Trends in System Safety: A European View?

John A McDermid
Department of Computer Science
University of York
Heslington, York, YO10 5DD, UK
John.McDermid@cs.york.ac.uk

Abstract

This brief paper sets out some perceptions of trends in regulatory policy, public perception of safety, systems and software engineering which impact safety engineering practises. The paper takes a European perspective, but is largely a personal view. It should also be noted that many of the change drivers are of international scope.

Keywords: Safety critical systems.

1 Introduction

Like many aspects of engineering, safety practises evolve over time. In safety, the change drivers come from a number of sources including:

- regulatory policy;
- public perception of risk;
- requirements for increased functionality and sophistication of products;
- advances in technology for systems and software engineering.

The latter two drivers are essentially global, but the first two are more "localised" in their nature. This paper discusses all the four areas, but emphasises those issues of regulation and policy which (so far as the author is aware) distinguish the trends in Europe from those elsewhere in the world. All four change drivers pose issues for safety engineering – in some cases there are research challenges, and in other cases the difficulties arise in changing working practises.

Having identified issues which pose challenges for safety engineering, the paper then considers responses to the challenges, focusing mainly on technical and research issues. Also, some consideration is given to changes in standards.

It is not practical, in such a short paper, to do full justice to the subject. Also, it is unlikely that any individual will be aware of all trends, in all industries. Thus what follows should be viewed as personal perceptions of significant changes in a number of industries and technologies, but with a bias towards the aerospace industry and software technology.

2 Trends and Challenges

The following trends are those that have struck the author as providing particular challenges to achieving or assessing safety; there is no claim that the set of issues identified is exhaustive.

2.1 Regulatory Changes

In some application sectors, e.g. Air Traffic Management (ATM) and Railway Signalling, there is a trend to move from "rule based" to "risk based" regulation. The details, and the stage in the transition process, vary between sectors, but there are some common factors:

- Requirements (rules) for specific equipments are being eliminated or downgraded;
- Suppliers are required to manage the risks associated with their system or service within an overall "target level of safety", apportioning safety budgets to sub-systems as appropriate;
- Suppliers of systems (at all levels in the system hierarchy) are required to demonstrate that they have managed risks appropriately, typically through a safety case;
- Regulation is carried out through review and acceptance of safety cases, not checking conformance to more prescriptive rules.

These trends pose a number of challenges:

A. It is often difficult to set and agree safety targets for component systems, especially where the range of accident scenarios to be considered is large;

B. It is difficult to produce a safety case for a low level system, e.g. an ATM air-ground transceiver, as the designer and/or operator of the system will not have full knowledge of the operational environment for the system;

C. It is hard to produce, or assess, a set of inter-related safety cases intended to provide a coherent, overall, argument for a set of interacting systems or services.

These challenges are exacerbated by the fact that the regulatory changes represent a major cultural change for the organisations involved.

2.2 Public Perception of Risk

It has long been understood that expert "computation" of risk can be at variance with the public perception of risk. In some senses this situation is not getting worse, but the media attention to accidents, e.g. the recent railway accidents in the UK, have heightened public awareness of the issues. Unfortunately, various pronouncements, e.g. politicians saying that "expense is no object where safety is concerned", following accidents tends to increase awareness without increasing understanding.

In essence there is only one challenge:

D. To find more effective ways of communicating and discussing risk with the public, especially in the presence of factors, e.g. multiple fatalities, which strongly influence perceptions of risk.

This may be an issue which is particularly significant in the UK at present, but it is clearly an issue of broader concern.

2.3 Application Requirements

There are many ways in which application, or system, requirements are becoming more demanding. The list below identifies some general trends which can be seen in a number of sectors, e.g. aerospace, automotive and defence. As the categories are quite general, some examples are given, without being too specific about system details:

- System complexity is rising exponentially in some sectors. For example the current generation of military aircraft have about 3-5MLoC of safety critical code, but the next generation is likely to have an order of magnitude more;
- Systems are becoming more highly integrated, e.g. adaptive cruise control systems in cars may interface with engine management, transmission (clutch and gear change) and perhaps braking. To analyse the cruise control system requires an understanding of all these related systems, and changes in one of these systems may impact the cruise control functionality;
- Systems are gaining higher authority, and in many cases the user has no ability to intervene in particular aspects of the system operation. Perhaps the most extreme example is that of aerodynamically unstable aircraft, e.g. EuroFighter, which simply cannot fly without the computer systems;
- Systems are also being developed to have much greater autonomy, i.e. the ability to operate without any human intervention. The obvious example is unmanned air vehicles which may deliver weapons without a "man in the loop";
- Finally there is a greater recognition of "systems of systems", that is interacting systems that were not, and perhaps could not, be designed as a whole. Civil airspace is an example of a system of systems so the concept is not new – but the problem increases as the sophistication and degree of interaction of each of the systems increases.

Arguably these issues could all be viewed as posing challenges in their own right. However they can be grouped as follows to form three challenges:

E. Developing techniques for cost-effectively analysing complex and highly integrated systems, in a modular fashion which provides support for efficient analysis of (incremental) change;

F. Developing methods for designing safe high authority, autonomous systems and for analysing their safety;

G. Developing methods for designing safe systems of systems and for analysing their safety.

Arguably points F and G could be merged, but there seems to be an intrinsic difference in the challenge posed when dealing with systems of systems which are not the responsibility of a single design authority.

2.4 Technology Changes

There are many technological changes which affect safety critical systems. The list below identifies just four such changes, focusing on computer hardware and software:

- The rate of change of commercial processor design is enormous. More significantly, the design of modern processors (caches, pipelines, predictive execution) make it very difficult to predict worst case execution time and show that systems meet their deadlines;
- There is a growing trend to use distributed systems, e.g. the so-called integrated modular avionics/ systems (IMA/IMS) in aerospace, in safety related applications. Classical methods of safety analysis and software verification are hard to apply to such systems, and tend to conflict with one of the aims of using such technologies – ease of change and incremental certification;
- There is considerable pressure to use mainstream software development techniques, due to availability of tools and trained staff. At the forefront of this trend is object oriented (OO) technologies and methods such as UML. The issue here is that safety analysis and, to a lesser extent, verification techniques are of limited applicability to OO designs;
- There is increasing use of automatic code generation, and techniques such as systematic reuse. Safety analysis and verification techniques typically assume more classical development processes and, for example, it is unclear how to reuse verification evidence for a software component when it is reused.

There are several ways of addressing the above issues, so there are more than four challenges, e.g. H & I can be viewed as alternatives, or as being complementary:

H. Find stochastic means to analyse timing behaviour of modern processors, placing bounds or distributions on execution times. Develop associated scheduling theory to allow for occasional time "over-runs".

I. Encourage a bifurcation of the processing industry, separating the "main stream" processors from those

intended for critical applications – whilst still achieving growth in processing capability;

J. Develop new safety analysis techniques to support development of IMA/IMS, and change certification practises to allow for incremental certification;

K. Develop hazard and safety analysis techniques for OO development methods such as UML. More significantly, "restrictions" need to be placed on methods such as UML to make it suitable for use in such applications – e.g. to avoid unbounded response times caused by dynamic object creation;

L. Define design restrictions and verification techniques for OO programming languages, e.g. C++, so they can be analysed and tested effectively;

M. Develop strategies for cost-effectively assessing automatically generated code. In particular, there is a need to deal with small changes at a cost proportional to the scale of the change – and this may require alterations in the approach to code generation as well as verification;

N. Develop approaches to "modularising" and reuse of safety analyses to support design and code reuse.

In addition, there is a need to support the transition between "historic" and new processes. However this is so dependent on organisation, culture, etc. that we do not pose it as a separate (technical) challenge.

2.5 Other Observations

There are always concerns about the costs and timescales of developing and assessing safety critical systems and software, and about the difficulties of managing change. These are not mentioned as specific trends or challenges as they have been a concern for some time – however they form part of the "backdrop" to many of the trends and challenges outlined above. Indeed one facet of this is the growing pressure in many industries to use Commercial Off the Shelf (COTS) products. However, much has been written about this subject, and it seems more profitable to focus on other issues.

In the above there has been no attempt to draw out systematic distinctions between the European and US approaches. There are many areas of similarity, especially in technology – for example projects in Europe and in the USA, e.g. JSF, are already committed to using C++, despite the limited knowledge of how to certify code produced in the language.

However there are differences in culture, especially an apparently greater willingness in the USA to take on new technology – and perhaps an excessive reluctance of Europeans to grapple with the issues posed by new technologies, and just to say "they should not be used".

3 Responses to the Challenges

There is a range of activities in research institutions, in industrial projects, and in standards development which are addressing the above challenges. Space does not permit an exhaustive analysis of these activities, instead the opportunity is taken to highlight what the author perceives as key issues. Inevitably there is a bias in such perceptions – hence the statement above that these are personal views.

3.1 General Principles

Rather than try to consider each of the challenges individually it is possible to identify some approaches which give a way of addressing several of the challenges at once. We can view these approaches as being general principles. We consider two such approaches, or general principles.

First, it is desirable to design and analyse systems in such a way that we can achieve compositionality – that is to combine components, or analysis results, so that we can derive properties of the composed system from those of its parts. For some properties, e.g. mass, compositionality normally holds, but it is far from clear how to achieve it for safety. System architecture is a key element, but it is hard to discuss architecture in general so we focus on three aspects of development and assessment:

1. There is current work considering how to achieve modularity in safety cases (Kelly 2000). The aim is to allow parts of a large safety case to be developed separately, then combined, by defining "interfaces" between the separate elements. Thus, for example, it would be possible to tell if a low-level safety case satisfies the requirements of a higher level one, and to assess the impact of change.

2. The OO community has pioneered the ideas of contracts (Meyer 1997), although the principles go back to work by Cliff Jones (Jones 1981). The key aim of this work is to identify what a component guarantees at one side of an interface, and thus what may be relied on, or assumed, at the other. Software can be composed so long as the guarantees are strong enough to meet the assumptions (rely conditions). Work is being undertaken to extend these ideas to IMA, with the complication that it is necessary to deal with failure conditions (Conmy 2001).

3. The idea of contracts is quite general, and some elements of this approach can be seen in the safety case work referenced above (Kelly 2000). There is potential value in extending these ideas to safety analyses. Further, where software is reused and defined in patterns (Gamma 1995) there may be value in linking the associated safety/verification evidence in such a way that it can be composed when the components are composed. This seems to be a natural extension of the contract-based approach to achieving compositionality – but so far as the author is aware, not one which is being followed.

This addresses, to some degree, challenges C, E, F, G, J and N. In another paper at this conference John Rushby (Rushby 2002) discusses an approach to compositional verification, which illustrates some of the above issues, and also highlights the second principle – automation and formality.

The complexity of modern systems means that it is increasingly difficult to analyse them manually – but the majority of fault trees, FMEAs, and other safety analyses are still constructed by hand. To cope with increasing complexity automation is needed – and, where software is concerned, formal techniques are necessary to provide the basis for automation:

- Work has been done on automating safety analysis for specific technologies, e.g. valves and pumps, for some years. As systems engineers move to the use of modelling tools there is value in deriving safety analyses from these models, both to reduce cost and to help ensure that the models and analyses stay in step. Whilst there have been some promising steps forward in this area (Papadopoulos 2000) much more needs to be done, e.g. in dealing with common cause failures, failure recovery, etc. This is an important issue which requires more attention.
- One of the key aspects of the safety process is to identify hazards. With software, a key issue is to identify contributory causes to system level hazards. Typically this is not done to any level of detail, and industrial practice is to determine the SIL level for software. However there is some progress on adapting HAZOP principles (Kletz 1992) to OO designs (Hawkins 2002), expressed in UML (Rumbaugh 1999). UML models, e.g. state charts, are "formal enough" that it is possible to automate the generation of deviations. This work is in its formative stages, but it seems to offer the potential of deriving (formal) safety requirements for software.
- There is also an issue of automated verification. There are effective program analysis tools, such as the SPARK Examiner (Barnes 1997) which enable effective software verification. However the sort of safety properties which arise out of the OO analysis tend to be trace-based, not classical pre- and post-conditions. Thus there is a mismatch between the requirements for formal analysis and the available technology. This is an area where Hoare and He's pioneering work on unified models of programming (He 1998) and practical realisations such as Circus (Woodcock 2002) may be relevant. There seems to be an opportunity for some interesting, focused, research in this area.

These approaches address some of the core "software safety" challenges, i.e. J, K and, to some extent, M.

There is little value in proposing extensions to UML if they are not accepted by the community, and by the tool vendors. Encouragingly, the Object Management Group (OMG) which is responsible for overseeing the evolution of UML (and related activities) has a group working on safety – and it is hoped that this will give a way of bringing the ideas outlined above into widespread use.

Also, although automation is essential, it has drawbacks – some of these are considered by Galloway (2002).

3.2 Specific Initiatives

There are other research activities addressing some of these challenges, e.g.:

- Authority and autonomy – work in the recently launched Defence and Aerospace Research Partnership in High Integrity Real-Time Systems (HIRTS DARP) is considering both how to design and to assess safety of such systems.
- Analysis of OO programming languages – work has been undertaking adapting classical program analysis techniques, e.g. program slicing, to analysis of C++ (Whitford 2002). Whilst this work is encouraging, much more needs to be done to provide effective methods and tools.
- There is considerable work on program timing analysis, including studies of stochastic approaches to worst case execution time (Edgar 2001).

This work addresses, in part, challenges F, H, L and M.

3.3 Standards

There are many system and software safety standards (Hermann 1999) however there has been much debate about the value of these standards. In some sectors there are now active programmes to update or replace standards. Some of the changes which, in part, reflect the above challenges include:

- There is a general move towards objective-based standards which require system developers or operators to analyse for hazards, set safety targets, then provide evidence that these targets have been met. This trend is apparent in the UK Defence sector, and is likely to be reflected in a new issue of DS 00-56. In some cases, e.g. ATM, the regulatory process is also evolving accordingly.
- System safety standards, e.g. ARP 4754 (SAE 1996) are being reviewed and revised. This standard is intended to deal with complex and highly integrated systems - but it is hard to see this in the standard, as currently written. It is to be hoped that issues of compositionality will be addressed in the update to the standard.
- There is an emerging activity to update DO178B (RTCA 1992) to produce DO178C. A driving force behind this is to deal with the issues of OO systems – but hopefully the revision will also incorporate some of the excellent guidance which has already been published to supplement DO178B.

Nothing has been said here about IEC61508 (IEC 1999). So far as the author is aware, after more than a decade in gestation, there is little enthusiasm for an update to IEC61508. However much of what is in IEC61508 seems dated. The control/protection system distinction which is firmly embedded in the standard does not even deal with many current systems, let alone systems of systems. Similarly, the recommended techniques for development of software seem outdated – it is thus hard to see how the standard will remain relevant, unless it is updated.

3.4 Residual Issues

The discussion above has said nothing directly about challenges A, B, and D. This is not to say that they are not important, but they require different approaches. They are largely organisational, managerial and cultural issues – and organisational change, education and training is needed to address these challenges.

Also, challenge I is a commercial issue – and not one which can be influenced, to a significant degree, by research. However it is perhaps worth noting that there are perhaps 10 times more embedded processors than PCs, so the commercial opportunity is significant.

Also, space has not admitted a full discussion of challenge M, analysing auto-generated code. There is a considerable amount of work in this area, perhaps some of the most interesting being that undertaken by QinetiQ on analysing the software in EuroFighter (details are not in the public domain, but some of the technical aspects of the approach are (Arthan 2000).

4 Conclusions

The system and software safety community face a number of challenges due to cultural and technical changes. There is no pretence that this paper has addressed all the issues – but it is hoped that it has set out some of the more pertinent ones. (However one serious omission was the issue of analysing complex data used in safety critical systems.)

An underlying problem for the safety community is that it is always having to "play catch up" as development technology evolves. Whilst it might be argued that this is undesirable, it is unrealistic to expect otherwise. Perhaps the best way to address this is to try to achieve influence with the technology developers, e.g. by participating in bodies such as OMG, and helping to evolve standards.

The paper has tried to indicate relevant research, focusing on European work more than that in the USA. There are some promising lines of exploration, and the possibility that the general principles, e.g. of compositionality, will prove of long-lasting value to the community. Also, there are other promising developments, e.g. the increase in power of model checkers which enables much larger problems to be tackled formally.

There is considerable opportunity and scope for research – but a concern at the difficulty of getting such work recognised and funded. It is relatively easy to "sell" technological research, by comparison with investigating ways of constraining new technologies! It seems that this is likely to be a long-lasting challenge for the safety community – and perhaps one that transcends national and continental boundaries.

5 References

ARTHAN R, CASELEY P, O'HALLORAN C, SMITH A (2000), ClawZ: Control Laws in Z, in Proc. of ICFEM 2000, IEEE Computer Society Press.

BARNES, J. (1997), High Integrity Ada: The SPARK Approach, Addison Wesley,.

CONMY P.M., McDERMID J.A. (2001), High level failure analysis for Integrated Modular Avionics, in Proceedings of Australian Workshop on Safety Critical Systems, LINDSAY P.A. (Ed.).

EDGAR S., BURNS A. (2001), Statistical Analysis of WCET for Scheduling, Proceedings of the IEEE Real-Time Systems Symposium.

GALLOWAY, A.J., McDERMID J.A., MURDOCH J.M., PUMFREY D.J., Automation of System Safety Analysis: Possibilities and Pitfalls, in *Proc. of ISSC 2002*, System Safety Society, Denver.

GAMMA E., HELM R., Johnson R., VLISSIDES J. (1995) Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley.

HIRTS DARP, www.cs.york.ac.uk/hise/darp/.

HAWKINS R, MCDERMID J.A. (2002), Performing Hazard and Safety Analysis of Object oriented Systems, in *Proc. of ISSC 2002*, System Safety Society, Denver.

HERMANN D.S. (1999): Software Safety and Reliability, IEEE Computer Society Press.

HE J., HOARE, C.A.R. (1998), Unified Theories of Programming, Prentice Hall.

IEC (1999), IEC 61508: Functional Safety of Electrical /Electronic/Programmable Electronic Systems.

JONES C.B. (1981), Development Methods for Computer Programs including a Notion of Interference, DPhil Thesis, University of Oxford.

KELLY T P 2001, Concepts and Principles of Compositional Safety Cases, COMSA/2001/1/1, available at http://www-users.cs.york.ac.uk/~tpk/

KLETZ T. (1992), HAZOP and HAZAN: Identifying and Assessing Process Industry Hazards, Institution of Chemical Engineers.

MEYER B.M. (1997), Object-Oriented Software Construction, Second Edition, Prentice Hall.

PAPADOPOULOS Y., McDERMID J.A., SASSE R., HEINER G. (2000), Analysis and Synthesis of the Behaviour of Complex Programmable Electronic Systems in Conditions of Failure, Reliability Engineering and System Safety.

RTCA (1992), Software Considerations In Airborne Systems and Equipment Certification DO-178B/ED-12B

RUMBAUGH J., JACOBSON I., BOOCH G. (1999), The Unified Modelling Language Reference Manual, Addison Wesley.

RUSHBY J.M. (2002), Trends in System Safety – US View. http://www.itee.uq.edu.au/~pal/ACS/SCS02/RushbyTalk.pdf

SAE (1996), Aerospace Recommended Practice (ARP) 4754: Certification Considerations for Highly-Integrated or Complex Aircraft Systems

WHITFORD S. (2002) Software Safety Code Analysis of an Embedded C++ Application, in *Proc. of ISSC 2002*, System Safety Society, Denver.

WOODCOCK J.C.P., CAVALCANTI, A.J. (2002), The Semantics of Circus. In Didier Ber, Jonathan P. Bowen, Martin C. Henson and Ken Robinson, editors, *ZB 2002: Formal Specification and Development in Z and B*, LNCS, 2272:184-203. Springer-Verlag.