of 2 to a factor of 20. Similar ranges would be likely for CARA since EPC effects appear to be more stable across industries than GTTs, since the latter are more context-dependent. A list of EPCs (see below) will therefore be developed for CARA specific to the ATC context. The CARA technique will then allow the assessor to rate the degree of impact of each selected EPC on the task. This therefore requires assessor judgement, based on his or her experience, but also from any related qualitative material available such as task analysis or other Human Factors analyses, as well as incident information where this is available. Subject Matter Experts (e.g. air traffic controllers) may also help determine the degree of the EPC's impact for a specific

Ref No.	EPC DESCRIPTION
1	A need to unlearn a technique and apply one which requires the application of an opposing philosophy.
2	Unfamiliarity, i.e. a potentially important situation which only occurs infrequently or is novel.
3	Time pressure.
4	Traffic Complexity leading to cognitive loading.
5	Difficulties caused by poor position hand-over or shift hand-over practices.
6	Difficulties caused by team co-ordination problems or friction between team members, or inter-centre difficulties.
7	Controller workplace noise/lighting issues, cockpit smoke.
8	Weather.
9	On-the job training.
10	Cognitive overload, particularly one caused by simultaneous presentation of non-redundant information.
11	Poor, ambiguous or ill-matched system feedback.
12	Shortfalls in the quality of information conveyed by procedures.
13	Low vigilance or fatigue
14	Controller shift from anticipatory to reactive mode.
15	Risk taking.
16	High emotional stress and effects of ill health.
17	Low workforce morale or adverse organisational environment.
18	Communications quality.
19	Over or under-trust in system or automation
20	Unavailable equipment/degraded mode.
21	Little or no independent checking (e.g. lack of two pairs of eyes when needed).
22	Unreliable instrumentation or tool.

Table 2: Proposed Error Producing Conditions

situation. The longer-term intention of CARA will however be to provide 'anchor points' to help the assessors rate each selected EPC, as this has been found useful in other industries. Candidate EPCs for ATM are shown in Table 2. These have been derived by reviewing the original HERA and NARA techniques, as well as the Eurocontrol HERA taxonomy of errors and error factors, and from a practical knowledge of factors in incidents.

6.3 CARA Calculation Method

HEART uses a simple calculation method to combine GTT HEP and EPC values, and CARA will adopt the same procedure. It also allows modification of the strength of affect of EPCs through a weighting process. It is currently proposed that these processes are not changed. They are described in Kirwan (1994). As an example, there may be a communication task (using Radio-telephony) [GTT 9] such as instructing an aircraft to climb to a particular flight level, but there is a risk of giving the wrong FL. In the situation in question there may be a predicted problem in the quality of communications: CARA EPC 18 [maximum effect assumed to be a factor of 10 in this example] is used by the assessor and rated at half the maximum effect. The resulting HEP would therefore be around 0.03. It should be noted that if too many EPCs are used, the HEP rapidly approaches unity (1.0), and so with techniques like HEART and CARA, EPCs should only be used when they are definitely present in the current system, or are judged likely to be present in the future system.

The above outline of CARA shows the adaptation of the HEART and NARA-style techniques to ATM, via contextualising the GTTs and EPCs. The GTTs have been contrasted with the human errors identified in ten preliminary safety cases and have been found to be able to accommodate such error types. The EPCs have also been shown to be relevant to the types of contributory factors from incidents and those factors considered by assessors for safety analysis purposes. The next and critical step will be to quantify the GTTs for ATM, and to confirm or change the EPC maximum effects for the ATM context. Then the CARA technique will be ready for user trials.

7 Conclusion

HRA has been found to be useful in other industries and could be adopted by ATM. In the short term an ATM-specific tool is being developed based on generic tasks and probabilities with ATM-relevant modification factors. The candidate approach, now under development, is called CARA. This approach uses the HEART format, adapted to the context of air traffic management operations. The preliminary architecture has been developed and outlined in this paper, and work is now progressing on the quantification of the GTTs and EPCs, prior to trialling of the technique in ATM safety assessments.

Acknowledgements

The authors would like to acknowledge the useful comments of Dr. Oliver Sträter, Jacques Beaufays and Brian Hickling of Eurocontrol on early drafts of this paper.

References

Cooper, S.E., Ramey-Smith, A.M., Wreathall, J., Parry, G.W., Bley, D.C., Luckas, W.J., Taylor, J.H., and Barriere, M.T. (1996). A technique for human error analysis (ATHEANA) - Technical Basis and Method Description. Nureg/CR-6350. US Nuclear Regulatory Commission, Washington D.C. 20555.

Dolby, A.J. (1990). A comparison of operator response times predicted by the HCR model with those obtained from simulators. International Journal of Quality and Reliability Management, 7, 5, pp. 19 – 26.

Dougherty E.M. (1990). Human Reliability Analysis – Where Shouldst Thou Turn? Reliability Engineering and System Safety Vol 29: 283-299

Embrey, D.E., Humphreys, P.C., Rosa, E.A., Kirwan, B. and Rea, K. (1984). "SLIM-MAUD: An Approach to Assessing Human Error Probabilities Using Structured Expert Judgement". NUREG/CR-3518, US Nuclear Regulatory Commission, Washington, DC 20555.

Embrey, D.E. (1986). SHERPA - a systematic human error reduction and prediction approach. Paper presented at the International Topical Meeting on Advances in Human Factors in Nuclear Power Systems, Knoxville, Tennessee.

Forester, J., Bley, D., Cooper, S., Lois, E., Siu, N., Kolaczkowski, A., and Wreathall, J. (2004). Expert elicitation approach for performing ATHEANA quantification. Reliability Engineering & System Safety, 83, 2, 207 – 220.

Gertman, D., Blackman, H.S., Marble, Byers, J., Haney L.N., and Smith, C. (2005). The SPAR-H Human Reliability Analysis Method. US Nuclear Regulatory Commission Report NUREG/CR-6883, Washington DC 20555.

Gibson, H., Basra, G., and Kirwan, B. (1999). Development of the CORE-DATA database. Safety & Reliability Journal, Safety and Reliability Society, Manchester, 19, 1, 6 - 20.

Gilroy, J. and Grimes, E. (2005). The development and application of a rail human reliability assessment tool. In Proceedings of the Second European Conference on Rail Human Factors, London, 21-23 November 2005.

Hannaman, G.W., Spurgin, A.J., and Lukic, Y.D. (1984). Human cognitive reliability model for PRA analysis. Report NUS-4531, Electric Power Research Institute, Palo Alto, California.

Hollnagel, E. (1993). Human reliability analysis: context and control. London: Academic Press.

Hollnagel, E. (1998). CREAM: Cognitive Reliability Error Analysis Method. Elsevier Science Ltd, Oxford

Hunns, D.M. (1982). The method of paired comparisons. In Green, A.E. (Ed.) High risk safety technology. Chichester: Wiley.

Isaac, A., Kirwan, B. and Shorrock, S. (2002). Human error in ATM: the HERA project. Reliability Engineering and System Safety, 75, 2, 257 - 272

Kantowitz, B., and Fujita, Y. (1990). Cognitive theory, identifiability, and Human Reliability Analysis. Reliability Engineering and System Safety, 29, 3, pp.317 – 328.

Kennedy, R., Kirwan, B., Summersgill, R., and Rea, K. (2000). Validation of HRA techniques – déjà vu five years on? In Foresight & Precaution, Eds. Cottam, M., Pape, R.P., Harvey, D.W., and Tait, J. Balkema, Rotterdam.

Kirwan, B. (1988). A comparative evaluation of five human reliability assessment techniques. In Human Factors and Decision Making. Sayers, B.A. (Ed.). London : Elsevier, pp 87-109.

Kirwan, B. (2002). Human error identification in human reliability assessment – Part 1 – Overview of approaches. Applied Ergonomics, 23 (5), 299 – 318.

Kirwan, B. and Ainsworth L.A. (eds) (1992). A Guide to Task Analysis. London: Taylor and Francis.

Kirwan, B. (1994). A guide to practical human reliability assessment. London: Taylor & Francis.

Kirwan, B., Kennedy, R., Taylor-Adams, S. and Lambert, B. (1997). The validation of three Human Reliability Quantification Techniques - THERP, HEART, and JHEDI: Part II - Results of validation exercise. Applied Ergonomics, 28, 1, 17 - 25.

Kirwan, B. (1997a). Validation of Human Reliability Assessment Techniques: Part 1 - Validation Issues. Safety Science, 27, 1, 25 - 41.

Kirwan, B. (1997b). Validation of Human Reliability Assessment Techniques: Part 2 - Validation Results. Safety Science, 27, 1, 43 - 75.

Kirwan, B. (1997c). The development of a nuclear chemical plant human reliability management approach. Reliability Engineering and System Safety, 56, 107 – 133.

Kirwan, B., Gibson, H., Kennedy, R., Edmunds, J., Cooksley, G., and Umbers, I. (2004) Nuclear Action Reliability Assessment (NARA): A data-based HRA tool. In Probabilistic Safety Assessment and Management 2004, Spitzer, C., Schmocker, U., and Dang, V.N. (Eds.), London, Springer, pp. 1206 – 1211.

Le Bot, P. (2003). 'Methodological validation of MERMOS by 160 analyses' in Proceedings of the International Workshop Building the New HRA: Errors of

Commission from Research to Application. NEA/CSNI/R(2002)3. OECD: Issy-les-Moulineaux.

Mosleh, A. et al (2006: draft). Evaluation of current HRA methods for NASA use. NASA HRA Methods Workshop, Kennedy Space Centre, Florida, Jan 16 – 18.

Seaver, D.A. and Stillwell, W.G. (1983). Procedures for using expert judgement to estimate human error probabilities in nuclear power plant operations. NUREG/CR-2743, Washington DC 20555.

Shorrock, S.T. and Kirwan, B. (2002). Development and Application of a Human Error Identification Tool for Air Traffic Control. Applied Ergonomics, 33, 319 - 336.

Sträter, O. (2000). Evaluation of Human Reliability on the Basis of Operational Experience. GRS-170. GRS. Köln/Germany. ISBN 3-931995-37-2. August 2000.

Sträter, O., (2005). Cognition and safety - An Integrated Approach to Systems Design and Performance Assessment. Ashgate. Aldershot.

Sträter, O. (Ed.) (2004). Human reliability analysis: data issues and errors of commission. Special Issue of Reliability Engineering and System Safety, 83, no. 2.

Swain, A.D. (1989). Comparative evaluation of methods for human reliability analysis. GRS-81, Gessellschaft fur Reaktorsicherheit (GRS)mbH, Schwertnergasse 1, 5000 Koln 1, Germany.

Swain, A.D. and Guttmann, H.E. (1983). - A handbook of human reliability analysis with emphasis on nuclear power plant applications USNRC-Nureg/CR-1278, Washington DC-20555.

Taylor-Adams, S.T., and Kirwan, B. (1995). Human Reliability Data Requirements. International Journal of Quality & Reliability Management, 12, 1, 24-46.

Tversky, A. and Kahneman, D. (1974). Judgement under Uncertainty: heuristics and biases, Science, 185, 1124 – 1131.

Williams, J.C. (1986). "HEART - A Proposed Method for Assessing and Reducing Human Error", Proceedings of the 9th "Advances in Reliability Technology" Symposium, University of Bradford.

Williams, J.C. (1988). A data-based method for assessing and reducing human error to improve operational performance. In IEEE Conference on Human Factors in Power Plants, pp. 436 - 450. Monterey, California, June 5 - 9.

High Integrity from Specification to Code

AMBERS: Improving Requirements Specification Through Assertive Models and SCADE/ DOORS Integration

Marcelin Fortes da Cruz
Airbus UK Systems Engineering,
Bristol, UK

Paul Raistrick
Esterel Technologies,
Crowthorne, UK.

Abstract

Development of requirements specifications is a key activity in the development of a system. Errors in a requirements specification can cost orders of magnitude more to detect and fix than errors in the implementation. Model based development techniques can help validation of requirements specifications by allowing early simulation and testing. However models are created by interpreting written requirements, and potential representation errors continue to exist.

This paper reports on 'AMBERS', or Assertive Model-Based Engineering Requirement Specifications, an Airbus initiative to improve the quality of engineering specifications by providing a common framework for requirements engineers and modelling engineers to work in. The AMBERS framework builds on the Software Cost Reduction US-NRL method to augment textual requirements with assertive (Parnas) function tables and creates a bridge to model-based developments by using these tables as proof objectives that a model must comply with. This supports proof-guided simulation and testing, allowing more effective use of validation activities.

Extending DOORS and SCADE to provide a two-way traceability between model and requirements specification, and to provide support for automatic proof generation has allowed developing a tool support prototype for the 'AMBERS' approach.

1 Introduction

As a major aircraft manufacturer Airbus creates many system specifications. These specifications are passed to internal and external organisations for implementation and delivery. Airbus carries out system integration on test-rigs, simulators, prototype aircraft and finally production aircraft. Clearly specification quality is key to successful procurement; specification errors can be orders of magnitude more costly to detect and fix than implementation errors. It suffices to say that the number of 'No Fault Found' (NFF) and Service Bulletin (SB) are tightly dependent on the quality of the specifications. Specifications are key as far as correct, complete and consistent systems interpretation, representation and implementation are concerned.

Airbus manages systems procurement via a set of documents, which capture systems and components requirements that are supported by design models expressed in various industrial tools. To manage these documents Airbus uses a set of defined development stages, known as Aircraft Concurrent Engineering (ACE) stages. These stages identify which documents are to be produced at which stage in line with appropriate international standards: e.g. SAE ARP 4754, ARP 4761, RTCA DO 178(A, B), DO-254.

1.1 Lifecycle – ACE Programme

The purpose of the ACE programme is to provide a temporal framework that is common to all systems engineering departments together with progress review and decision points. The ACE programme covers the systems Feasibility, Conceptual, Definition and Development aspects of the systems at aircraft and systems level. The Airbus ACE programme is made of 14 engineering milestones, which are shown in summary in Figure 1.

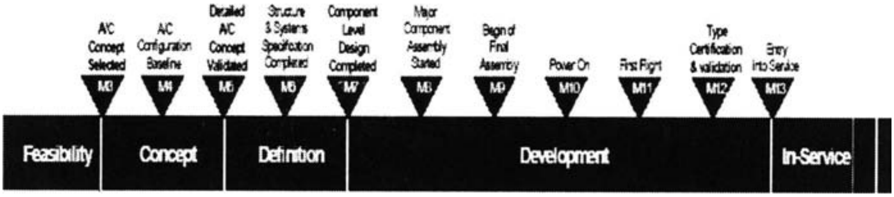

A/C Concept Selected
M3
A/C Configuration Baseline
M4
Detailed A/C Concept Validated
M5
Structure & Systems Specification Completed
M6
Component Level Design Completed
M7
Major Component Assembly Started
M8
Begin of Final Assembly
M9
Power On
M10
First Flight
M11
Type Certification & validation
M12
Entry into Service
M13
Feasibility
Concept
Definition
Development
In-Service

Figure 1: Aircraft Concurrent Engineering (ACE) Stages Outline

1.2 Current Problems in Systems Requirements Specifications

Within the development process many different stakeholders are required to contribute towards, review and analyse specifications. The different stakeholders include plant and control system design specialists, safety specialists and reliability specialists. A key difference between these stakeholders is the ways in which they view the specifications.

Plant and Control system design specialists are interested in specifying the relationship between inputs and outputs of the system through the definition of statics and dynamics equations and control laws. For instance, Control laws are developed using modelling tools in which control laws are implemented and simulations are run against a model of the system environment to validate whether the control laws give the required behaviour. This produces a model-based definition of system behaviour. In subsequent sections we describe this view as a 'Model Based Engineering' (MBE) view.

Other specialists, for example safety specialists, develop requirements that constrain the possible implementation. For example a safety specialist is interested in constraining system behaviour to one that is safe, which may include specifying input/output pairs that should not be produced. A reliability engineer is interested in specifying availability and reliability constraints on the system. These requirements are typically expressed as a textual requirement. In subsequent sections we describe this view as a 'Requirements Based Engineering' (RBE) view.

Each specialist applies formalisms and techniques that are independent of those used by other specialists. From this situation the following problems arise:

- Different mindsets and concerns between System Design (Model-Based) and System Integrity (Requirements-Based). This may lead to misinterpretation of specifications between the different stakeholders, leading to divergent 'models' of a system.
- Non-integrated system interpretations and representations throughout the ACE milestones. Lack of integrated models means that identifying where differing interpretations have been taken is very difficulty and time consuming.
- Wide range of tools in use leading to incompatible formats. Whereas there are system representations that can, in principle, be integrated and checked for consistency it is often not feasible cost-effectively. This is because the different disciplines will use different tools with incompatible formats, so checking consistency requires expensive format conversions between models involving syntactic and semantic issues with regard to the different formalisms.
- Over-specification from insufficiently co-related requirements. As there are many stakeholders contributing to the final specification the same requirement may be expressed many times, and with unnecessary exactness. Identification of over-specification is made difficult because of different terminology used and because similar requirements are spread throughout large documents.
- Under-specification due to lack of accuracy in expressing the intended properties; for instance typed variables without boundaries.

1.3 An Approach to Address Requirements Problems

The brief summary of issues in the previous section, combined with the following observations set the background for AMBERS:

- On the one hand aircraft functions are interpreted and represented as models. Models are factually programs within the CASE or CAD tools.
- On the other hand aircraft requirements and specifications are interpreted and represented as constraints. Constraints can also be modelled and thus be factually programs (within the modelling tool). Models of constraints can be tightly associated to aircraft function models to constrain them and provide guaranteed behaviour by construction under known engineering assumptions within an 'Assume-Guarantee' Engineering Process.
- Could we combine both Model-Based Engineering (MBE) and Requirement-Based Engineering (RBE) interpretations and representations into homogenous formalisms that would allow building functions as constrained programs that could be simulated, tested, analysed and traced?

From this we can see the key question to be answered by AMBERS. There are broadly two types of problem to be addressed; those arising from the lack of integration of formalisms and those that could be termed 'standard' technical problems (i.e. problems that have been widely reported in other requirements engineering processes). The approach embodied in AMBERS is to identify ways to integrate disparate models and to introduce methods to help address the technical issues.

From a technical issues point of view formal methods are an attractive option as they allow powerful levels of consistency checking. However there are practical issues with 'traditional' formal methods. Historically practicing engineers have been reluctant to learn and use formal methods. Also the support provided by many of the formal methods available now for model based development could be weak or cumbersome.

The Software Cost Reduction Methods (SCR) [Heitmeyer C.L., Jeffords R.D. and Labaw B.G. (1996)] has widely been reported as a successful method to introduce rigour and formalism into the requirements engineering process in an engineer-friendly fashion. Consequently the AMBERS approach has taken key elements from SCR to support the requirements based engineering process.

To support model-based development the AMBERS approach is to identify a model based development tool with sufficiently rigorous semantics to allow for integration with the SCR method, and which supports open standards to allow integration with requirements management tools. There are a number of tools that support model-based development. However many of them have been developed from a simulation pedigree, and thus have relatively weak semantics. For example the semantics of a model can depend on the specific tool version or on hidden simulation settings.

SCADE (an acronym which stands for Safety Critical Application Development Environment) is a model-based development tool that is based on the Lustre formal language [Halbwachs N. et al (1991)] which is marketed by Esterel Technologies. SCADE has capabilities that allow design modelling, simulation, trusted code generation and proofs to be carried out over a model. As SCADE designs are defined in a formal language the semantics of the model are precisely defined. Model behaviour is therefore neither dependent on tool versions nor effected by hidden simulator settings.

Designs in SCADE are defined as a set of modular nodes that have a strongly typed interface. These nodes are connected together in SCADE diagrams by 'wires' that indicate data flow. SCADE allows definition of four kinds of node (also known as operators); safe state machine nodes provide a mechanism to model control flow, data flow nodes model data flow, imported nodes allow external source code to be used and textual nodes allow direct entry of the SCADE formal language. Behind the safe state machine and data flow nodes is an associated textual definition of the node in the SCADE formal language. At root these nodes (except imported nodes) are textual, however SCADE provides a graphical notation for data flow and safe state machine nodes.

Figure 2 shows an example of SCADE data flow notation implementing a counter. The diagram should be read left to right, with a feedback line returning through the 'PRE' block providing a memory. The inputs to the diagram are shown on the left (Init, Reset and Incr) and the output on the right (count). The diagram consists of a '+' operator, a switch to allow counter reset, an initialisation operator ('->') to ensure that count always has a valid value and the 'PRE' block previously described. This diagram highlights an important feature of SCADE designs: that they are based on a synchronous cyclic model of execution. In this model inputs are filled from the environment at the start of each cycle and held constant until the end of the cycle when the output is available. The 'PRE' operator is necessary to disambiguate the feedback from the output of the design to the input of the design. The output of the 'PRE' operator is the value of the input flow at the last cycle.

Figure 3 shows an example of a SCADE Safe State Machine (SSM) that is used to model control flow within a SCADE design. As you can see the look and feel of the state machine has similarities to UML Statecharts in that states may contain state machines, actions can be carried out on transitions, entry to a state, while in a state and on exit from a state. However there are important differences. As SSMs are deterministic and unambiguous a number of features of Statecharts are not supported, for example transitions allowed across state boundaries.

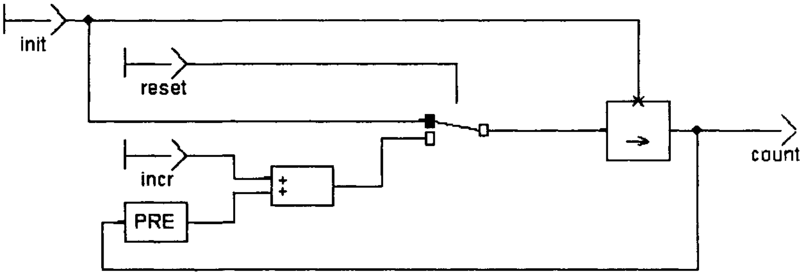


Figure 2 : SCADE Data Flow Implementation of a Counter

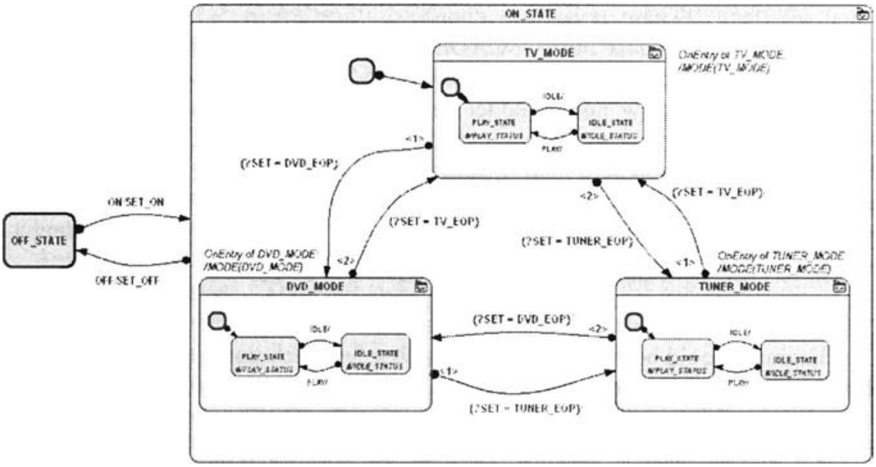


Figure 3 : Example of a SCADE Safe State Machine

SCADE allows the definition of proof objectives as a separate SCADE Node, called an 'Observer' node. The observer node encodes the property to be proved as a Boolean condition that evaluates to false if it is broken. The property is encoded using standard SCADE design elements, so no new notation is required. The observer node is connected to the node that you wish to carry out the proof over and the design verifier is used.

Figure 4 shows an example of an observer node, the connection to a design node and the internals of the observer node. In this example a cruise control system is being subject to proof of the property within the observer node. The observer node encodes a proof property which, informally, can be stated as 'If the Brake Input is more than 0.0 then the second element of the CruiseState data flow should be

True'. As you can see logical predicates such as 'implies' are available within SCADE.

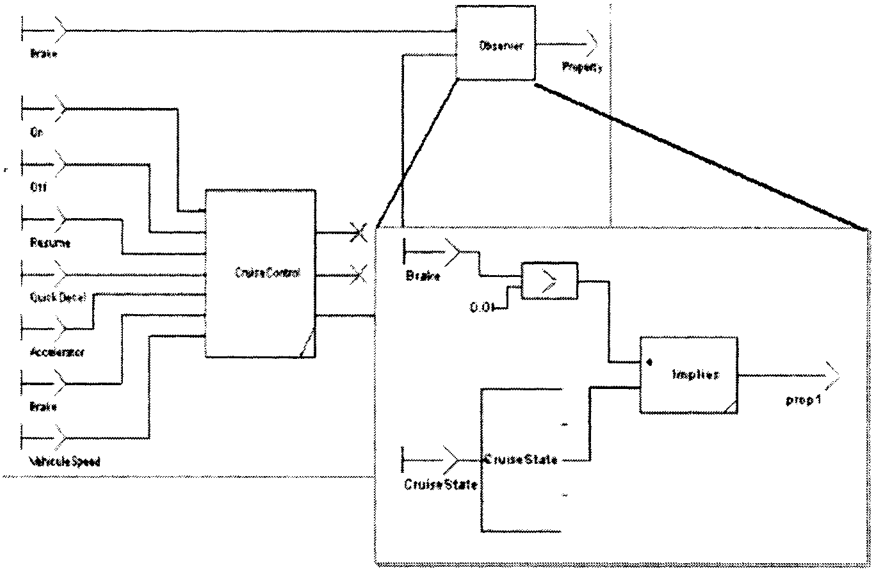


Figure 4 : Example of Proof Construction in SCADE

The design verifier attempts to falsify the property encoded in the observer. If it is able to falsify the property an example set of values that falsify the property are created. These values can be 'executed' in the simulator allowing the engineer to see the concrete behaviour that causes the property to be falsifiable. So it is possible for engineers without detailed knowledge of formal methods to rapidly understand proof failures and to identify errors within either the property (requirement-based) or the design (model-based).

In many ways SCADE wraps up a formally based model within a model-based development environment in a similar way to SCR that wraps up formalism behind a set of tables. For this reason SCADE was a natural choice to support AMBERS.

In order to answer the question posed at the start of this section it is necessary to firstly incorporate SCR methods into the requirements based engineering process, and then to build a bridge from the requirements based system definition to the model based system definition within SCADE.

1.4 AMBERS as Process Improvement

A final consideration in the development of AMBERS is that, as a process improvement activity, a way of managing the development and introduction of the method is required. Although the process improvement aspects of AMBERS are not the main focus of this paper it is valuable to point out how process improvement aspects have affected the architecture of the methodology.

AMBERS is an encompassing framework for the development and validation of requirements specifications. It is the authors belief that introduction of a large framework in a 'big bang' approach would rapidly lead to AMBERS becoming shelfware. To address the risk of this AMBERS has been developed as a framework of different elements. These elements have been defined to allow:

- Specific parts of AMBERS to be detailed separately in conjunction with stakeholders.
- Pilot studies and demonstrators of specific aspects of AMBERS to be developed quickly.
- Regular reviews of progress with stakeholders, and feedback of comments.

Consequently the framework of AMBERS is not completely described here. This paper reports on the technical details of parts of the framework.

2 AMBERS Methodology

Building on work from [Fortes da Cruz. M.Au. (2001)] 'AMBERS' is aimed at building, simulating, testing and analysing constrained programs that interpret and represent Functions as Assertive Signatures. It consists of a model-based part, and formal extensions to requirements documents.

The model-based part is a framework of four integrated dependency models:

- Conceptual;
- Functional;
- Behavioural;
- Causal.

These models are integrated into the requirements based engineering workflow through the definition of rules to ensure consistency with the SCR tables that extend requirements specifications.

Of these four models our work has concentrated on detailing and demonstrating the functional and behavioural models. The conceptual and causal models are merely outlined here for completeness.

2.1 Conceptual Model

The conceptual model is a high level description of the concepts pertinent to the system under development: e.g. form ACE milestones M0 TO M6. The detail of this model is not further described, but its purpose is to provide an unambiguous definition of terms and relationships. The relationships defined within this model then provide a set of constraints that the other models should comply with.

The conceptual model is based on the four sets of variable abstraction from the Parnas/SCR approach:

- Monitored variables: which are driven by the system assembly positions and motions to the system sensing;
- Controlled variables: which are driving the system assembly positions and motions from the system actuating;
- Input variables: which are converted/read to the system control from the system sensing;
- Output variables converted /written from the system control to the system actuating.

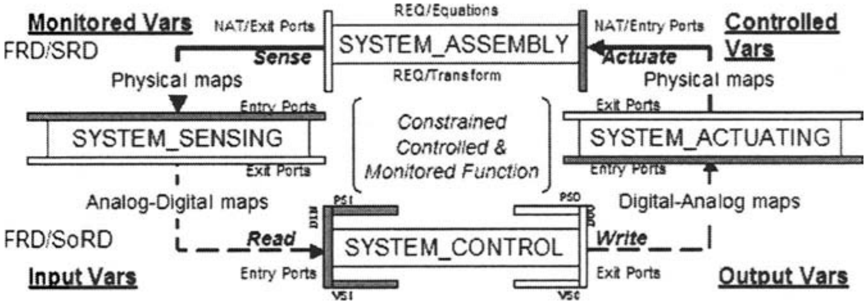


Figure 5: Parnas Four Variable Model

Although the detail of this model has not been defined yet it is possible to visualise it as an entity-relationship or class type diagram with associated data dictionary. Each concept (functional and behavioural class or entity) identified within the model will have associated attributes and abstract procedures. Constraints on the attributes and the procedure productions will also be detailed within the model in the very early stages.

2.2 Functional Model

The functional model is a detailed description of the required functional system behaviour. Within AMBERS the functional model is represented as a hierarchical model of 'Graphical Function Blocks'. We will refer to these simply as function blocks in the remainder of the paper. At the leaf of the model the function blocks are realised by data flow diagrams of control blocks within SCADE, as shown in Figure 2. Figure 6 shows an example of a function block. In the figure a function block is shown encompassing two lower level function blocks. The notation uses standard positions of 'ports' to provide cues to the reader about the type and purpose of a flow. The notation arranges input ports on the left hand side of the block and outputs on the right hand side of the block. A special kind of port, the parameter port is arranged on the left hand side to allow function blocks to be parameterised. Parameterisation is provided to support reuse of blocks.

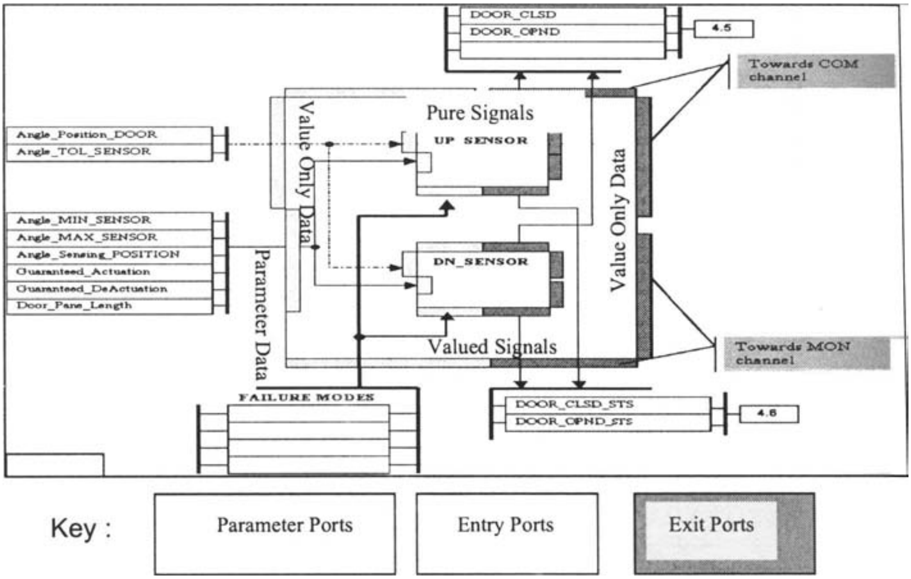


Figure 6: Graphical Function Block Notation

The notation supports three different kinds of flows:

- Data ports - integer, float, bool, character or string flows. These flow from left to right through the top left/ top right ports.
- Pure Signal Ports - Physical or digital discretes.
- Valued Signal Ports - Tuple of (boolean, basic_type). A valued signal is raised whenever the value constituent changes. When there is no change in value the last value is available for interrogation.

The functional model allows representation of both data flow (through data ports) and control flow aspects (through Pure Signal and Valued Signal ports) of the specification.

2.3 Behavioural Model

The behavioural model defines system state behaviour and activation of functions within the functional model. Signal-Action Graphs (SAG) are defined using an extension of the Grafcet notation [David R. and Alla H. (editors) (1992)] to define control flow at an abstract level. Again the behavioural model is refined hierarchically until at the leaf level a concrete specification is implemented using SCADE Safe State Machines, as shown in Figure 3.

2.4 Causal Model

The causal model supports various analyses, for example Reliability, Availability, Maintainability, and Safety (RAMS). This model will consist of symbolic (logic-based) and probabilistic (Bayesian) networks built from the interface variables and the functions defined in the conceptual, functional and behavioural models.

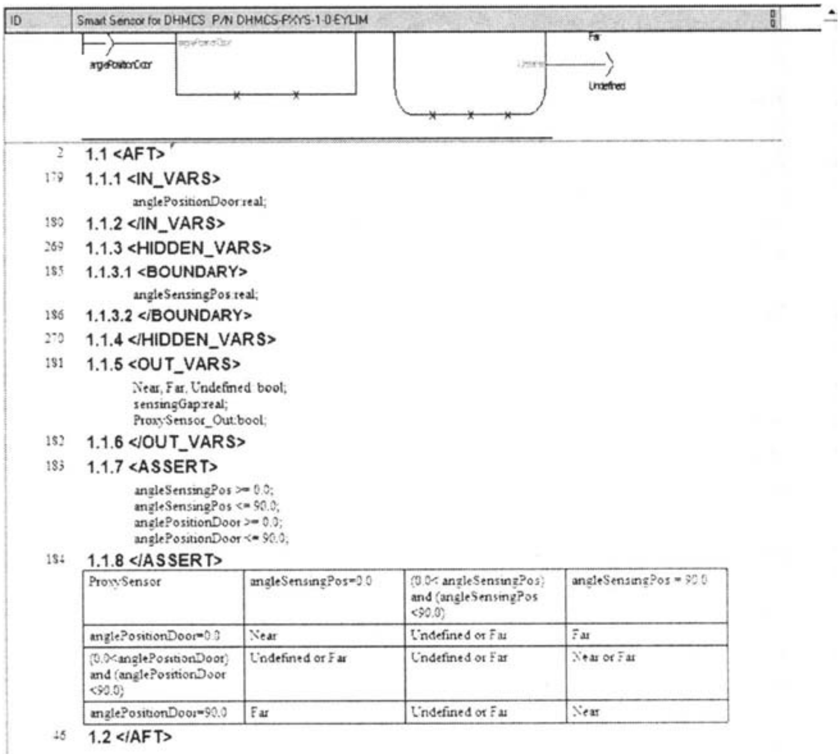
2.5 Requirements Extensions

Within AMBERS the objective of the requirement process is to defining formalised total functions, i.e. functions that are completely defined over the engineering scope using Parnas/SCR tables' semantics. To achieve this Parnas tables are embedded in the specification in a form that allows automatic parsing. Broadly a requirements specification in AMBERS consists of an informal textual part followed by a formal part, known as an 'Assertive Function Table' (AFT) defining the function formally.

The AFT is made up of the function signature and a table. The function signature consists of:

- Input variable clause – This clause lists all inputs and their types.
- Hidden variable clause – Hidden variables are a SCADE concept that allows inputs to be hidden, thereby removing clutter from diagrams. Within AMBERS these variables are used to parameterise functions to allow reuse.
- Output variable clause – This clause lists all outputs and their types.
- Assertion list clause – This clause allows a list of assertions to be given which provide more information about the scope of the function.
- Assertive Function Table clause – This is a Parnas table providing a specification for the function.

Figure 7 shows an example captured in DOORS.



ID | Smart Sensor for DHMCS P/N DHMCS-PXYS-1-0-EYLIM

2 1.1 <AFT>

179 1.1.1 <IN_VARS>

anglePositionDoor:real;

180 1.1.2 </IN_VARS>

269 1.1.3 <HIDDEN_VARS>

185 1.1.3.1 <BOUNDARY>

angleSensingPos:real;

186 1.1.3.2 </BOUNDARY>

270 1.1.4 </HIDDEN_VARS>

191 1.1.5 <OUT_VARS>

Near, Far, Undefined: bool;
sensingGap:real;
ProxySensor_Out:bool;

192 1.1.6 </OUT_VARS>

193 1.1.7 <ASSERT>

angleSensingPos >= 0.0;
angleSensingPos <= 90.0;
anglePositionDoor >= 0.0;
anglePositionDoor <= 90.0;

194 1.1.8 </ASSERT>

ProxySensor	angleSensingPos=0.0	(0.0< angleSensingPos) and (angleSensingPos <90.0)	angleSensingPos = 90.0
anglePositionDoor=0.0	Near	Undefined or Far	Far
(0.0<anglePositionDoor) and (anglePositionDoor <90.0)	Undefined or Far	Undefined or Far	Near or Far
anglePositionDoor=90.0	Far	Undefined or Far	Near

45 1.2 </AFT>

Figure 7: Example of Assertive Function Table

The example of Figure 7 is a table representing a door position sensing function. The function has one input, 'anglePositionDoor', which is of type 'real', one hidden variable (listed under the '<Boundary> ' header) called 'angleSensingPos' of type real. It has five outputs and four assertions.

The four assertions (between <ASSERT> and </ASSERT>) are listed on separate lines, but are combined and apply across the table. Thus any proof of compliance to the table is carried out under the assertion that 'angleSensingPos' may only take a value from 0.0 to 90.0, and similarly for 'anglePositionDoor'.

The table below the assertions has the following form:

<Name of Function>	<PreCondition 1>	<PreCondition 2>	<PreCondition 3>
<PostCondition 1>	<Assertion 1_1>	<Assertion 1_2>	<Assertion 1_3>
<PostCondition 2>	<Assertion 2_1>	<Assertion 2_2>	<Assertion 2_3>
<PostCondition 3>	<Assertion 3_1>	<Assertion 3_2>	<Assertion 3_3>

In this table the first row and the first column describe conditions that we are interested in when defining respective system behaviours. The rest of the table defines the behaviours that we expect under the conditions defined in the appropriate column and row.

This table provides a specification for the function as a conjunction of implications. In long hand the table specifies that the following should be true of an implementation:

(<Precondition_1> ∧ <Postcondition_1> → <Assertion 1_1>) ∧
(<Precondition_1> ∧ <Postcondition_2> → <Assertion 2_1>) ∧
(<Precondition_1> ∧ <Postcondition_3> → <Assertion 3_1>) ∧
. . .
(<Precondition_3> ∧ <Postcondition_3> → <Assertion 3_3>)

The table is built under the Parnas conditions of conditions pair-wise disjunction and completeness [Parnas D.L. (1992) and (1993)].

Where '∧' is the 'and' operator and '→' is the 'implication' operator. Informally the statement can be read as "If <Precondition_1> and <Postcondition_1> are true then <Assertion_1_1> should also be true." Operators other than the standard material implication could be contemplated such as implication involving temporal conditions: e.g. Implies-after N-steps.

When the table is considered for proof the assertion list is also taken into account as statements of fact about the environment that are not checked by the proof, but are added to ensure that engineering boundaries are taken into account.

The benefit of this approach is that the requirements based engineer is able to think about the constraints to place on his function in an unambiguous manner, but is able to express them in a fashion he/she is familiar with.

2.6 Bridging Between Requirements and the Model

So far in this description the requirements and the model have remained as separate entities. We create a bridge between them by translating the formal part of the requirements specification into the model. This will allow consistency checks to be carried out between the requirements and the model.

As discussed earlier SCADE has a built in proof capability. With some customisation it has been possible to automatically enter the formal part of the requirements into the SCADE model, and then to use the proof capability to automatically generate proofs from the model. In this way we are able to automatically provide a mechanism to show compliance of a model to the formal part of the requirements specification.

The steps to achieve this are:

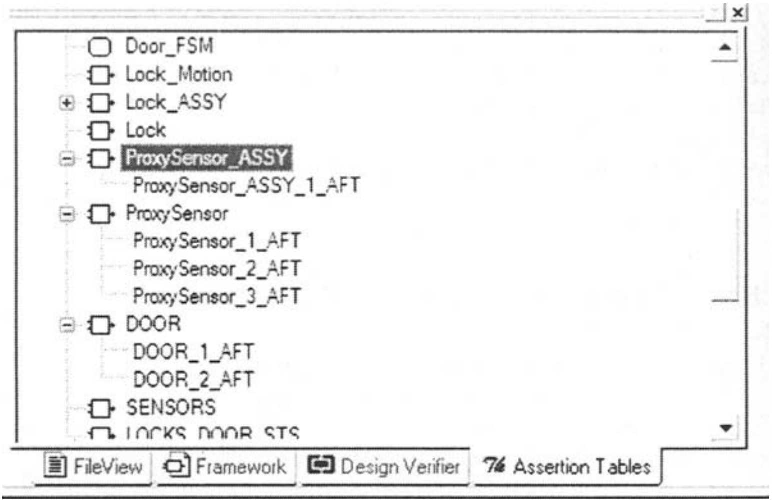
1. Import the requirements AFT into the model that is being used to validate the requirements.

2. From that model generate a 'proof' model where the proofs are to be carried out.
3. Generate proofs.
4. Analyse failed proofs to identify whether the error is in the model or in the requirements.
5. If necessary return to step 1 until all proofs have been proved.

Steps 1 to 3 are detailed in the sections below.

2.6.1 Step 1 – Importing the Formal Requirements

The tools developed to support AMBERS parse a DOORS requirements module, and for each AFT creates supplementary information in the SCADE model that holds the formal information. As discussed previously SCADE models a system as a set of nodes. The import tool analyses the AFT to identify the node that the table should be applied to. It is possible for one node to have more than one AFT table associated to it. This is allowed, as it may be beneficial for an AFT to address different aspects of the same area of functionality. Figure 8 shows a view within SCADE of the AFTs and the nodes they are associated with.



- Door_FSM
- Lock_Motion
- Lock_ASSY
- Lock
- ProxySensor_ASSY
  - ProxySensor_ASSY_1_AFT
- ProxySensor
  - ProxySensor_1_AFT
  - ProxySensor_2_AFT
  - ProxySensor_3_AFT
- DOOR
  - DOOR_1_AFT
  - DOOR_2_AFT
- SENSORS
- LOCKS_DOOR_STS

Figure 8: AFTs Associated with SCADE Nodes

These AFTs may be edited within SCADE to allow errors to be corrected quickly. Figure 9 shows the SCADE screen that allows the AFT to be viewed and edited.

At this stage the AFT has no implications for the model, it is merely a set of additional information available to the model-based engineer.

2.6.2 Step 2 – Generating the Proof Model

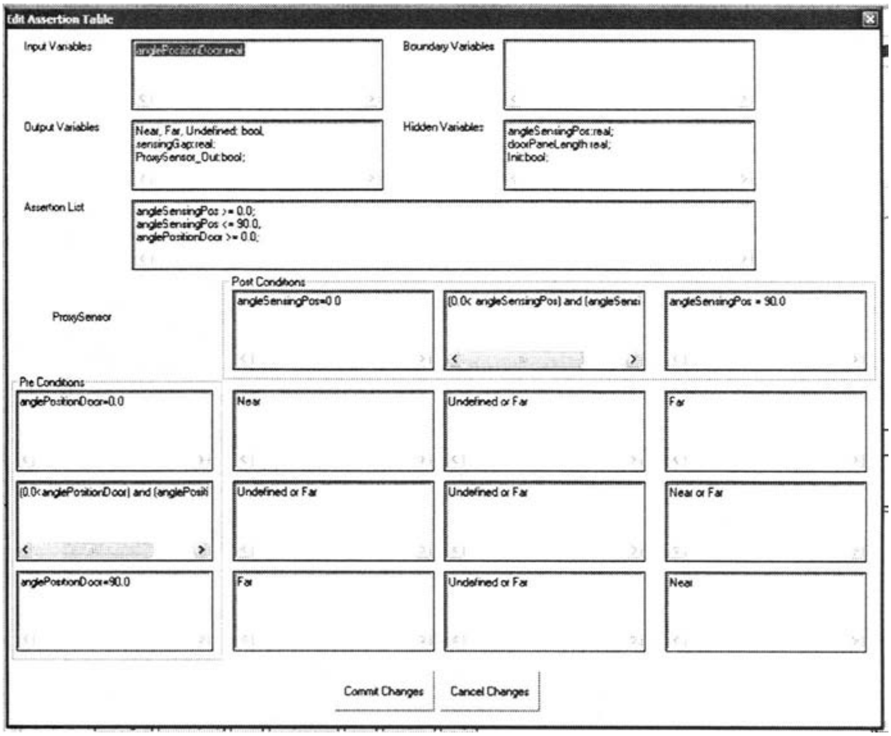
Generation of the proof model is an entirely automatic operation. A new model is created that:

- Contains the AFT translated into a set of proof objectives.
- Imports the original functional model as a read only library.

As discussed earlier SCADE allows proof objectives to be encoded as a standard SCADE node. The translation therefore translates each cell of the table into a separate node to act as a proof objective. Figure 10 shows an example of the proof nodes generated.

An example of the actual SCADE node generated from the cell in row 2 and column 2 of Figure 7 is shown in Figure 11. In this figure you can see that the assertion list has been converted to a set of SCADE assertions :

```
assert angleSensingPos >= 0.0 ;
assert angleSensingPos <= 90.0 ;
assert anglePositionDoor >= 0.0 ;
assert anglePositionDoor <= 90.0 ;
```



Input Variables: anglePositionDoor:real

Boundary Variables:

Output Variables: Near, Far, Undefined: bool,
sensingGap:real;
ProxySensor_Out:bool;

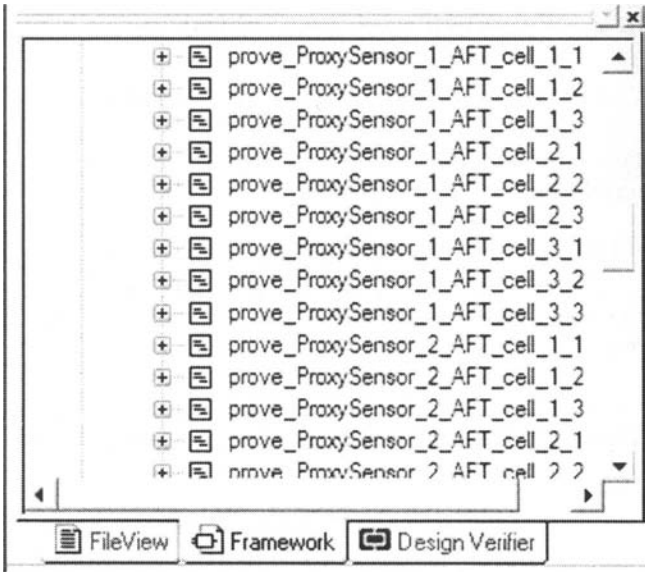
Hidden Variables: angleSensingPos:real;
doorPanelLength:real;
Init:bool;

Assertion List:

```
angleSensingPos >= 0.0;
angleSensingPos <= 90.0,
anglePositionDoor >= 0.0;
```

Pre Conditions	Post Conditions		
ProxySensor	angleSensingPos=0.0	(0.0< angleSensingPos) and (angleSens	angleSensingPos = 90.0
anglePositionDoor=0.0	Near	Undefined or Far	Far
(0.0<anglePositionDoor) and (anglePosit	Undefined or Far	Undefined or Far	Near or Far
anglePositionDoor=90.0	Far	Undefined or Far	Near

Figure 9: SCADE Representation of an AFT



prove_ProxySensor_1_AFT_cell_1_1
prove_ProxySensor_1_AFT_cell_1_2
prove_ProxySensor_1_AFT_cell_1_3
prove_ProxySensor_1_AFT_cell_2_1
prove_ProxySensor_1_AFT_cell_2_2
prove_ProxySensor_1_AFT_cell_2_3
prove_ProxySensor_1_AFT_cell_3_1
prove_ProxySensor_1_AFT_cell_3_2
prove_ProxySensor_1_AFT_cell_3_3
prove_ProxySensor_2_AFT_cell_1_1
prove_ProxySensor_2_AFT_cell_1_2
prove_ProxySensor_2_AFT_cell_1_3
prove_ProxySensor_2_AFT_cell_2_1
prove_ProxySensor_2_AFT_cell_2_2

Figure 10: Generated Proof Nodes

The node under proof is then invoked to populate the variables that make up the proof:

```
Near , Far , Undefined , sensingGap =
ProxySensor(anglePositionDoor , angleSensingPos ,
doorPaneLength , Init , guaranteedActuation ,
guaranteedDeActuation) ;
```

The table cell and associated conditions are encoded:

```
precond_1 = angleSensingPos = 0.0 ;
postcond_1 = anglePositionDoor = 0.0 ;
cell_1_1 = Near ;
```

The proof is then set up and placed in an output variable:

```
prove_ProxySensor_1_AFT_cell_1_1_OUT = Implies((precond_1 and
postcond_1) , cell_1_1) ;
```

2.6.3 Step 3 – Generate Proofs

Proof generation is a simple activity. All that is necessary is to select the property that we wish to prove (in this case prove_ProxySensor_1_AFT_cell_1_1) and select the analyze option.

The results are reported, amongst other cases, as either proved (valid), contradictory or falsifiable. If the proof is falsifiable then SCADE generates a simulation scenario that can be executed to show the erroneous behaviour.
Figure **12** gives an example of the script created.

```
node prove_ProxySensor_1_AFT_cell_1_1(
   anglePositionDoor : real ;
   angleSensingPos : real ;
   guaranteedActuation : real ;
   guaranteedDeActuation : real ;
   doorPaneLength : real ;
   Init : bool)
   returns (
   prove_ProxySensor_1_AFT_cell_1_1_OUT : bool) ;
var
Far : bool ;
Near : bool ;
ProxySensor_Out : bool ;
Undefined : bool ;
cell_1_1 : bool ;
postcond_1 : bool ;
precond_1 : bool ;
sensingGap : real ;

let equa eq_prove_ProxySensor_1_AFT_cell_1_1[ , ]
   assert angleSensingPos >= 0.0 ;
   assert angleSensingPos <= 90.0 ;
   assert anglePositionDoor >= 0.0 ;
   assert anglePositionDoor <= 90.0 ;
    Near    ,    Far    ,    Undefined    ,    sensingGap    =
ProxySensor(anglePositionDoor    ,    angleSensingPos    ,
doorPaneLength    ,    Init    ,    guaranteedActuation    ,
guaranteedDeActuation) ;
    precond_1 = angleSensingPos = 0.0 ;
    postcond_1 = anglePositionDoor = 0.0 ;
```

Figure 11: Generated SCADE Proof Objective

```
# ---------------------------------------------------------------
# Simulation scenario file for SCADE Simulator
# Task:
prove_ProxySensor_1_AFT_cell_1_1.prove_ProxySensor_1_AFT_cell_1_1_OUT
# Model: AMBERed_Observer
# Node:  prove_ProxySensor_1_AFT_cell_1_1
# ---------------------------------------------------------------

SSM::set anglePositionDoor 0.0000000000
SSM::set angleSensingPos 0.0000000000
SSM::set guaranteedActuation $SSM::default_real
SSM::set guaranteedDeActuation $SSM::default_real
SSM::set doorPaneLength $SSM::default_real
SSM::set Init $SSM::default_bool

SSM::cycle

SSM::set anglePositionDoor 0.0000000000
SSM::set angleSensingPos 0.0000000000
SSM::set guaranteedActuation -1.0000000000
SSM::set guaranteedDeActuation 0.0000000000
SSM::set doorPaneLength 0.0000000000
SSM::set Init f
```

Figure 12: Example SCADE Simulation Script

2.7 Modelling Process Considerations

Within the ACE programme the systems lifecycle functions are decomposed through five layers from the Top-Level Function (TLF – Layer – 0) to the Leaf-Level Function s (LLF – Layer 4). Intermediate Sub-Level Functions (SLF –Layer 1, 2 3) are used to manage the size of decomposition steps. Figure 13 shows the steps of the lifecycle. The line spiralling from the left to the right and down shows the typical floe of activity through the lifecycle.

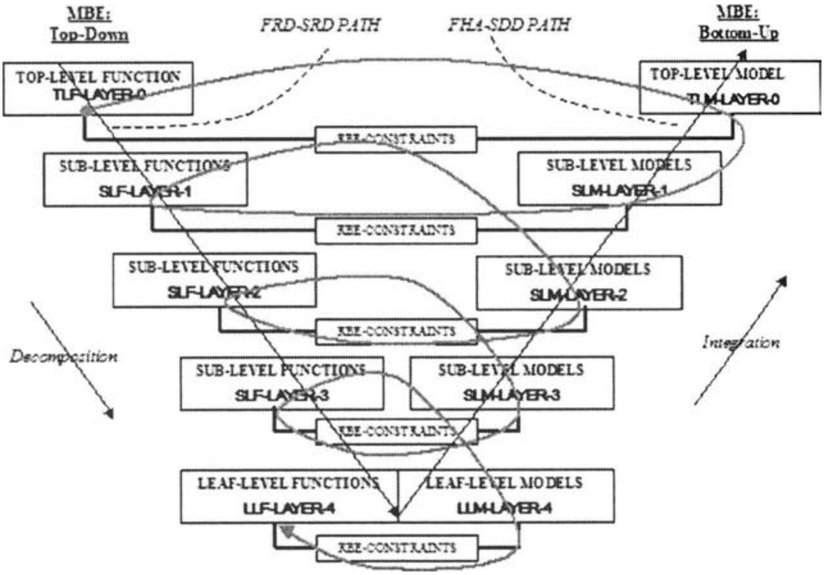


Figure 13: ACE System Lifecycle

At each level, functions are defined with associated AFTs, and models are defined to validate the functions. The functions are analysed based upon their abstract signatures at their respective layers prior to proceeding to their further decomposition. Models defined at the same level are subject to analysis using the related AFTs to show compliance to the requirements.

Once leaf layer functions and models have been defined then a process of integration is followed where models are validated at their respective layers before integration at the next higher layer using simulation, test and proof.

3 Door Health Monitored and Control System (DHMCS) –AMBERS Demonstrator Project

In order to demonstrate and validate the AMBERS approach a pilot study was conducted. The aim of the pilot study was to reach the following objectives:

- Establish the practical connection between an RBE interpretation and an MBE representation, and vice versa.
- Show the joint processes in action for simulation, scenario based testing and analysis (proof).
- Show in practice (in working models) the issues of:
  - Validation: MBE faithfulness, RBE feasibility;
  - Verification: MBE verifiability, RBE consistency;
- Set-up a technology prototype as a proof of concept involving:
  - DOORS Assertive Function Tables (AFT);
  - SCADE/Design Verifier Observer project based upon library of models.

3.1 Description of System

The 'DHMCS' is a 'simplified' controlled flapper-door that allows direction of a high-energy hot or cold airflow. The door provides a mix of air within a given duct. The door pane may be locked in the up or in the down position. Sensors provide the door moving and locking positions. A selector valve controlled by software allows operation of the hydraulic actuator to move the door's arm.

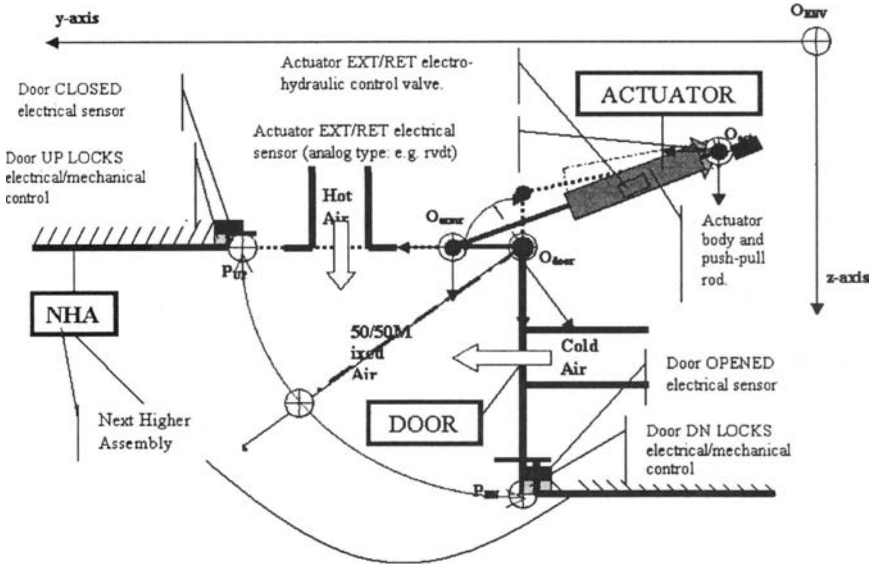

y-axis
O
Actuator EXT/RET electro-hydraulic control valve.
Door CLOSED electrical sensor
ACTUATOR
Actuator EXT/RET electrical sensor (analog type: e.g. rvdt)
Door UP LOCKS electrical/mechanical control
Hot Air
O
O
O
Actuator body and push-pull rod.
z-axis
P
NHA
50/50Mixed Air
Cold Air
Door OPENED electrical sensor
Next Higher Assembly
DOOR
Door DN LOCKS electrical/mechanical control
P

Figure 14: Schematic of the DHMCS System

3.2 Functional Decomposition Down the Layers of ACE

A sketched functional decomposition is used as the basis for analysis, consisting of the following items:

- Actuator sub-function;
- Door sub-function
- Control and Monitoring sub-function
    - Control and Monitoring
    - Controls and Displays Interface

At layer 1 of the DHMCS the system is broken into an overall DHMCS node and an interface node. The interface node is defined to allow implementation of an appropriate simulation user interface for the detailed models. Figure 15 shows the graphical function block break down at this level.

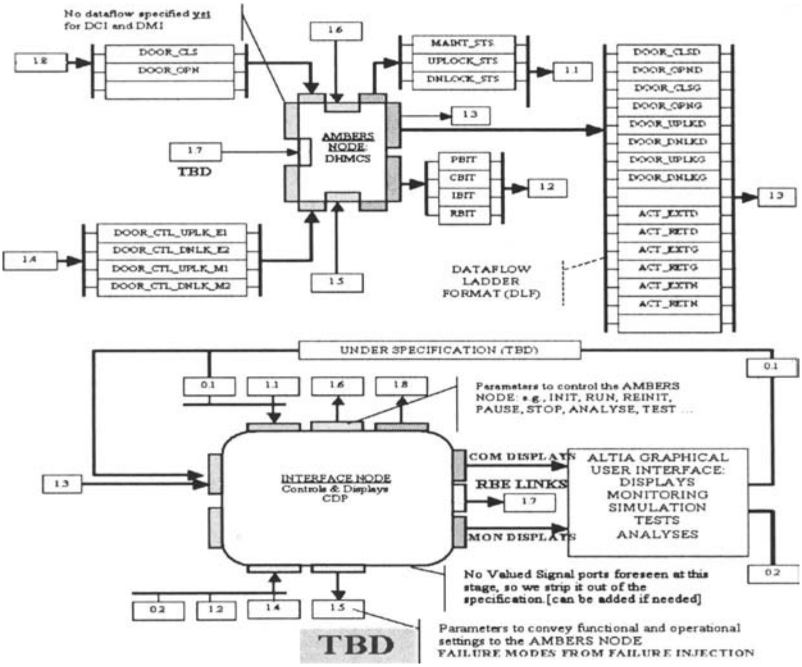

No dataflow specified yet for DCI and DMI
1.8
DOOR_CLS
DOOR_OPN
1.6
MAINT_STS
UPLOCK_STS
DNLOCK_STS
1.1
DOOR_CLSD
DOOR_OPND
DOOR_CLSG
DOOR_OPNG
DOOR_UPLKD
DOOR_DNLKD
DOOR_UPLKG
DOOR_DNLKG
ACT_EXTD
ACT_RETD
ACT_EXTG
ACT_RETG
ACT_EXTM
ACT_RETM
1.3
1.3
AMBERS NODE: DHMCS
1.7
TBD
PBIT
CBIT
IBIT
RBIT
1.2
1.4
DOOR_CTL_UPLK_E1
DOOR_CTL_DNLK_E2
DOOR_CTL_UPLK_M1
DOOR_CTL_DNLK_M2
1.5
DATAFLOW LADDER FORMAT (DLF)
UNDER SPECIFICATION (TBD)
0.1
0.1
1.1
1.6
1.8
Parameters to control the AMBERS NODE: e.g., INIT, RUN, REINIT, PAUSE, STOP, ANALYSE, TEST ...
1.3
INTERFACE NODE
Controls & Displays
CDP
COM DISPLAYS
RBE LINKS
1.7
MON DISPLAYS
ALTIA GRAPHICAL USER INTERFACE: DISPLAYS MONITORING SIMULATION TESTS ANALYSES
No Valued Signal ports foreseen at this stage, so we strip it out of the specification.[can be added if needed]
0.2
0.2
1.2
1.4
1.5
TBD
Parameters to convey functional and operational settings to the AMBERS NODE
FAILURE MODES FROM FAILURE INJECTION

Figure 15: Layer_1 Decomposition of DHMCS

At Layer_2 the DHMCS block is decomposed into an actuator and door component, as shown in Figure 16.

The flow labels (1.7, 2.7 etc.) relate to data dictionary entries. Flows with a '2' as the first number correspond to Layer_2 which further decomposes Layer-1: e.g. '2.5' describes a dataflow at Layer-2 whilst '1.7' describes a dataflow coming from Layer_1 down to the Layer-2 detailing here the parameterisation of the 'Actuator'.

At the base level (Layer_4 of the ACE lifecycle) the decomposition is mapped concretely to a SCADE model. The SCADE model implementation follows a one-to-one map with the AMBERS notation as illustrated in Figure 17 for the DHMCS.Door with Locks, Sensors and the door assembly (Door_Assy).

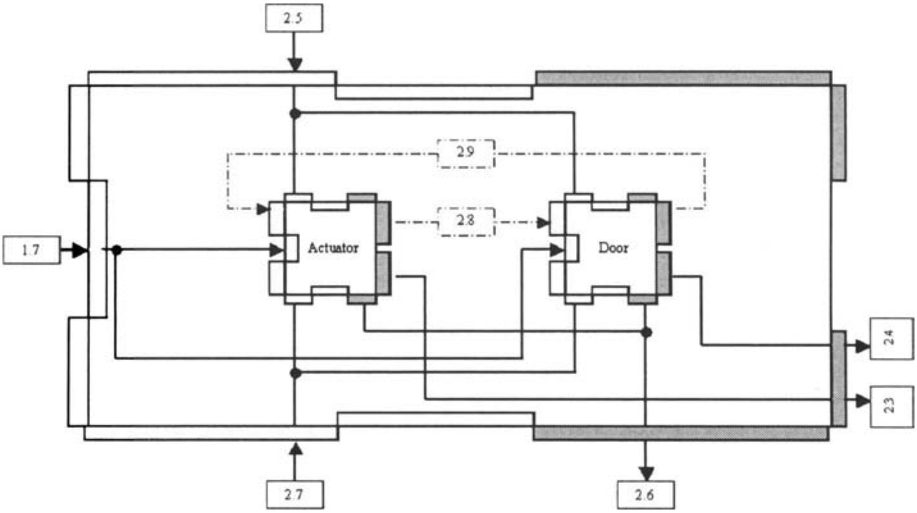


Figure 16: Layer_2 Decomposition

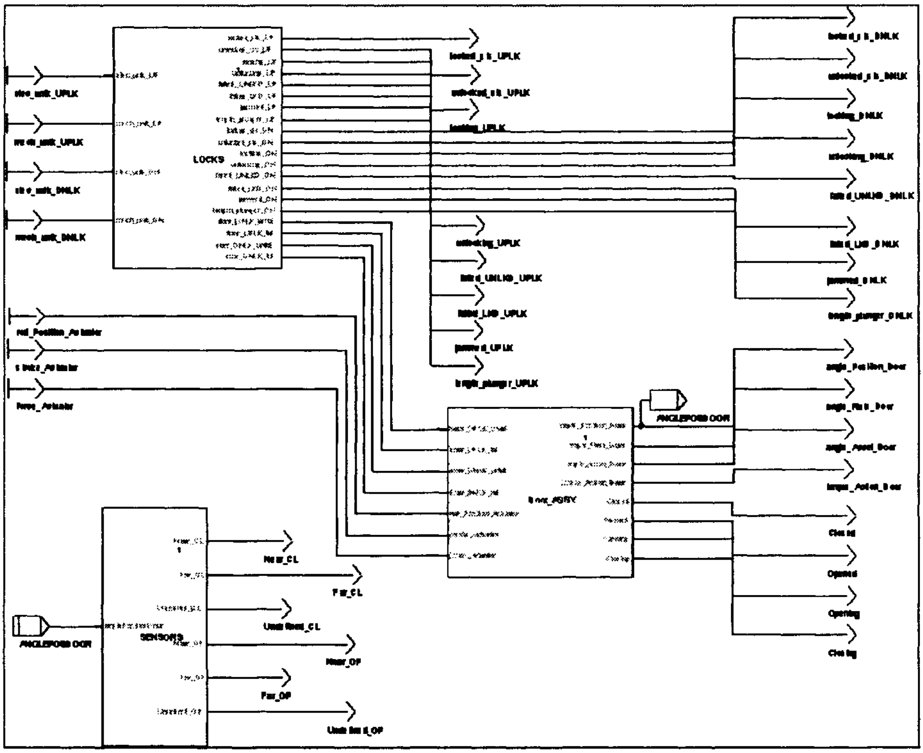


Figure 17: Concrete Mapping of AMBERS to SCADE Design

3.3 Validation and Analysis

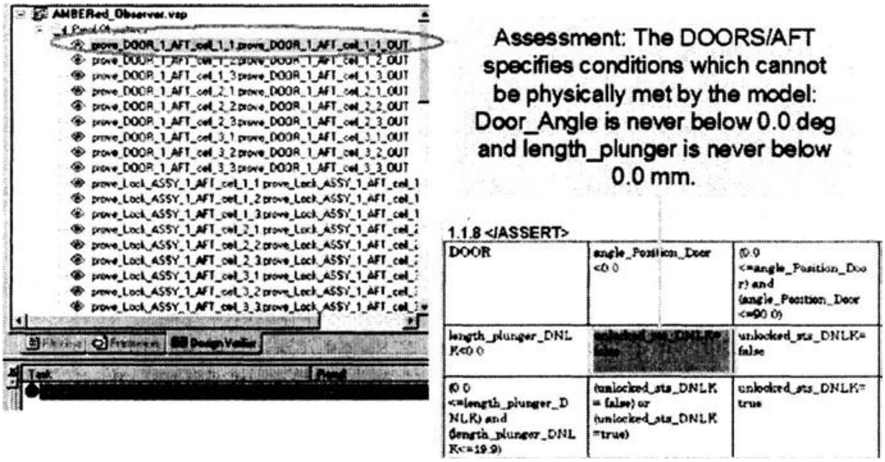
Once the model has been mapped into a SCADE design it is possible to run simulations and carry out proofs to validate the requirements. Within AMBERS validation by simulation is carried out before attempting to prove compliance with proof objectives. This is carried out to ensure that the model is broadly correct, and to identify any stability issues with the model.

Once the model is judged to be broadly correct with respect to simulation behaviour then it is subject to proof against the formal part of the written specification. There are a number of possible results of running the proof;

1. The proof objective is proved – in which case there is good confidence that the model and requirements are consistent.
2. The proof times out – in this case the proof has reached a predetermined limit, for example number of internal proof cycles, and the result is indeterminate. The proof could continue for longer and reach a conclusion, or it could continue for an arbitrary length of time. In this case it is helpful to return to the formal part of the requirements to identify

whether additional information could be provided to assist the proof, or a decision could be made to provide confidence in the property purely through targeted simulation and tests.

3. The proof objective is shown to be false for the system modelled – this may be because the model is incorrect, or because the requirements are defined either too tightly (over-specification) or too loosely (under-specification). At this point it is necessary for the model-based engineering and the requirements-based engineering specialists to review the simulation generated by SCADE and discuss whether the issue arises from the requirements or the model. Figure 18 shows an example of over-specification.



```
AMBERed_Observer.vsp
  Proof Objectives
    prove_DOOR_1_AFT_cell_1_1 prove_DOOR_1_AFT_cell_1_1_OUT
    prove_DOOR_1_AFT_cell_1_2 prove_DOOR_1_AFT_cell_1_2_OUT
    prove_DOOR_1_AFT_cell_1_3 prove_DOOR_1_AFT_cell_1_3_OUT
    prove_DOOR_1_AFT_cell_2_1 prove_DOOR_1_AFT_cell_2_1_OUT
    prove_DOOR_1_AFT_cell_2_2 prove_DOOR_1_AFT_cell_2_2_OUT
    prove_DOOR_1_AFT_cell_2_3 prove_DOOR_1_AFT_cell_2_3_OUT
    prove_DOOR_1_AFT_cell_3_1 prove_DOOR_1_AFT_cell_3_1_OUT
    prove_DOOR_1_AFT_cell_3_2 prove_DOOR_1_AFT_cell_3_2_OUT
    prove_DOOR_1_AFT_cell_3_3 prove_DOOR_1_AFT_cell_3_3_OUT
    prove_Lock_ASSY_1_AFT_cell_1_1 prove_Lock_ASSY_1_AFT_cell_1
    prove_Lock_ASSY_1_AFT_cell_1_2 prove_Lock_ASSY_1_AFT_cell_1
    prove_Lock_ASSY_1_AFT_cell_1_3 prove_Lock_ASSY_1_AFT_cell_1
    prove_Lock_ASSY_1_AFT_cell_2_1 prove_Lock_ASSY_1_AFT_cell_2
    prove_Lock_ASSY_1_AFT_cell_2_2 prove_Lock_ASSY_1_AFT_cell_2
    prove_Lock_ASSY_1_AFT_cell_2_3 prove_Lock_ASSY_1_AFT_cell_2
    prove_Lock_ASSY_1_AFT_cell_3_1 prove_Lock_ASSY_1_AFT_cell_3
    prove_Lock_ASSY_1_AFT_cell_3_2 prove_Lock_ASSY_1_AFT_cell_3
    prove_Lock_ASSY_1_AFT_cell_3_3 prove_Lock_ASSY_1_AFT_cell_3
```

Assessment: The DOORS/AFT specifies conditions which cannot be physically met by the model: Door_Angle is never below 0.0 deg and length_plunger is never below 0.0 mm.

1.1.8 </ASSERT>

DOOR	angle_Position_Door <0.0	(0.0 <=angle_Position_Door) and (angle_Position_Door <=90.0)
length_plunger_DNLK<0.0	unlocked_sts_DNLK= false	unlocked_sts_DNLK= false
(0.0 <=length_plunger_DNLK) and (length_plunger_DNLK<=19.9)	(unlocked_sts_DNLK = false) or (unlocked_sts_DNLK =true)	unlocked_sts_DNLK= true

Figure 18 : Identification of Over-specification in the Requirements

4 Conclusion

The AMBERS demonstrator project lasted six months and has achieved the objectives that were assigned to it. A technology prototype has been developed to bridge between requirements and models. The work on the DHMCS example has shown:

- Establishment of a practical connection between RBE interpretation and MBE representations and vice versa.
- A joint process was demonstrated integrating simulation, scenario based tests and analysis (proof)
- The issues of:
    - Validation: MBE faithfulness, RBE feasibility;
    - Verification: MBE verifiability, RBE consistency;
- Viability of a technology based on COTS tools based upon:
    - DOORS Assertive Function Tables (AFT);

  - SCADE/Design Verifier Observer project based upon library of models.

The AMBERS process allows definition of an unambiguous framework where systems design and system integrity specialists can technically and scientifically discuss the issues highlighted by both sides. The Assertive Function Tables based upon Parnas/SCR tables successfully support of that platform and provide straightforward build-up for simulations and test case scenarios.

The overall embedding of a formal part in informal specification text in a DOORS formal module is simple and requires a limited training, which may be enhanced by adapting a Graphical user Interface to intuitively pre-fill the formalised part.

4.1 Further Work

Some issues were highlighted during the demonstration project that require further work:

- Proof across non-linear mathematical functions: Current proof technology does not handle non-linear functions effectively. In most cases a linear approximation of a non-linear function has to be provided to the proof tool. This can affect the validity of proofs and require additional simulation to give confidence that the approximation holds.
- The current prototype requires a single assertive function table be associated with a full signature. It would be more natural, where more than one AFT is required for a single node, to associate many AFTs to a single signature.
- Correct and complete parsing of the formalised part in order to ascertain a correct translation into the SCADE environment;
- Extension of the work done on the functional dependency and behavioural dependency models to extract a causal dependency model that will be the basis for dependability and safety analyses: i.e. Fault Tree, Markov networks, FHA, SSA for reliability and availability assessments;
- Industrialisation of the RBE-MBE process in a networked database of constrained models. This would involve co-related configuration control of models and requirements within a sound AMBERS-based engineering process. To achieve this a co-operative framework with equipment suppliers and with different engineering sites would be required.

The authors of this paper wish to extend their warm thanks to Mr Richard Crisp from Telelogic for his very helpful support in handling the DOORS extensions. Mr Mick Dunne, John Cahill and Mike Yates from Airbus UK were instrumental in supporting this project within the Integrated System Engineering Framework (ISEF) and the Landing Gear R&T realm. Finally the AMBERed six months project would not have been launched without the effective support of Mr Sanjiv Sharma, Martin Dobson and Dr Benita Lawrence who are respectively Landing

Gear Modelling & Simulation, Safety group leaders and Head of Performance Integrity, at Airbus Industrie.

5 References

Parnas D.L,(1993). 'Predicate Logic for Software Engineering'. IEEE Transactions on Software Engineering, VOL 19. N0 9, September 1993

Parnas D.L. (1992). 'Tabular Representations of Relations'. Telecommunications Research Institute of Ontario. Communications Research Laboratory. Report CRL N0 260, October 1992.

Fortes da Cruz. M.Au, (2001). 'Building Systems as Transformers'. Irish Workshop On Formal Methods, IWFM'01, Dublin, May 2001.

Heitmeyer C.L., Jeffords R.D. and Labaw B.G. (1996). 'Automated Consistency Checking of Requirements Specifications'. ACM Transactions on Software Engineering and Methodology, Vol. 5, No. 3, July 1996, pp 231-261

Halbwachs N., Caspi P., Raymond P. and Pilaud D. (1991). 'The Synchronous dataflow programming language LUSTRE'. Proceedings of the IEEE, Vol 79, Issue 9, September 1991, pp 1305-1320.

David R. and Alla H. (editors) (1992). 'Petri Nets and Grafcet: Tools for Modelling Discrete Event Systems'. Prentice Hall, New York 1992.

Formalising C and C++ for Use in High Integrity Systems

C M O'Halloran, C H Pygott
QTIM, QinetiQ
Malvern, UK

Abstract

UK MoD has long been an advocate of the use of mathematically formal verification in software for safety critical applications. In the past this has been focused on the SPARK Ada subset, but it is increasingly becoming difficult to find suppliers willing or capable of delivering Ada programs. Instead, there is a pressure to use more commercially attractive languages, such as C and C++. In order to maintain the high levels of confidence necessary for critical applications, this means being able to formally reason about these 'new' languages.

This paper covers two related programmes that are developing formal semantics for restricted subsets of C and C++ respectively. It will also consider how the formal semantics will be exploited in a verification environment.

1 Introduction

UK MoD has long been an advocate of the use of mathematically formal verification in software for safety critical applications. In the past this has been focused on the SPARK Ada subset, but it is increasingly becoming difficult to find suppliers willing or capable of delivering Ada programs. Instead, there is a pressure to use more commercially attractive languages, such as C and C++. In order to maintain the high levels of confidence necessary for critical applications, this means being able to formally reason about these 'new' languages.

This paper covers two related programmes that are developing formal semantics for restricted subsets of C and C++ respectively. It will also consider how the formal semantics will be exploited in a verification environment.

There has been a shift in the safety critical development community following the decision by UK MoD to permit the procurement of safety critical software developed in languages other than Ada. Attention has moved towards more widely-used languages such as C and C++, and a need has arisen for practices and tools that can support safety related software development in these languages.

Research into this aspect of C++ development is currently some two to three years behind that on C, so this paper starts by discussing how the requirements for a high integrity subset of C++ are being defined as a precursor to the development of a formal subset semantics.

Hopefully by the time of the presentation, more concrete progress of C++ fomal semantics can be presented.

2 Developing High Integrity Guidance for C++

Lockheed Martin's JSF++

Background to JSF++

The Joint Strike Fighter (JSF) is a major 'next generation' military aircraft programme being jointly funded by the UK and US governments, and being led by Lockheed Martin.

A decision that some of the avionics would be developed in C++ was taken early on, reflecting the availability of C++ programmers and the comparative dearth of Ada programmers (the MoD and DoD's preferred programming language in the 80s and early 90s). This included development of some safety related functionality.

This proposed use of C++ for safety related applications raised concerns in both the UK and US software safety communities. In mitigation, Lockheed Martin developed a coding standard aimed at addressing the perceived weaknesses in the language (insofar as its uses in safety applications were concerned). This coding standard became known as "JSF++" (JSF 2005).

The strategy of JSF++ is illustrated in Figure 1. The starting point for JSF++ was MISRA C (MISRA 2004), a well-established and peer reviewed coding standard produced by the Motor Industry Software Reliability Association (MISRA), which addresses the use of C in safety related systems and defines a strict subset of C. C++ is not actually a superset of C, as there are features of C (e.g. function use before declaration) that are prohibited in C++. However, those features of C not included in C++ are also disallowed by MISRA C, and consequently C++ is a strict superset of MISRA C. Where behaviour of the C subset of C++ was limited by MISRA C, the same or more strict limitations would be applied by JSF++.

For those features of C++ that are outside the scope of MISRA C, Lockheed Martin sought expertise from recognised industry and academic C++ experts,[1] to create similar controls.

[1] These included Bjarne Stroustrup, the creator of C++.

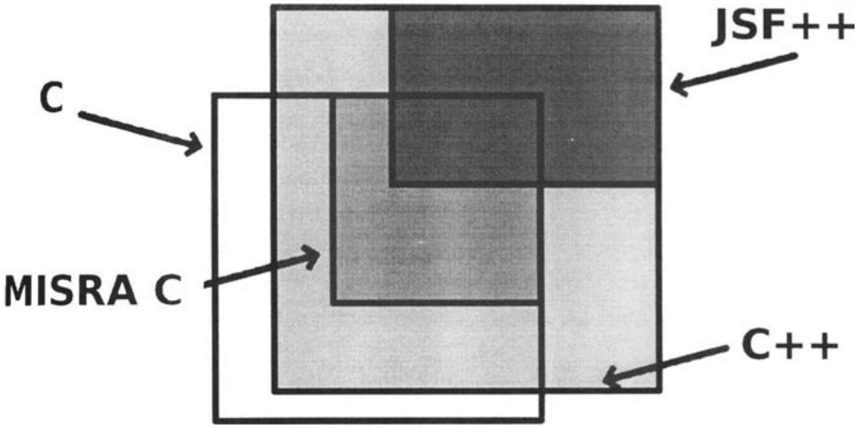


Figure 1: The relationships between C, C++, MISRA C and JSF++

Some undesirable behaviours, such as buffer overrun, cannot be addressed by subsetting the language. Whilst one strategy would be to place a verification obligation on the developer, to show that buffer overrun could never occur, the approach adopted by JSF++ is to define a series of container classes that code is required to use for all array objects. These containers can trap 'out of bounds' access, and so prevent unpredictable behaviour. In effect, they impose the Ada array access model onto C++.

Review for MoD

In order to address their duty of care under the UK Health and Safety at Work Act and the concerns being raised about the use of C++ in safety related applications in some parts of the UK software industry, MoD asked QinetiQ to perform an assessment of JSF++.

Before starting the review of the proposed JSF coding standard, the ISO language definition (C++ 2003) and some eight existing C++ coding guides were reviewed to identify issues that might be expected to be addressed. In particular, the language definition was searched for behaviour that was:

- *Unspecified*: that is "behaviour, for a well-formed program construct and correct data, that depends on the implementation. The implementation is not required to document which behaviour occurs". For example the order of evaluation of sub-expressions in a statement.
- *Undefined*: that is "behaviour, such as might arise upon use of an erroneous program construct or erroneous data, for which the language standard imposes no requirements". For example, the effect of dereferencing a NULL pointer.
- *Implementation defined*: that is "behaviour, for a well-formed program construct and correct data, that depends on the implementation and that each implementation shall document". For example, the size of primitive types, such as int.
- *Indeterminate*: that is, behaviour not defined in the C++ language standard due to negative or missing statements. For example, some language statements define that a construct shall not use a particular feature. It is therefore left indeterminate what would happen if such a construct did use that particular feature.
- "*Behaviour that requires no diagnostics*" is a feature of the language which does not follow the required or expected 'rules' but for which the language standard states that no diagnostic information is required to be given to the user. For example, a class virtual member function may have a definition provided whilst also being declared pure (forcing a definition to be provided by any derived class).

Not all of these represent a safety issue. In particular, implementation defined behaviours and behaviours that require no diagnostics are generally benign (in a given environment), but may cause portability or long term maintenance issues. The results of the analysis are shown in Table 1.

Type of issue	Number of issues
Unspecified behaviours	50
Undefined behaviours	106
Implementation defined	81
Indeterminate	5
Behaviours that require no diagnostics	18
Issues from other sources	66

Table 1: Identified C++ issues

Its also worth noting that coding standards exist for reasons other than to avoid unspecified etc. behaviours. Coding standards typically also aim:

- To improve clarity for review and maintenance: e.g. not allowing variable names to differ by just replacing lower case letter 'l' by the digit '1'. Note that this is a human perception issue. The compiler has no problem distinguishing "countl" from "count1", but a reviewer or maintainer is likely to think that they refer to the same object. Arguably, the 18 issues that are classified as 'behaviour that requires no diagnostic' (Schofield and Pygott, 2006) are of this type.
- To provide a consistent style across a program or set of programs. These rules are very similar to those to improve clarity (above), but whilst the clarity rules are based on objective issues of human perception, style issues are more subjective. For example, the adoption of a particular layout style or naming conventions. Such conventions, whilst having no impact on the interpretation of a program as far as the compiler is concerned, can improve communication between project team members, reviewers and maintainers. Unlike the clarity issues, there is no 'right' approach: the benefit comes from have an agreed common approach.
- To avoid common programmer errors. Programmers work for the majority of the time using a small subset of the programming language. There is a tendency to get into the mental habit of saying 'this feature works like...', remembering only a subset of the behaviour actually defined by the language standard. These rules act as reminders of the 'edge conditions' where a familiar construct may behave in an unexpected manner. For example, in C and C++ enumeration types map to integers. The normal expectation is that each enumeration type member is distinct. However, if required, members can be assigned explicit values, as in:

```
enum {red=4, orange, yellow, green, blue, indigo=6, violet};
```

  It may not be obvious to the programmer (though it is fully specified in the language) that yellow and indigo have been declared to be identical values (namely 6), as have green and violet (7). The MISRA rule set (MISRA 2004) addresses this by requiring that enumeration type declarations either:

  - provide no explicit integer assignments
  - assign a value to the first member only (the rest are then sequential)
  - assign a value to all members, so any equivalence is explicit

- To incorporate good practice, particularly with regard to 'future proofing'. HICPP (HICPP 2004) includes rules that programs should 'only throw objects of class type' and always 'catch exceptions by reference'. These don't protect against any particular problems or assist clarity, but do allow code to be re-used with less likelihood that some limitation will require a major rewrite. If an exception is thrown as a class object, and the re-use requires more information to be passed, a derived class can be used to extend it. If exceptions are caught by reference, all the information in the derived class will be available to the handler, particularly if the exception is re-thrown. This flexibility would not be available if the exception were thrown as a primitive type, or caught by copying.

Results of the JSF++ Review

The initial version of JSF++ was analysed against the identified C++ vulnerabilities. The results are shown in Table 2.

Classification of vulnerabilities	Issues
Completely covered by JSF++ rule	127
Duplicate	41
No issue, no action required	18
New rule needed	4
Change to rule required	18
Change to rule documentation required	118

Table 2: How the initial JSF++ addressed the identified C++ issues

As can be seen, something over half of the issues were entirely addressed by the proposed rules, or were deemed to be 'not an issue', either because they were a duplication of another issue or were not actually a problem that needed addressing (as was the case with most of the 'behaviours that require no diagnostics').

Of the remaining 140 issues, either a new rule was required to address the issue, an existing rule needed to be extended to fully capture the issue, or the documentation of an existing rule needed to be extended to ensure all the reasons for the rule were captured. The last of these points is an important one, because all coding standards recognise that there are going to be circumstances where a project is going to have to deviate from certain rules. In such cases it is important that the project can justify why the rule can be deviated from safely in these particular circumstances. In order to construct such a justification, it is necessary that all the reasons for having the rule in the first place are understood, so an argument can be made that all relevant aspects have been considered.

The current version of JSF++ incorporates these changes.

MISRA C++

Whilst the above activities had shown that the JSF++ was acceptable for UK safety related military systems, at the time (early 2005) Lockheed Martin regarded it as commercially sensitive, and were keeping it private.[2] However, JSF was not the only project looking to use C++ in safety related systems, and indeed, a number of non-real time, SIL1/2 ground based systems were already in development. It was felt that a more publicly available and peer-reviewed coding standard would be desirable.

At the Defence Aerospace Research Programme (DARP) conference in April 2005 the authors held a one day workshop on the safety related/safety critical use of C++ in avionic systems. One of the overwhelming conclusions of the workshop was that what was needed was "MISRA C++" (note the quotation marks). This would be a coding standard like MISRA C, with an associated rationale, and which would achieve a similar acceptance by developers and certifiers across multiple domains. (Certainly in the UK, MISRA C has become the de-facto high-integrity C standard for many domains, not just the motor industry). "MISRA C++" would not necessarily be related to MISRA C, however, or indeed to MISRA, particularly as earlier in the year, the author had made contact with a representative of MISRA at the Safety Critical Systems Conference (February 2005) and ascertained that MISRA had (at that time) no interest in C++.

The Avionics Software Standards Committee (ASSC) was approached about forming a working group to develop a C++ coding standard. ASSC is an MoD-sponsored defence avionics industry special interest group (managed by ERA). This structure was seen to provide two advantages:

[2] Subsequently, most of JSF++ has been made available on a public website (JSF 2005).

- its open nature would mean that a number of high integrity C++ tool vendors would be willing to offer their coding standards as a starting point, providing gearing for the effort available to the project, and avoiding having to 'reinvent the wheel' for established good practice (not to mention avoiding IPR issues over already published obvious good practice)
- its open nature would also mean that many of the supplier and certification bodies would be involved in its development, hence achieving the objective of developers and certifiers acceptance.

However, shortly afterwards, MISRA announced that they intended to start working on their own risk-reduction C++ subset. Given the 'brand recognition' that MISRA C has achieved, there seemed little point in developing a competing standard as this would simply confuse the market and make it harder to get the desired industry and academic buy-in. The working group planned around the ASSC has therefore been merged with MISRA's. This solved one immediate problem, namely that of how to write rules to cover those aspects of C++ inherited from C without infringing the IP of existing C coding standards.

The MISRA C++ working group are starting from the assumption that anything in MISRA C that is still relevant to C++ should also be in MISRA C++. For example, MISRA C bans function use before declaration, but as this is not allowed in C++, that rule becomes unnecessary. MISRA C has 126 rules for the core language (i.e. excluding libraries), some 55% of which are directly relevant to C++ and can be reused (virtually) unaltered. Some 8% of the rules are irrelevant to C++, and the remainder require some rewording, usually because the principle still applies, but C++ provides more mechanisms by which the effect being addressed can be made manifest.

MISRA C++ is also reviewing and incorporating rules from existing coding standards, including HICPP (HICPP 2004) (with the agreement of the IP holder). The aim is to incorporate rules that address objective issues, i.e. avoidance of unspecified (etc.) features, the clarity of code for review and maintenance, and avoidance of common programmer errors, whilst excluding more subjective issues, such as rules for consistent style or good practice for 'future proofing'.

It is expected that no language feature will be banned in its entirety. A likely exception is the *goto* statement. Indeed, there is a stronger reason to ban *goto* in C++ than in other languages. The justification for banning *goto* is usually along the lines that it can create programs that are difficult (if not impossible) to understand or analyse. In C++, the use of *goto* can cause programs to terminate in an unpredictable manner.

At the time of writing (August 2006), it is expected that MISRA C++ will have some 200 to 300 rules (roughly double that of MISRA C). By the time this paper is published, it is planned that a draft rule set will have been published for peer review.

ISO's proposed 'Software Vulnerabilities' standard

When the initial assessment of JSF++ was being made, there were some lively exchanges between Lockheed Martin and QinetiQ. Frequently the issue in dispute was not the technical interpretation of the behaviour of some program construct, but more fundamentally what aspects of behaviour should be in-scope or out-of–scope of a standard intended for safety related/critical use. The fundamental problem was a lack of a benchmark for the objectives of a high-integrity coding standard.

Early in 2006, the International Organization for Standardization (ISO) expressed a desire to create a generic standard for high-integrity software, "Guidance to Avoiding Vulnerabilities in Programming Languages through Language Selection and Use." The drafting of the standard is being chaired by members of the MITRE Corporation, with ISO-affiliated national bodies feeding in proposals and voting on drafts. For the UK, the affiliated body is the British Standards Institute (BSI).

The main focus of the working group is 'predictable execution', though certainly in the early stages, there is something of a divergence of opinion as to the overall objective:

- whether the focus is safety, security or both
- whether the target audience is the 'average programmer' or whose involved in recognisably critical systems
- whether the objective is to set the benchmark for the most critical systems or "raise the floor" for all development

At the 2006 DARP C++ workshop, the participants were asked what they would like to see in a generic high-integrity software standard. In addition to predictable execution was:

- a desire to record programmer 'intent' in applications, so that any verification can be carried out with reference to the code's specification, as occurs with SPARK Ada annotations

- a desire for a mechanism to record how effective guidance is, particularly:
    - are certain rules regularly giving false positives – being flagged as an issue which subsequent investigation shows is not actually a concern. This might indicate that the rule is drawn too widely and should be made more focussed.
    - are certain rules frequently being deviated from, by being modified or removed by project specific guidance? If so, why?

It is hoped that when this standard is complete it will fulfil the objective of providing a benchmark for assessing future coding standards proposed for critical systems.

Formal semantics for C++

The long term plan is to provide a formal semantics of a sub-set of MISRA C++, in the same way that $C^\flat$ (described in the next section) has provided a formal semantics of a sub-set of MISRA C. As with $C^\flat$, the aim is to provide the formal foundation for analysis tools.

Currently, work on C++ is running some two to three years behind that on C, so that by looking at the state of C development now we can see ahead to features that will become available to C++ over the coming years. In particular, we can see the approaches taken to formalising the language semantics.

3 Formalising C – the $C^\flat$ Subset

The popular perception of C is that it is the last language that should be used for safety critical software. This prejudice is not without good reason and has been the subject of books detailing problems with the language, for example Koenig (Koenig 1989) details many of the syntactic and semantic areas where mistakes commonly occur. However, a body of opinion has been growing for over ten years that there is no fundamental reason why software written in C cannot be of at least of as high an intrinsic quality and consistency as other commonly used languages (Hatton 1994).

This view holds that with disciplined usage, policed and supported by tools, C can be at least as "safe" as a language such as Ada. Indeed C is such a simple language it could be argued that its disciplined use supported by tools could lead to better programs than those written in Ada. This is because the Ada language is rather complicated and has its own areas of weakness – for example, parameters to a procedure may be passed by copy or by reference but should make no difference to the meaning of the program. Unfortunately this is not always the case and is impossible for a compiler to check and enforce. For this, and other reasons, the SPARK subset of Ada was developed (Barnes 1997).

Subsequent work developed a formal semantics in Z for the SPARK subset (Carré, O'Halloran, and Sennett, 1993) that led to the identification of some curiosities and the specification of a tool to check for run-time errors (Garnsworthy, O'Neill, and Carré, 1993). The SPARK subset of Ada83, and now the SPARK95 subset of Ada95, provide a prescriptive language basis suitable for formal verification of the most critical applications. The MISRA C guidelines, which form an analogous approach for C, are proscriptive, and forbid the use of certain C language constructs. This results in them providing a looser specification for a safety critical subset of C.

In early 2002 QinetiQ's Systems Assurance Group, SAG, developed a syntactic subset of C that conformed to the MISRA C guidelines. The rationale for the "Restricted C" subset (March, Smith, and Whiting, 2003), which is now called $C^\flat$, is that it should combine expressive power, simplicity, predictability, verifiability, and evolution.

Expressive Power

Before any work on formalising $C^\flat$ took place it was important to establish whether this subset was too restrictive, or whether it would be expressive enough for projects developing safety critical code. Conformance to MISRA C guidelines helped somewhat, but since $C^\flat$ is more restrictive (except where concerning fonts, naming conventions, etc.) it was not clear that it would still prove to be usable. The same approach was taken with SPARK originally: as projects adopted the early definition of SPARK, there was pressure to add features to the language subset in order to accommodate the practices of industry, without compromising the basic principles of SPARK. One of the principle strengths of SPARK over rival Ada subsets, such as Ana or Ava, was this accommodation of industry's needs.

As part of SAG's advisory role on safety critical software to MoD, the $C^\flat$ syntactic subset was given to parts of industry. This was targeted at helping projects which were unsure about how to use C in safety critical applications, and at learning best practice from projects that had already justified the use of C in safety critical applications. After discussion in 2003 with Eurojet GmbH in Munich, the $C^\flat$ subset was used as a basis for their programmer's manual (NATO 2005) and tool support for developing a relatively small amount of critical software concerned with monitoring a jet engine. In fact the de-facto subset used was slightly more restrictive than that of $C^\flat$ and the critical software has been subsequently developed.

Around 2003 SAG also passed on the $C^\flat$ definition to The Mathworks. The subsequent code generator for embedded C applications is slightly less restrictive than $C^\flat$, for example allowing a dangling else where $C^\flat$ forbids it, but if the code is always automatically generated this not a problem. An informal study into the code generation for embedded C applications indicates that, apart from a few syntactic deviations, the implicit C subset is compatible with $C^\flat$.

The early indications are that $C^\flat$ is expressive enough for critical applications, but it requires more validation.

Simplicity

The C language is itself relatively simple and $C^\flat$ removes the remaining "complications", thereby reducing its expressive power from full C. An informal abstract syntax[3] is presented below for expressions.[4]

```
EXP ::=     id_exp(ID) |
            cons_exp(CONST) |
            subscript_exp(Subscript_Exp[EXP]) |
            fun_call_exp(Fun_Call_Exp[EXP]) |
            comp_sel_exp(Comp_Sel_Exp[EXP]) |
            unary_exp(Unary_Exp[EXP]) |
            sizeof_exp(Sizeof_Exp[EXP]) |
            sizeof_tname_exp(Sizeof_Tname_Exp) |
            cast_exp(Cast_Exp[EXP]) |
            bin_exp(Bin_Exp[EXP]) |
            cond_exp(Cond_Exp[EXP])
```

The first two syntactic categories allow an expression in $C^\flat$ to be either an identifier (that depending upon context can evaluate to either a left or right value),

[3] The abstract syntax is in fact a free type in the Z language, and is the basis for a formal semantics that has been defined over this free type.

[4] It is assumed that a static evaluation of types and other healthiness checks have taken place during the production of the abstract parse tree.

or a constant expression. Constant expressions in this abstract syntax include strings, and any brackets are dealt with by a front end that parses the concrete syntax. The subscript expression covers arrays. The function call category denotes the call of a function, with the semantic restriction that function calls in an expression cannot have side effects. This semantic restriction is enforced by a front end checker of healthiness conditions. The component selection expression covers both direct and indirect selection of structures. The syntactic category of unary expressions covers simple value-producing operators, such as negation, and the operators of dereferencing (*) and taking addresses (&). The next two categories are concerned with the size of an expression and a type. The cast expression is self explanatory, and the syntactic category of binary expression is concerned solely with simple binary operations such as integer arithmetic. Finally the *cond_exp* category is the ternary conditional operator whose type coercions for each of the two sub-expressions are calculated by the front end healthiness checker.

The abstract syntax for statements is presented in the same style as that for expressions.

```
STMT::=        compound_stmt(Compound_Stmt[STMT])  |
               assign_stmt(Assign_Stmt)  |
               pre_stmt(Inc_Dec_Stmt) |
               post_stmt(Inc_Dec_Stmt)  |
               fun_call_stmt(Fun_Call_Stmt)  |
               void_fun_call_stmt(Fun_Call_Stmt)  |
               if_stmt(if_stmt[STMT])  |
               switch_stmt(switch_stmt[STMT])  |
               while_stmt(while_stmt[STMT])  |
               do_stmt(Do_Stmt[STMT])  |
               for_stmt(For_Stmt[STMT])
```

A compound statement is a list of declarations and a (possibly empty) sequence of statements. The most significant syntactic category in $C^\flat$ is that of assignment as a statement. This means that assignments cannot appear in an expression (this will be rejected by the front end healthiness checker) which in turn eliminates the problem of side effects in expressions (apart from function calls, which are checked separately). The assignment category includes assignment with *addition*, *subtraction*, *multiplication*, *division*, *modulus*, left and right assignment, and finally bitwise *and*, *or*, and *xor* assignment.

The syntactic category of pre-increment and -decrement are equivalent to *addition* and *subtraction* with assignment. The post-increment and -decrement category is semantically distinct from any of the assignment constructs of the previous paragraph. The function call statement allows side effects and returns a value. The void function call category is similar except that a void value is returned.

The usual control constructs of *if*, *while*, *do* and *for* statements are permitted, but *break* and *continue* statements are not permitted within them. These control statements must have compound statements as bodies, which avoids the common programmer error of adding extra lines which do not form part of the body of the statement. An *else* part is required with an *if* statement, thereby avoiding the dangling else problem. These features make these control constructs well behaved, unlike in full C where control can, for example, jump out of the construct. The switch statement requires the *break* statement to be present at the end of each labelled statement within the switch statement, except for the default case where it is optional. The effect of this structure within the switch statement makes it semantically well-behaved in a manner analogous to a case statement in a language such as Ada.

With the above syntactic and semantic restrictions, and with union types forbidden, $C^\flat$ is a regular and simple language, from both a programmer's and a formal semantics point of view.

Predictability

A central problem of predictability for the C language is when side effects combine with the fact that an expression can be evaluated in any order, depending upon the compiler used. For example, (a + b) + (c + d) can lead to the evaluation of sub-expressions a, b, c and d in any order, e.g. completely interleaved, even in the presence of parentheses. If a, b, c and d are assignments then the final state of the memory is difficult to predict.

Another problem of predictability for the full C language is that of undefined behaviour, which can arise in a number of ways. For example dividing by zero leads to undefined behaviour, as does accessing beyond the end of an array, assigning to a piece of memory that has not been allocated or various other dynamic healthiness conditions. In his PhD thesis (Norrish 1998), Norrish presents an operational semantics for most of the C language, where these dynamic healthiness conditions are preconditions for an operational rule concerning the behaviour of a C construct to be "executed".

The $C^\flat$ subset achieves predictability largely by syntactically disallowing side effects. The only permitted side effect is a function call in an expression, and this is policed by a checker of healthiness conditions that also checks for syntactic conformance. As indicated previously many of the healthiness conditions are dynamic. Rather than incorporate the necessary dynamic semantic preconditions into a functional verification tool it has been decided to separate this out into a separate Abstract Interpretation phase. An Abstract Interpreter has been designed that not only checks for dynamic run time errors, such as accessing beyond an array bound, but also checks for preconditions to legal assignments and function calls, and for all the other healthiness preconditions necessary for defined behaviour of compliant C programs.

The Abstract Interpreter requires a model of memory with offsets and the many other lower-level constructs necessary to articulate the dynamic healthiness conditions for compliant C programs. The Abstract Interpreter symbolically executes a C program, carrying around symbolic values and the conditions under which these values are meaningful. The conditions are subject to separate predicate simplification technology.

Employing a front end healthiness checker guarantees (subject to sound simplification and environmental assumptions) that any programs that are accepted will be in the $C^\flat$ subset and will execute predictably according to the semantics of C. To enhance confidence that this is the case a formal semantics of $C^\flat$ has been defined in Z, based upon Norrish's work (Norrish 1998).

Verification

There is confidence that the operational semantics defined by Norrish (Norrish 1998) is faithful to the C language definition because it has been mechanically checked using the HOL theorem prover, and certain properties one would expect of the language were mechanically deduced from the operational semantics. This is not a guarantee that the operation semantics is that of C, but it is a significant confidence-building measure that the semantics are valid.

The formal semantics of $C^\flat$ in Z is essentially a big step operational semantics that lends itself to defining a predicate transformer semantics for statements in $C^\flat$. Such a predicate transformer semantics is currently being defined in terms of the operational semantics in order to specify a verification tool for $C^\flat$ programs.

A prototype verification tool for $C^\flat$ programs is under construction against a newer specification in Z that builds upon the operational semantics and the previously defined predicate transformer semantics for statements. The intention is to use operational semantics to generate verification conditions based on specification statements annotating the code. This should not be confused with a

Floyd-Hoare logic approach to verification of C, which is the subject of promising research on separation logic (Reynolds 2002) and is much more mathematically sophisticated and ambitious.

It is more ambitious because it is attempting to reduce program correctness to syntactic manipulations of logical formulae at the program level. The approach reported here relies on lower level semantic transformations where correctness conditions can be presented at the program level at various points. To a user of a verification system the distinction will be a fine one and is of much more relevance to tool builders.

If the prototype tool can be successfully employed on real programs then the semantic gap between the source code and the compiled code will be significantly smaller than in a language such as SPARK. This makes the verification of the compiled code against $C^\flat$ source code simpler than an equivalent exercise for SPARK, although this is out of scope at present.

Evolution

$C^\flat$ is intended to be a language core for critical applications that can be relaxed and evolve. One evolution of $C^\flat$ would probably be a new language in which the restrictions that make reasoning simpler and human mistakes less likely to occur have been relaxed. For example, removing the requirement for an else part in an if statement makes no semantic difference, but introduces the dangling else problem discussed earlier. This is only really a problem when code is developed manually, therefore for automatically generated code this restriction could be removed.

A more significant relaxation is to allow side effects in expressions. As already discussed, predicting the final state of memory can be computationally difficult. However, by removing current syntactic restrictions in $C^\flat$ and introducing semantic restrictions, it should be possible to tractably determine whether the presence of side effects will result in the same memory state, regardless of the order of evaluation. It is anticipated that this could be performed efficiently by the front end healthiness checker, allowing this concern to be separated out from the functional verification task.

Side effects are already accommodated within the formal semantic model described by Norrish (Norrish 1998). The $C^\flat$ semantic model also accommodates side effects, but they are redundant because of the syntactic restrictions. The ability to formally extend a verification tool to verify a more permissive language depends upon how badly it is required, which will only occur through practice and experience.

4 Summary

Substantial progress has been made towards having a tool-supported, formally defined subset of C, suitable for use in safety critical projects.

Similar work is planned for C++, and is lagging some 2 to 3 years behind the work on C$^\flat$ – but will the result be called C$^\flat$++ or C$^{\flat\flat}$?

A challenge for the future is how to deal with the next 'new' language to be adopted by a safety critical project. Can any common approach or tool be developed from experience with SPARK, C$^\flat$, or C$^{\flat\flat}$?

5 References

Barnes J (1997). High Integrity Ada: The SPARK Approach, Addison-Wesley, 1997. ISBN 0-201-17517-7.

C++ (2003). ISO/IEC 14882:2003(E), Programming Languages – C++ Language Standard, ISO, 1998.

Carré B, O'Halloran C, and Sennett C T (1993). Final Report on Work to Define a Formal Semantics for SPARK, DRA customer report, 1993.

Garnsworthy J, O'Neill I, and Carré B (1993). Automatic Proof of the Absence of Run-Time Errors. In: Ada: Towards Maturity – Proceedings of the 1993 AdaUK conference, IOS Press, 1993. ISBN 9051991428.

Hatton L (1994). Safer C, McGraw-Hill, 1994. ISBN 0-07-707640-0.

HICPP (2004). High-Integrity C++ Coding Standard Manual v2.2, The Programming Research Group, May 2004. Available from: http://www.codingstandard.com/HICPP_MANUAL_REQUEST.htm

JSF (2005). Joint Strike Fighter Air Vehicle C++ Coding Standards for the System Development and Demonstration Program, Document Number 2RDU00001 Rev C, December 2005. Available from: http://www.jsf.mil/downloads/down_documentation.htm

Koenig A (1989). C Traps and Pitfalls, Addison-Wesley, 1989. ISBN 0-201-17928-8

March M, Smith A, and Whiting E (2003). Concrete and Abstract Syntaxes for Restricted C, QinetiQ internal report, Version 2, July 2003.

MISRA (2004). MISRA C: Guidelines for the Use of the C Language in Critical Systems, Motor Industry Research Association, 2004. ISBN 0-9524156-2-3.

NATO (2005). EJ200, Digital Electronic Control and Monitoring Unit (DECMU) Software Programmer's Manual for C, EJ 494/12000 Issue 1E, NATO UNCLASSIFIED, Date of Issue: 08-AUG-05.

Norrish M (1998). C Formalised in HOL, PhD thesis, Cambridge University, 1998.

Reynolds J C (2002). Separation Logic: a Logic for Shared Mutable Data Structures. Invited Paper, Proceedings of the 17th IEEE Symposium on Logic in Computer Science, 2002; 55-74.

Schofield A, and Pygott C (2006). A Tabulation of the Unpredictable Features of the C++ Language, QinetiQ Report QINETIQ/S&DU/TIM/CR060019, September 2006.