Issues in the Conduct of PSSA

S K Dawkins, T P Kelly, J A McDermid, J Murdoch, D J Pumfrey; University of York; York, UK.

Keywords: Processes, Safety Assessment, Safety Management, Standards and Guidelines

Abstract

Aerospace Recommended Practices (ARPs) 4754 and 4761 introduce the concept of preliminary system safety assessment (PSSA) as a key stage in the safety process for systems on civil aircraft. PSSA is intended to follow functional hazard assessment (FHA). Its purpose is to assist in validating a proposed system architecture and to allocate (derived) safety requirements to components of that architecture. Although the ARPs claim to represent "best practise" some of their recommendations, including the conduct of PSSA, are novel, and it is not always clear how to interpret and apply them. The purpose of this paper is to give some guidelines on the conduct of PSSA, based on our experience of assisting a number of organisations in developing safety processes in response to the ARPs.

We discuss some major issues which, in our experience, cause significant difficulties in using the ARPs. The ARPs are clear about the purpose of PSSA – but in our experience the purpose isn't always adequately understood (in part this is due to the nature of the example in ARP 4761 Appendix L). Where practical, we illustrate our concerns by presenting a critique of the example in Appendix L of ARP 4761.

Introduction

The "new" civil aerospace guidelines ARP 4754 (ref. 1) and 4761 (ref. 2) set out requirements for the system safety process as applied to aircraft (and engines). ARP 4754 is the "higher level" document dealing with general certification. ARP 4761 gives a more detailed definition of the safety process, and presents a worked example of the process in Appendix L.

The guidelines identify several phases of the system safety process, see figure 1. Hazard assessment is carried out primarily through functional hazard assessment (FHA). FHA is used to identify hazardous functional failures of the aircraft and to help identify safety requirements, e.g. the acceptable rate of occurrence of hazardous functional failures. Preliminary system safety assessment (PSSA) is concerned with analysing proposed system designs (architectures) to validate the safety of the design, and to identify derived safety requirements which will guide further development of the design. Typical derived safety requirements include budgeting (allocating) failure rate requirements to components of the system architecture and setting development assurance levels (DALs). DALS are a mechanism used within the ARPs to classify system components based on their most severe failure condition associated with an aircraft level function. They range from level A - where failure is considered catastrophic, down to level E, where failure has no safety effect. Each level reflects the amount of rigour required from development assurance techniques such as reviews and testing.

These two ARPs are intended to document good practise. However we have seen difficulties arise in the application of the principles underlying PSSA which suggest two problems. First, in producing the guidelines good ideas have been drawn in from a number of sources – so the resultant process isn't good *practise*, but rather a hypothetical "good process" which hasn't been fully validated, and hence which has some weaknesses. Second, the guidelines (especially ARP 4761) are being used as tutorials – and they do not contain enough explanation to function in this role (although they are much better than many standards).

Against this background, the purpose of this paper is to illustrate some of what we see as being key difficulties with the ARPs, and to identify ways of ameliorating some of the problems. The paper necessarily stops a long way short of being a tutorial, but we hope it addresses some of the key issues with the guidelines.

It is worthwhile making a few further comments before discussing the above issues in detail. First, the comments are couched in terms of the ARPs, but we believe they are more general. For example, there is a phase known as sub-system hazard analysis (SSHA) in MilStd 882C (ref.3) which is very similar to PSSA,

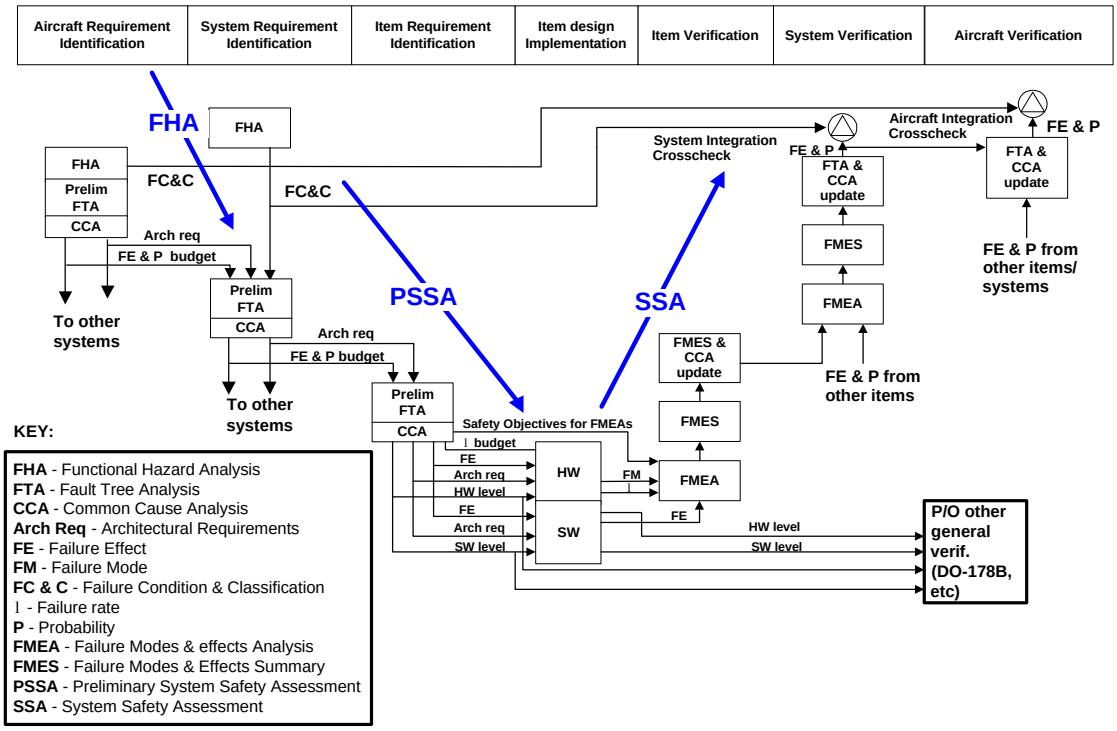

Figure 1 - The ARP Safety Assessment Process

and our comments apply to SSHA also. Similar phases appear in other standards, so we believe that our comments are relevant in the context of many functional safety standards. Second, the issues raised below reflect experience in working with aircraft manufacturers in Europe and system suppliers in Europe and the USA. Thus our belief is that these problems are quite widespread. Third, we present much of our discussion in terms of Appendix L in ARP 4761. This serves the dual purposes of presenting comments which help to interpret the guidelines, and avoids discussing commercially sensitive details of systems we have seen.

Finally, this paper should not be seen as trying to undermine the ARPs. We strongly believe in the principles underlying the guidelines. Instead the following should be seen as trying to strengthen the guidelines, and to make it easier for them to be taken up in the industry.

The next section outlines our views on the role of PSSA in the safety and design process, reinforcing points made in the ARPs, but also pointing out some of the difficulties of meeting the intent of the ARPs. The third section identifies difficulties we have seen in conducting PSSA, particularly in terms of understanding the role of PSSA. This is firmly based on our industrial experience, but is presented in general terms to avoid issues of confidentiality. The fourth section discusses more detailed technical issues with the guidelines, identifying ways in which the guidelines need to be augmented (or the examples modified) in order to make them more effective (e.g. in a tutorial role). This is presented largely in terms of examples in the ARPs. The final section briefly identifies some other issues with the guidelines which are outside the scope of this paper, and indicates areas of future work.

The Role of PSSA in the Process

PSSA occurs at a critical stage in the aircraft development process, see figure 1. At this stage the design – especially the architecture of the systems – is emerging and evolving, and the designers are having to make trade-offs between (potentially) conflicting requirements. The role of the PSSA is to act as a *design driver* to ensure

that the design (system architecture) chosen is credible from a safety perspective[1].

ARP 4761 identifies the role of PSSA as follows:

"PSSA is a systematic examination of the *proposed system architecture(s)* to determine how failures can cause the functional hazards identified by the FHA. The objective of the PSSA is to *establish the safety requirements* of the system and to determine that *the proposed architecture can reasonably be expected to meet the safety requirements* identified by the FHA." (ref. 1, our *emphasis*).

There are two key elements here:

- Validating the architecture – showing that the architecture is *credible* as a way of meeting the safety requirements derived in FHA. Note: this is as much a project risk management issue as a safety issue. In validating the architecture we are seeking to reduce the risk of reaching the end of the development process and finding that the system is not certifiable – hence introducing cost and time over-runs into the process.
- Producing derived safety requirements (DSRs) – establishing requirements on the system components which, if they are met, will enable the architecture to meet its safety requirements. As well as the component failure rates alluded to above, this covers DAL assignment, establishing maintenance check intervals, architectural requirements (e.g. for redundancy) and process issues – such as requirements to check independence of failures (through common cause analysis), etc.

Note that these elements are inter-linked as the validation is contingent – the architecture can only meet the safety requirements established at FHA, provided the components meet their DSRs.

Also, there is an implicit aim of PSSA not made explicit in the ARPs – the need to reconcile DSRs for different hazards. Typically each system will contribute to more than one aircraft hazard, and DSRs will arise for each hazard. These DSRs need to be reconciled to ensure a coherent design, and to avoid over-engineering.

In addition PSSA should contribute to trade-offs (trade studies) between alternative architectures and the selection of the architecture which best meets the overall requirements placed on the system (the aircraft). In other words, PSSA should be seen as a part of an overall systems engineering programme. This point does not seem to be stressed adequately in the guidelines. Indeed there is a design-analysis-revision "loop" into which the PSSA should fit, see figure 2.

The left-hand side of figure 2 shows the PSSA process as described in Appendix B of ARP 4761. The right hand side provides an interpretation of this description. This interpretation highlights:

- The need for safety and design perspectives when selecting the most appropriate architecture with regard to both performance and safety requirements.
- The need for feedback to allow revisions or derived requirements that result from PSSA work to be addressed in support of system development. The ARP definition (on the left of the diagram) does not highlight the importance of this positive feedback.

The intent behind both of these points is to allow PSSA to contribute early to the design process and to accept and cope with any resulting changes or new requirements.

As well as being crucial, PSSA is also difficult. PSSA occurs at the stage in the lifecycle where:

- There is most uncertainty. During the earlier stages of preliminary hazard identification and FHA the hazards associated with the system concept are fixed. By the System Safety Analysis stage, the design should be stable. It is during the PSSA stage that information becomes available to evaluate design options and this leads to significant uncertainty about design solutions and proposals.
- There is a need to address technical detail, and identify where information is incomplete or needs altering for the new context.

[1] This suggests that the name of the phase is perhaps rather misleading.

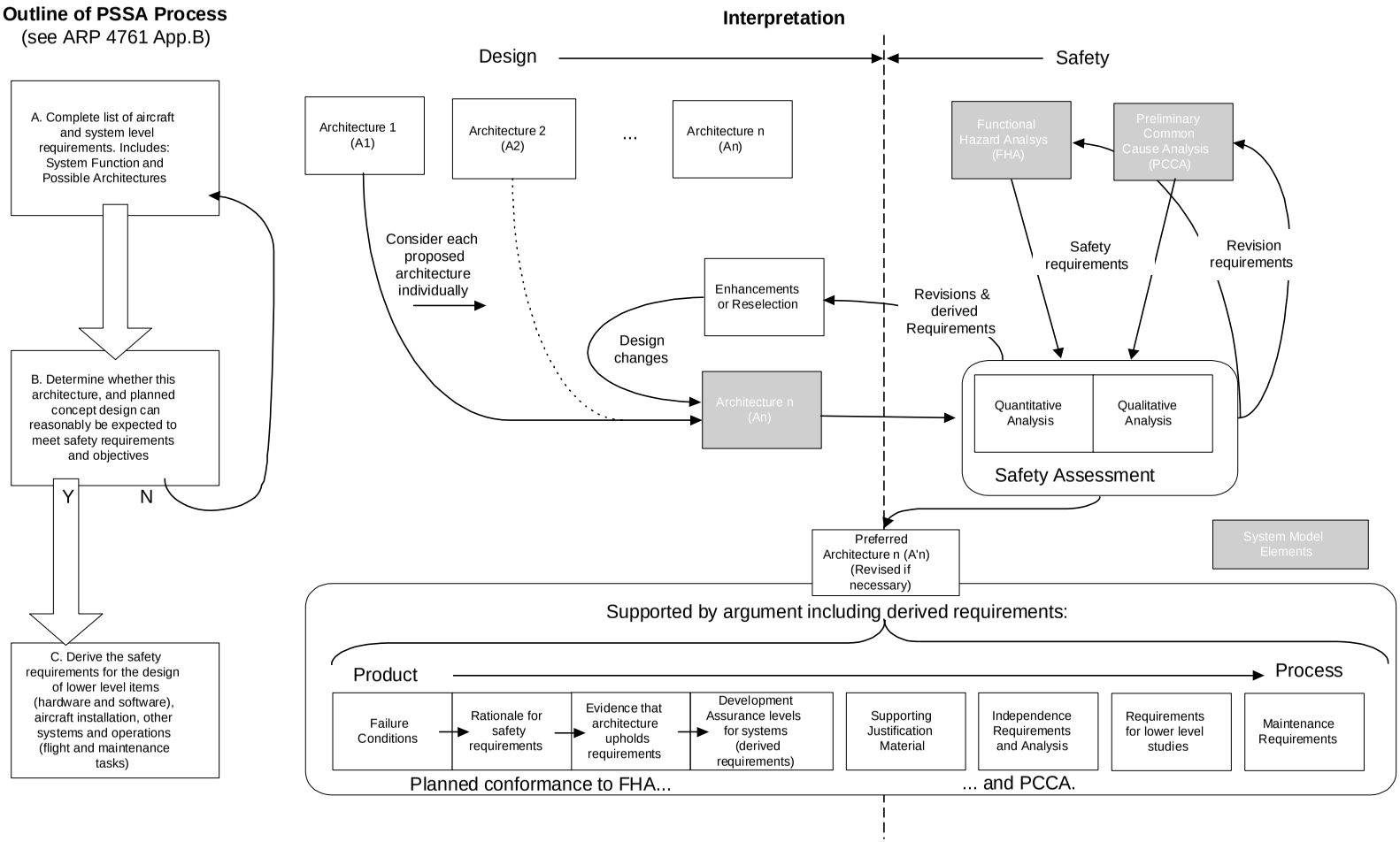

Figure 2 - Interaction and iteration during PSSA to enhance the system model

Finally, it should be stressed that PSSA is a risk reducing, and value adding, phase. If it is done well, there is a much greater likelihood that the system design can be completed without the necessity for major rework (due to belatedly discovered safety problems). Further, in our experience, systems are often over-engineered to deal with uncertainty in the design – good PSSA should reduce this tendency, and lead to more nearly optimal designs being produced.

Perceived Problems with PSSA in Practise

We have seen a number of PSSAs produced in (and for) the European aerospace industry, given courses on the conduct of PSSA, and talked to others involved in applying the ARPs (including some of the authors of the guidelines). The following is a distillation of what we perceive as problems of principle – which arise in practise! The order of the points is not very significant – but perhaps the first point is most important (and arguably it includes, or implies, all the others):

- Mistaking the role of PSSA – we have seen several PSSAs which try to show that the proposed design *is* safe, not that it *can be* safe *if* the components are implemented appropriately. Specifically we have some PSSA documents which presented large amounts of safety analysis then concluded that there are "no derived safety requirements" – and this misses the crucial role of PSSA in contributing to the design.

- Sources of DSRs – where do DSRs come from, and how do we know they are reasonable? We have found analysts have difficulty with this issue, and we see two sub-points:

  i. Establishing failure rate budgets, or targets, is a "free choice" – we can put any numbers into the fault trees so long as they give the appropriate top event probability. However this is of little use if we cannot find components which meet this sort of requirement. Where the requirements are on components we can use manufacturers data sheets, etc. (weighted for the operating environment). Also historical data on reused sub-systems and equipments is a useful guide – but it is only a guide, and all of these figures are still DSRs, see below.

  ii. Where there is a requirement for a "new design" how do we know what the failure modes are? In general we need to carry out exploratory hazard or safety analysis (e.g. HAZOP) to identify potential hazardous failure modes. An illustration of one way in which this can be done is given in (ref. 4), but we note that the guidelines are "mute" on how to identify such failure modes. This problem is exacerbated as we are inevitably dealing with incomplete designs at PSSA.

- DSRs versus "facts" – we have seen confusion about what constitutes a DSR. For example, where existing equipment is being reused between aircraft, the achieved failure rates and modes in the existing aircraft are taken as givens (facts) for the new aircraft[2]. Even if these rates and modes are achieved in the new aircraft (and the environment, etc. is different so this is an assumption) they are still DSRs for the purposes of the PSSA and need to be verified later in the safety process. At the PSSA stage there are probably no "facts" and all information about failure rates and modes should be treated as DSRs requiring verification.

- Tracking the designs – there is a genuine conundrum in doing PSSA. To meet its objectives we want to do PSSA early and thus influence the design, but we will then be faced with the cost of updating the PSSA at each design change. Conversely, by waiting until the design is "stable" we will save money in PSSA, but lose the ability to influence the design cheaply. What is needed is a "lightweight" way of doing PSSA early on, which becomes more rigorous as the design matures – but this is a research issue. (Note: ARP 4761 Appendix L tries to illustrate this, but there are inconsistencies in the example!)

- Responsibility for the design – we have seen cases where the safety analysts have ended up driving the design, or the designers say "if you have these concerns then why don't you propose the design". Again there is a genuine problem here. The designers *must*

[2] This sort of confusion arose in PSSAs where the analyst concluded that there were no derived requirements.

own the design, as they have responsibility for making trade-offs between different factors, e.g. safety, weight, cost, availability, etc. However it may be that the safety perspective gives insight into effective designs. In principle the solution here is for more closely integrated working of the design and safety teams – perhaps via so-called integrated project teams – and perhaps for the designers to do some of the early PSSA activities. However this exposes cultural difficulties and, in part, one of the issues facing the would-be implementers of the ARPs is that they need to achieve cultural change.

- Modifications – many systems these days are developed by modifying and adapting other systems, including moving them between members of the same aircraft family, but the guidelines seem to focus on the development of new systems. There are two sub-issues here. The first is that guidance is needed on how the PSSA approach should be modified to cope with reuse, modification and evolution of systems, where many more design parameters will be fixed and the options for trade-offs and allocation of budgets may be severely limited. The second issue is the concern that even small changes may necessitate significant re-analysis; in effect, the effort required to implement a change becomes proportional to the size of the system, not the size of the change. Guidance is required on how to identify the extent of the re-analysis required and, if possible, on how to structure the PSSA to limit the impact of minor changes.

We haven't suggested specific remedies for all of these problems. Where we don't propose solutions it is our view that the problems will tend to be reduced if the role of PSSA is properly understood.

Technical Issues in Performing PSSA

At a more technical level, we have seen various problems in performing PSSA, many of which can be illustrated in terms of the ARP 4761 Appendix L contiguous example. For brevity, we focus mainly on the use of fault trees, although we start with a more general point:

1. Types of DSRs – perhaps because of the content of ARP 4761 Appendix L, PSSAs we have seen tend to stress failure rates of components, but these are not the only DSRs. Others include:

   1.1. DALs for systems and components;

   1.2. Maintenance requirements, e.g. check intervals to ensure that dormant failures are revealed in a suitable period;

   1.3. Independence requirements to be verified later in the safety process;

   1.4. Recommendations for design changes (see below).

   The tendency to downplay these other forms of requirement could be ameliorated by being more explicit about ways of recording, propagating and tracking DSRs.

2. Propagating or allocating failure rate budgets – the ARP uses fault trees to allocate failure rate budgets (perhaps more accurately to represent the result of allocating failure rate budgets) in a "top down" manner. This reflects the spirit of PSSA, but some of the examples are technically flawed. Consider one of the PSSA fault trees from ARP 4761 Appendix L, reproduced as figure 3. The allocation has been done using simple arithmetic on probabilities at each gate. However the probabilities do not "add up", as there are common events in sub-trees. If one works out the cut sets and evaluates the probabilities bottom up, assuming that the lowest level allocated probabilities are correct, then the top event probability is 4.105xE-05 not 3.30xE-05. In other words, the figures allocated are *less stringent* than necessary to meet the requirement. This aspect of the ARP example is very misleading.

3. Independence assumptions – there are many implicit independence assumptions in the structure of the fault trees. Some of these are drawn out as DSRs, for verification in common cause analysis – but by no means all. More seriously, some of the "base events" in PSSA trees clearly aren't independent.

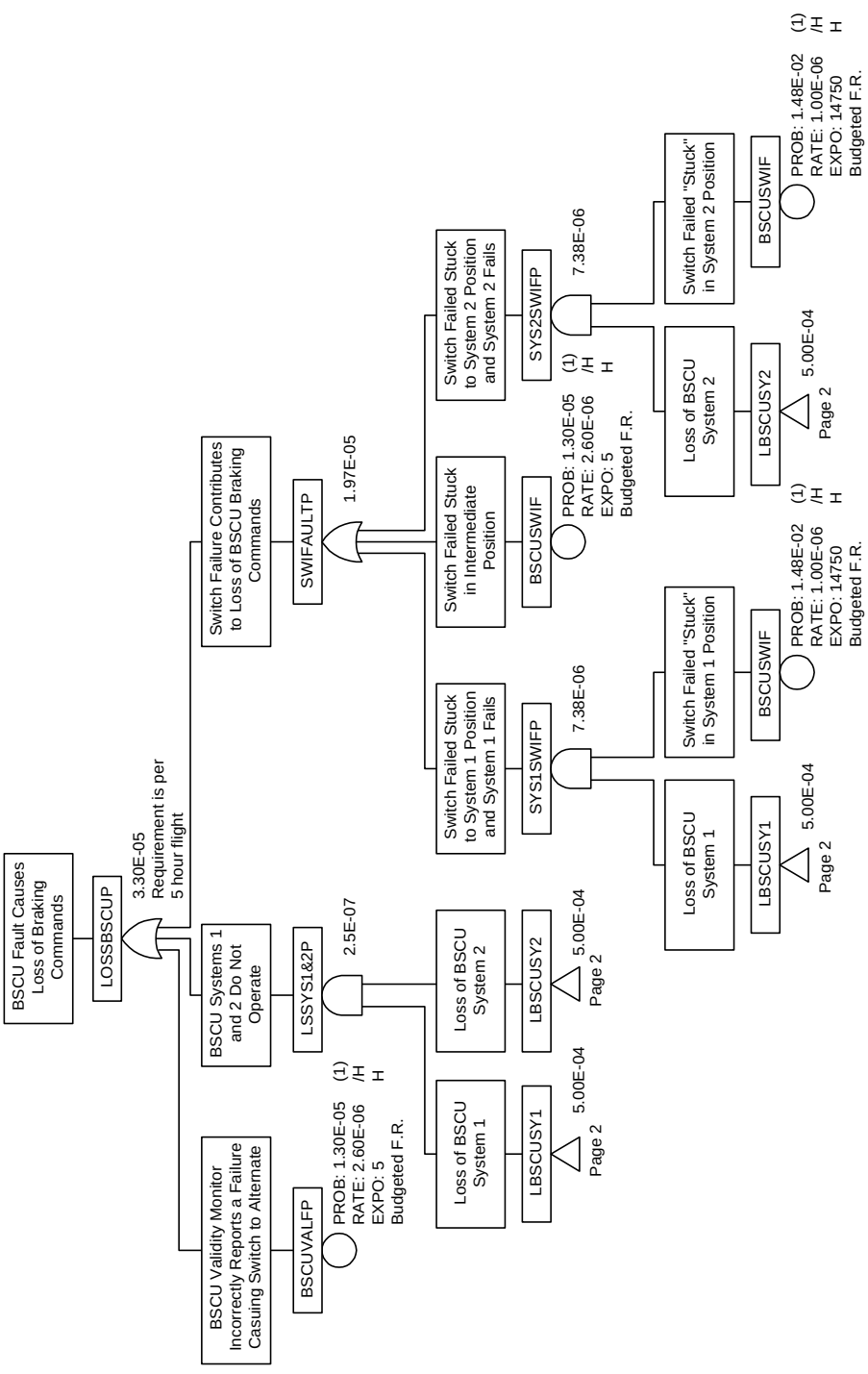

Figure 3 - Example of Failure Rate Allocation in Fault trees

This relates to the above point and also leads to optimistic setting of budgets. For example, the fault tree shown in figure 3 implicitly shows that the two BSCU devices (Brake and Steering Control Units) should be independent not only of one another, but also of the switching system that determines which BSCU is currently active.

4. Balancing budgets – the PSSA trees are treated in isolation, even where they relate to common equipment. In general, PSSAs

carried out for distinct hazards associated with a system will produce different DSRs in respect of the same sub-systems or components. These requirements need reconciling, or balancing (just choosing the most stringent may lead to over-engineering). Thus a process is needed to draw together all the PSSAs which relate to a system or equipment, and to adjust the DSRs to meet the high-level requirements established at FHA and to optimise the design. This step in the PSSA process is missing from the ARP.

5. Exposure and check intervals – implicitly the Appendix L examples include assumptions about check intervals for dormant failures in the exposure times for particular failures. At minimum these assumptions should be made explicit as DSRs (or recommendations) for the logistics team. As with the failure rate budgets the check times need to be balanced, to minimise maintenance disruption. More significantly, the combinations of exposure times in the trees are not always valid, as illustrated in "point 5" on figure 4, where two BSCU hardware components are shown with different exposure times. The BSCU validity monitor has a quoted exposure time of 100,000 hours - relating to about 20,000 flights between maintenance, whilst the BSCU command channel CPU quotes an exposure time of 0.004167 hours (relating to the 15 seconds of each flight during which braking is critical). To fulfil both requires the BSCU to be inspected after each flight. Hence the exposure times are inconsistent. For consistency, the check interval needs to be reduced to five hours (the average flight time quoted in the Appendix L contiguous example), and thus, the utility of the monitor has been significantly reduced.

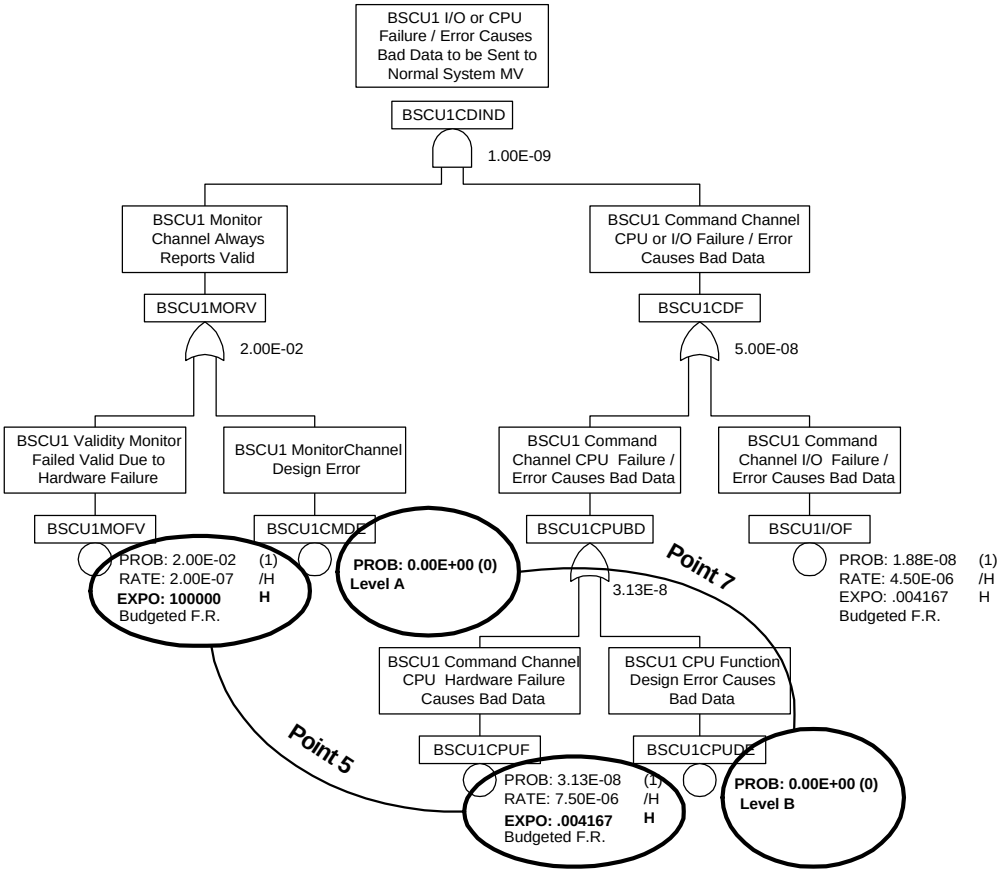

Figure 4 - Examples of issues relating to the Fault Tree Analysis

6. Use of different analyses – one of the reasons for the problem identified above is that fault trees are being used to describe situations that really require the expressive power of Markov models to do properly (although even here care is needed in dealing with exposure times, etc.). More generally, the ARP tends to treat dependence diagrams, Markov and fault trees as equivalent – but they are not. Fault trees and dependence diagrams (DDs) can express the same failure logic – but Markov can express repair and recovery strategies that cannot be expressed in FTA or DDs (ref. 5). The ARP needs to be clearer about this.

7. DALs – the treatment of DALs in Appendix L is unsatisfactory. According to established practise, and the rules in ARP 4754, DALs are applied at system level, and then allocated to sub-systems and equipments. In Appendix L the DALs are assigned in the trees to (seemingly) arbitrary components of the design (see "point 7" in figure 4). Further, ARP 4754 gives a table (see Table 4 in ref. 2) of ways in which it is legitimate to reduce and allocate DALs. Allocation of DALs to parts of the design should follow these rules – and it is far from clear that the logic behind the allocation in figure 4 does. The figure shows two BSCU design errors linked by an AND condition to one top event. A monitor design error is considered to relate to DAL level A, whilst the CPU function design error warrants DAL level B. These assignments are made without reference to the need to show independence and dissimilarity between the two design errors as required by ARP 4754. Note that lower level design detail implies that these functions are not independent, thus underlining our point.

The difficulty underlying the last three points is the fault trees are being made to "do too much". In points 5 and 6 fault trees are being used where other analyses would be more effective. In point 7, the fault trees are simply being used inappropriately – there is no "calculus of DALs" which can be employed in a manner analogous to the calculus of probabilities.

These problems in Appendix L arise, we suspect, at least in part because of the tools used. Fault tree tools support propagation of probabilities bottom up – not the top down allocation required for PSSA. They don't support the sort of manipulation of failure probabilities required for "balancing" budgets. Further, they typically don't support a mixture of budgets and real data – which is what is required to support evolution of the trees through the process. It would also help if the tools could mark and export DSRs. Thus there are research issues for the tool vendors as well.

Conclusions

The ARPs are making a useful contribution to the development of the aerospace safety culture by defining an improved safety process. We have, however, seen difficulties with attempts to use the process, as indicated above. There are a few other concerns which we mention briefly:

- Stopping: there is little practical guidance available on where (what level of design detail) to stop the PSSA process. Experience shows that techniques in common use today (particularly FHA) suffer from over-zealous application down to the level of design minutiae which add little safety insight, but significant cost. PSSA must avoid this trap.

- Engines: the guidelines say they relate to aircraft and engines – but no real guidance is given on how to deal with engines. Are they meant to be treated at the same level as aircraft, or as systems? We presume it is the former, but the guidelines don't make this clear. More generally there are issues which affect engines (most of the hazards arise from mechanical causes, not functional failures, etc.) which need better treatment.

- Complex systems: the guidelines say that they are concerned with complex systems, but where are the issues of complexity dealt with? For example, where is the treatment of human factors, which is arguably the most significant factor in aviation safety today? Similar arguments can be made about software, and forthcoming issues such as Integrated Modular Avionics. Overall, the guidelines do not seem to live up to their "billing".

Inevitably the understanding of how to apply the techniques will come from greater experience – but we feel that there is a need to update the

guidelines at an early stage, to address some of the issues outlined above.

Although the inclusion of the examples in ARP 4761 is laudable (they are the best published example of a "joined up" safety process we know) more complete examples are still needed. Perhaps more significantly, the examples need associated explanation of *why* the analysis has been done in a particular way – to enable others to apply the techniques in situations not covered explicitly in the guidelines.

We believe that the guidelines need to be updated with some urgency and more tutorial material developed. We have also identified some research issues which we believe the community needs to address to get the best value from the new processes defined in ARP 4754 and 4761. Finally, we note that a culture change, breaking down traditional barriers between designers and safety analysts is needed if the maximum benefit is to be gained from the ARPs and PSSA.

References

1. Society of Automotive Engineers Inc, Aerospace Recommended Practice (ARP) 4754: Certification Considerations for Highly-Integrated or Complex Aircraft Systems, November 1996.

2. Society of Automotive Engineers Inc, Aerospace Recommended Practice (ARP) 4761: Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment, December 1996.

3. US Department of Defense, Military Standard 882-C: System Safety Program Requirements, January 1993.

4. Murdoch J, McDermid J A, Wilkinson P, Astley K, Reid S, Applying HAZOP to Functional System Models in 16th International System Safety Conference 1998, Seattle WA US, Sept 14-19 1998.

5. Malhotra M, Specification of Dependability Models for Fault Tolerant Systems, Duke University, USA, 1993.

Biographies

Steven Dawkins, Department of Computer Science, University of York, Heslington, York, YO10 5DD, England. Tel: +44 1904 433385; Fax: +44 1904 432708
Email: steven.dawkins@cs.york.ac.uk

Steven Dawkins works as a Research Associate in the area of System Safety for the British Aerospace Dependable Computing System Centre (DCSC), specialising in the use of safety analysis to assess and certify innovative technologies.

Tim Kelly, Department of Computer Science, University of York, Heslington, York, YO10 5DD, England. Tel: +44 1904 432764; Fax: +44 1904 432708
Email: Tim.Kelly@cs.york.ac.uk

John McDermid, Department of Computer Science, University of York, Heslington, York, YO10 5DD, England. Tel: +44 1904 432726; Fax: +44 1904 432708
Email: john.mcdermid@cs.york.ac.uk

John McDermid is Professor of Software Engineering at the University of York. He directs the High Integrity Systems Engineering group at the University which is a recognised centre of excellence in safety critical computing for the UK Aerospace industry. He directs the Rolls-Royce University Technology Centre (UTC) in Systems and Software Engineering and the DCSC. He has published 6 books and over 200 papers.

John Murdoch, Department of Computer Science, University of York, Heslington, York, YO10 5DD, England. Tel: +44 1904 433375; Fax: +44 1904 432708
Email: john.murdoch@cs.york.ac.uk

John Murdoch has worked for over fifteen years in the aerospace industry. Currently he is with the Rolls Royce UTC at the University of York UK, with interests in process modelling for high integrity systems.

David Pumfrey, Department of Computer Science, University of York, Heslington, York, YO10 5DD, England. Tel: +44 1904 432735; Fax: +44 1904 432708
Email: david.pumfrey@cs.york.ac.uk

David Pumfrey is a research and teaching fellow in the DCSC, where he is currently investigating the use of HAZOP and related techniques for software hazard analysis. He is also one of the presenters of a highly successful series of short course on system safety and safety cases.