Australian Safety Critical Systems Association

Guiding Philosophic Principles on the Design and Acquisition of Safety-Critical Systems

Australian Safety Critical Systems Committee

December 2013

Preface

The Australian Safety Critical Systems Association (aSCSa) is a non-profit philosophical society established to promote the co-operation of academic, industrial, commercial and governmental organisations involved with the practice and advancement of safety critical and safety-related systems, in particular those systems containing software, in Australia.

The activities of the Association are directed towards providing national leadership, facilitation and the co-ordination of professional association activities, and encouraging member contribution relating to safety critical systems.

This document identifies the philosophic principles behind the design and acquisition of Safety-Critical Systems. A Safety-Critical System is one that provides functionality that contributes to the safe operation of a human environment, including any workplace as defined under the WHS Act 2011. A System will normally be regarded as Safety-Critical if it includes physical equipment; monitors or controls physical equipment; or provides information to guide in the monitoring or control of physical equipment.

Background

When systems are developed in ignorance of safety issues, there is increased potential for death, injury or harm to the environment resulting from unintended system behaviour. Both system suppliers and customers should be appropriately aware of safety issues when systems are procured, and take measures to assure that systems are appropriately safe throughout their life-cycle.

Computer systems and embedded computers pervade all aspects of modern daily life, and many implement functions that have the potential to cause death, injury or environmental damage if they do not operate correctly. Some of these systems include emergency service dispatch, car braking, aircraft flight controls, medical delivery systems, railway control, air traffic control and telecommunications systems. These systems are not inherently safe but require safety to be designed into them.

However computer systems technology is evolving, complex, error prone and generally not well understood. It can conceal hazards and even introduce new hazards. It is necessary for developers, procurers, users, maintainers and assessors to understand and manage the particular safety issues involved.

Software plays an important role in most modern systems. It provides an inexpensive and flexible means for introducing very powerful and complex features. This document pays special attention to systems containing Software, as well as to systems that contain Digital Hardware components developed with methods similar to Software development methods. Such components include firmware, application-specific integrated circuits (ASICs), programmable logic controllers (PLCs), and programmable gate arrays (PGAs).

The implementation of system functions by Software (or Digital Hardware) represents some unique risks to safety. Firstly, the flexibility of programming languages and the power of computing elements such as current microprocessors mean that a high level of complexity is easily introduced; thus making it harder to predict the behaviour of equipment under Software control. Secondly, Software appears superficially easy and cheap to modify. Thirdly, the interaction of other elements of the system with the Software is often poorly or incompletely understood.

The complex nature of Software and Digital Hardware means that conventional engineering methods are usually inadequate to establish the necessarily high levels of assurance that such components will behave correctly with respect to their safety requirements. In particular,

this question of Correctness assumes a heightened importance; even in the absence of equipment failure. The complexity of such systems means that it is not feasible to address Correctness by empirical testing alone. Advanced Correctness Techniques are needed to develop assurance in such systems.

As a sound foundation, mature engineering practices should be adopted: including structured planning and design; in-depth analysis at all levels of system development; a safety testing programme; rigorous and extensive documentation; and formal reviews. Additionally, mathematical specification and modelling activities can be used to enhance understanding of system structure and to support convincing Correctness arguments. The highest levels of assurance are to be gained through the inclusion of machine checkable system specifications and models within the safety evidence. Moreover, safety concerns should be considered at all stages of design and implementation.

Safety analysis has greatest value if used to support decision making in the acquisition, design improvement, usage and acceptance of a system, As such safety is best integrated as part of the system engineering process. Safety in design is predictive and operational activity should actively test safety estimates and assumptions, not just 'maintain safety'.

All parties should recognise their legal obligations. Apart from the Trade Practices Act[1] and the common law tort of Negligence, other statutes may be applicable to specific industries and applications. They should be seen to take adequate steps to identify their obligations and to manage the associated risks. In Australia, the Workplace Health and Safety (WHS) Act 2011 has focused attention on these issues. The parties involved in the acquisition and sustainment of systems have a duty of care arising from their legal obligation to take reasonably practicable steps to avert harm to members of the public, as well as their own employees. A breach of this duty could make them liable in the case of an accident.

A basic expectation is that the safety requirements of every system will be identified and assessed in a structured and systematic manner. To ensure that due consideration of safety is always given, the assumption should be that a system should be considered to be unsafe until it is shown to be otherwise.

[1] In Australia software is not considered a "good", but products containing software are. A software house may not be liable for the software it creates.

Principles

Safety-Critical Systems should be developed and evaluated according to sound principles of safety management. This document is based on certain Safety Engineering Principles, which are categorised as management and technical.

Management Principles

***Systems should be considered unsafe unless demonstrated otherwise*.** The aim of this principle is to avert situations where a system is procured and used in service without safety being considered.

The duty of care that the Acquirer and Supplier have implies that it should be explicitly demonstrated (by means of Hazard Analysis) that a system is not safety-critical.

***Safety issues should be addressed early in the development lifecycle, and tracked throughout*.** This is of crucial importance. It is rarely possible for safety to be introduced as an afterthought. It is almost impossible to demonstrate the safety of a system for which safety requirements were not identified at the outset, and in which the design took no account of safety issues. Here 'early in time' means that safety issues should be considered as part of the normal budgeting, tender and contracting process used for system acquisition.

Paying early attention to safety issues is the best way of ensuring that costly maintenance or re-engineering is not required later in the implementation and installation phases. Also, it has been shown that unsafe system behaviours can usually be traced to deficiencies in requirements specifications. Early consideration of safety issues may produce a design that entirely eliminates some sources of hazard. For these reasons, safety management should be applied stringently from the earliest stages of development, and addressed consistently throughout system development.

***Safety assurance depends on visibility of both the product and the process used to produce*.** Visibility means that the approach taken to system development, and all products of it (such as designs and documentation), should be made freely available to other parties involved in safety engineering and management. Visibility is essential if safety-related decisions are to be communicated effectively to others, thus generating valuable comments and criticisms that can be fed back into the development process.

Safety is best achieved through a diversity of people, skills and techniques, and open (fearless) communication. Safety engineering requires a range of skills: sound engineering judgement; domain knowledge; knowledge of appropriate testing methods; mathematical expertise in design verification etc. Such skills do not usually reside in a single person, or perhaps even a single organisation. Also, assurance of safety is increased by having others review (inter alia) specifications, designs, verification results and clarity of documents. Diversity is very important; it is equally important that it be effectively managed.

Safety demands the commitment and co-operation of all parties. The safety management process is very vulnerable. Successful safety engineering requires that all parties contribute fully, and do not withhold important information. It is vital that safety management meetings are run formally and correctly and that factional issues do not affect safety deliberations.

Safety assurance arguments should be independently reviewed and cross checked. Truly independent cross checking and review invokes the principles of diversity and visibility discussed above.

Independent review can reveal problems overlooked by the team that wrote the specifications, carried out proofs, etc. Such problems can arise from hidden assumptions, misinterpretation of requirements, inconsistencies in arguments, etc. Most importantly, technical justifications for safety (including formal specifications and proofs) should be be checked and ratified by parties independent of those who performed them originally.

Safety assurance should be transferable. Because safety management involves a number of agents, and uses people with a range of skills, it is essential to be able to convince others (perhaps without specialist knowledge) of one's conviction that the system is acceptably safe. In the event of a serious accident, it will be necessary to convince investigating bodies how best practice was applied and explain how safety conclusions were reached. In particular, it is important to show that design alternatives were explicitly considered and that the eventual choice between these designs took account of safety issues.

Safety should be demonstrated by means of an auditable and intelligible Safety Case. Safety assurance for a given system is embodied in the Safety Case. The Safety Case aims to provide a detailed technical argument that there is sufficient evidence that the system is safe. The safety argument should be clear to the reader and comprehensive with regards to the operational context of the system. Detailed arguments for safety should be made in such a way as to provide a self-contained record that details and justifies all safety-

related decisions, and provides the evidence that the developed system satisfies safety requirements. The aim is that it will withstand scrutiny by a Court, after-the-fact, whenever that may be.

Technical Principles

Safety is best achieved by using tried and trusted techniques where possible. The engineering of safe systems necessitates a balanced approach that combines vigilant safety management with engineering best practice, as well as extensive domain knowledge with a range of rigorous development and analysis methods. The development process should use proven and best industry practices, previous project experience and the experience of the people involved. Any use of novel methods or technologies for system design should be justified as providing sufficient assurance of safety.

Safety is a whole-of-life issue. Safety should be addressed at each stage in the system life-cycle, from capability development to final retirement.

Safe sustainment depends on careful planning. Safety issues in sustainment activities, such as manufacture, installation, maintenance, and retirement, should be addressed from the earliest stages of system development.

Safety is best achieved by means of an iterative, continuous and evolutionary development process. System design can be, and needs to be, influenced by safety concerns. However, subsequent safety arguments (making up an initial Safety Case) might reveal that the design is flawed in some way from the safety point of view and needs to be modified. The system development cycle should make explicit the need for a number of safety reviews that can lead to changes in the design.

Safety assurance is best achieved if components having safety requirements are kept as simple as possible. System components having safety requirements should be kept as simple as possible so as to make them easy to understand and to reduce the effort involved in assuring system safety. In particular, Software components with safety requirements should be simple in structure and make use of well-understood programming constructs and computing equipment.

Safety assurance is best achieved if components having safety requirements are isolated from the rest of the system. System components having safety requirements should be kept functionally and physically distinct from other system components, wherever possible. This will simplify hazard and other analysis, and again reduce the effort

involved in assuring system safety. More importantly, it is one of the most cost-effective known techniques for achieving designs that are inherently safe.

Safety requires assurance of robustness. The system should be designed so as to ensure that credible fault (or combination of fault) conditions do not give rise to system hazards. Fault-tolerant techniques should be used help the system prevent, detect or recover from equipment failure. The system should also be designed to ensure that credible events in the operational context do not give rise to system hazards.

Safety requires assurance of correctness. A major focus of safety engineering should be to ensure that the failure-free behaviour of the system is safe. It should be demonstrated - to an appropriate level of assurance - that system components will behave correctly with respect to their safety requirements (i.e. be hazard free) under normal (i.e. failure-free) operation.

If a system is not safe in the absence of equipment failure (which is to say unsafe by design), failure-mode and reliability analyses are of questionable value. Furthermore, any features serving to prevent, detect or recover from equipment failure need also to be shown to be correct in their safety preserving functionality

Safety assurance derives from the use of both testing and modelling. Testing alone is not sufficient to understand safety, while modelling alone is not sufficient to demonstrate safety. They are complementary activities; both are necessary to assure system safety. In the most critical cases, where high assurance is required, modelling should include mathematical proof of safety requirements.

This document was prepared by the Executive Committee Australian Safety Critical Systems Association for general distribution within the safety-related systems community.

aSCSa members involved in the preparation of this document were:

Mr Kevin Anderson, Mr Anthony Acfield, Dr Clive Boughton, Dr Tony Cant, Mr Chris Edwards
Mr Brett Martin, Mr Tariq Mahmood, Mr George Nikandros, Mr Derek Reinhardt and Dr Rob Weaver.

Contributions were also received from:
Mr Edmund Kienast, Mr Emil Vlad and Malcolm Watts

Appendix A

Relevant Standards and Documents

The following table provides a list of known standards and other documents where further information for safety-related systems containing software may be obtained. The reader should note that this is not an exhaustive list.

Reference Number	Title
IEC 61508	Functional safety of electrical/electronic/programmable electronic safety-related systems. Part 1: General requirements Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems Part 3: Software requirements Part 4: Definitions and abbreviations Part 5: Examples of methods for the determination of safety integrity levels Part 6: Guidelines on the application of parts 2 and 3 Part 7: Overview of techniques and measures
ISO/IEC 12207	Software Life Cycle Processes
ISO/IEC 15026	System and Software Integrity Levels
NASA-GB-1740.13-96	NASA Guidebook for Safety Critical Software Analysis and Development
UK Ministry of Defence Def Stan 00-25	Human factors for designers of equipment. [UK Ministry of Defence] Part 1 Introduction Part 2 Body size Part 3 Body strength and stamina Part 4 Workplace design Part 5 Stresses and hazards Part 6 Vision and lighting Part 7 Visual displays Part 8 Auditory information Part 9 Voice communication Part 10 Controls Part 11 Design for maintainability Part 12 Systems Part 13 Human Computer Interaction
UK Ministry of Defence Def Stan 00-56 Issue 4	Safety Management Requirements for Defence Systems. Part 1 Requirements Part 2 Guidance
US Dept of Defense MIL-STD-882 E	Department of Defence Standard practice --System Safety
US Dept of Defense MIL-STD-1629	Procedure for Performing a Failure Modes and Criticality Analysis
US Dept of Defense MIL-STD-1472	Human Engineering
US Dept of Defense MIL-HDBK-46855	Human Engineering Requirements for Military Systems, Equipment, Facilities
DEF(AUST) 5679 (Issue 2)	Safety Engineering for Defence Systems
IEC 812	Failure Modes and Effects Analysis & Failure Modes, Effect and Criticality Analysis
IEC 1025	Fault Tree Analysis

Reference Number	Title
IEC 1078	Reliability Block Diagrams
AS/NZS 4360	Risk management
AS/NZS 3931	Risk analysis of technological systems—Application guide
CENELEC EN 50126	Railway Applications — The Specifications and Demonstration of Dependability, Reliability, Availability, Maintainability and Safety (RAMS).
CENELEC EN 50128	Railway Applications — Software for Control and Protection Systems
CENELEC EN 50129	Railway Applications — Safety-related electronic systems for signalling
CENELEC EN 50159	Railway applications – Communication, signalling and processing systems – Part 1: Safety-related communication in closed transmission systems Part 1: Safety-related communication in open transmission systems
NATO STANAG 4044	Safety Design Requirements and Guidelines for Munition Related Safety Critical Computing Systems
NATO STANAG 4452	Safety Assessment Requirements for Munition Related Computing Systems
UK Chemical Industry HAZOPS guide	A guide to Hazard and Operability Studies
ANSI/IEEE 1228	Standard for Software Safety Plans
US Nuclear Regulatory Commission NUREG-0492	Fault Tree Handbook
US Nuclear Regulatory Commission NUREG-CR-2300	Probabilistic Risk Assessment Procedures Guide
US Nuclear Regulatory Commission NUREG CR-4780	Procedures for Treating Common Cause Failures in Safety and Reliability Studies, Vol 1 and 2
DO-178**B**	Software Considerations in Airborne Systems and Equipment Certification [RTCA = Radio Technical Commission for Aeronautics]
ISBN 1 85564 470 3	Systematic Safety Management in the Air Traffic Services", Richard Profit, Euromoney Publications PLC, 1995. [UK National Air Traffic Services]
NOHSC 1010	National Standard For Plant – Code of Practice 12-Jan-1995, Australian Government (Workplace Injury and Prevention Management).
ISO/TS 25238:2007	Health Informatics – Classification of Safety Risks from Health Software
ISO/TR 27809:2007	Health Informatics – Measures for Ensuring Patient Safety of Health Software
ISO 14971:2007	Medical Devices – Application of Risk Management to Medical Devices
IEC 80001-1:2010	Medical Devices – Application of Risk Management for IT Networks Incorporating Medical Devices