Keywords: t/ast/046, computer based safety systems, technical assessment guide, tag, sap, saps, assessing software, multi legged procedure, production excellence, confidence building, technical guidance, software reliability claims, wenra reference levels,

Date issued: 2008-02-**

Review date: 2012-02-**

OG Status: Fully open

Issue No: 002

Approved by:

Computer Based Safety Systems

T/AST/046 - 002

1 Purpose & Scope

1.1 This Technical Assessment Guide (TAG) addresses Safety Assessment Principle [1] (SAP) ESS.27 which deals with the software of computer based safety systems (SSs). Technical Assessment Guide T/AST/003 addresses SSs in general. This SAP is supported by other general and plant specific SAPs. Where a computerised SS is used, because the technology is inherently complex and not amenable to traditional methods of reliability assessment, SAP ERL.1 and SAP paragraph 177 should be applied[1]. ESS.27 presents the elements of a multi-legged procedure that should be used to demonstrate the adequacy of a SS using software based technology which is based on SAP ERL.1 and SAP paragraph 177.

1 Safety Assessment Principles for Nuclear Facilities. (HSE 2006 Edition)

1.2 The TAG contains **guidance** to advise and inform ONR inspectors in the exercise of their professional regulatory judgement. Comments on this guide, and suggestions for future revisions, should be recorded on the appropriate registry file.

2 SAPs addressed

2.1 Key principles

There is only one key principle that deals specifically with this topic, ESS.27. It is, however, underpinned by two more general principles, ERL.1 and ERL.2. The content of ESS.27 is reproduced below for ease of reference.

Engineering principles: safety systems	Computer-based safety systems	ESS.27
Where the system reliability is significantly dependent upon the performance of computer software, the establishment of and compliance with appropriate standards and practices throughout the software development life-cycle should be made, commensurate with the level of reliability required, by a demonstration of 'production excellence' and 'confidence-building' measures.		

2.2 Supporting principles

The key principle is supported by a number of other principles of relevance to computer based SSs. The SAPs identified below are explicitly referenced in the main body of this TAG. Other SAPs are implicitly referenced in Appendix 1 (e.g. A1.1 to ESS.18 and A1.2 to EDR.4). Note that T/AST/003 'Safety Systems' addresses the generality of SS SAPS (e.g. ESS.1 to ESS.27).

1 EKP.3 Defence in depth.

2 EDR.2 Redundancy, diversity and segregation.

3 ECS.1 Safety categorisation.

4 ECS.2 Safety classification of structures, systems and components.

5 EDR.3 Common cause failure.

6 EDR.3 (paragraph 172) Common cause failure/limitation placed on the system.

7 ECS.3 Standards.

8 ERL.1 Form of claims.

9 ERL.2 Measures to achieve reliability.

10 ERL.4 Margins of conservatism.

11 ESS.21 Reliability.

12 MS.1, MS.2, MS.3 and MS.4 Leadership and management for safety.

13 ECM.1 Commissioning testing

3 Relationship to licence and other relevant legislation

3.1 Licence conditions 14 and 15 (preparation and review of safety cases) apply particularly, and Lcs 17 (Quality Assurance), 20 and 22 (modification), 27 (safety mechanisms, devices and circuits), 28 (examination, inspection maintenance and testing) are also relevant.

4 Advice to assessors

4.1 General

1 The need to use computer based SSs should be justified since their inherent complexity means that demonstrating they are adequately safe is difficult (ESS.21). Ideally where diverse SSs are required, and one is computer based, the second one should be provided using a non-computer based system (EKP.3, EDR.2 and EDR.3). The topic of diverse computer based systems important to safety is addressed in Appendix 4.

2 The advice presented below outlines the muti-legged procedure philosophy, production excellence demonstration and confidence building elements, and sources of further guidance. For any production excellence or confidence building element the licensee has the option of non-compliance provided a satisfactory case can be made.

3 Assessors should ensure that the projects' safety management system (MS.1 to MS.4) is adequate for the needs of producing and building confidence in a computer based SS. The organisational arrangements should clearly define roles and responsibilities. The competence of those involved should be demonstrated and adequately managed [15].

15 Managing Competence for safety-related systems (HSE 2007)

4.2 Multi-legged procedure

1 In general the level of dependence on SSs incorporating complex technology should be limited to the order of 1E-1 failures per demand, unless a sufficiently robust justification can demonstrate the appropriateness of a lower value. The application of SAP ESS.27 to computer based SSs is a means of demonstrating the appropriateness of a lower value (i.e. lower than the "order of 1E-1 failures per demand" which is interpreted herein as 0.3 failures per demand or lower). However, with current techniques and taking all relevant factors (see Appendix 3) such as complexity into account, a probability of 1E-4 failures per demand is the best that can justifiably be claimed for computer based SSs [2, 4, 9, 10] (EDR.3 (paragraph 172), ERL.1 and ERL.2). Future advances in software engineering techniques could lead to a situation where a strong case could be made for a lower figure. Such a case would not then be ruled out of consideration.

2 - IAEA Safety Standards Series, Safety Guide No.NS-G-1.1 - Software for Computer Based Systems Important to Safety in Nuclear Power Plants. (2000)
4 - The Tolerability of Risk From Nuclear Power Stations (HSE 1992) ISBN 0-11-886368-1.
9 - BS IEC 61226:2005. Nuclear power plants - Instrumentation and control systems important to safety – Classification of instrumentation and control functions.

10 – The use of computers in safety-critical applications – Final report of the study group on the safety of operational computer systems, (HSE 1998) ISBN 0-7176-1620-7

2 Assessors should be wary of attempts to reduce a fault sequence frequency through risk reduction claims involving multiple low integrity computer based SSs (ERL.4). The concern is that inappropriately large amounts of credit may be claimed in total by combining several small amounts of risk reduction, the effects of such combination often being hidden within what is called the initiating event frequency value. Although functionally such low integrity systems are SSs in these circumstances they are usually embedded in other large systems that are classified as safety related. Unless the claim for all contributing 'risk reduction' elements of safety related instrumentation (SRI) taken together in a given fault sequence is clearly pessimistic, and not better than of the order of 1E-1 failures per demand in total, the separate SRIs should be treated as though they were SSs and assessed as such. Where such SRI is identified and the claimed risk reduction is 0.3 failures per demand or lower then they should be subjected to ESS.27.

3 Assessors should consider whether it is desirable to apply ESS.27 to computer based SRI such as control systems. Each separate computer based SRI that requires a reliability claim of better than of the order of 1E-1 failures per year, interpreted herein as 0.3 failures per year or lower, should be subjected to ESS.27.

4 ESS.27 has two key legs, production excellence demonstration and independent confidence building [1,5 and 6]. The philosophy of the multi-legged procedure is that system acceptance centres on the demonstrated high quality production and an independent searching examination of the systems' fitness for purpose that finds no significant number of errors.

5 - Software-based protection for Sizewell B: the regulator's perspective. Nuclear Engineering International, September 1991

6 - Software-based protection for Sizewell B: the regulator's perspective. Proceedings of the International Conference on Electrical and Control Aspects of the Sizewell B PWR, September 1992.

5 Where the confidence building activities reveal a significant number of software errors the quality of the production process is brought into question and the argument for system acceptance is weakened. Any remedial action must go further than software error correction, the licensee must restore confidence in the system by thoroughly examining the production process to establish that similar errors have not arisen elsewhere. The licensee must then demonstrate that the corrected software is fit for purpose by a sufficient repetition of the confidence building measures.

6 The production excellence and confidence building elements are outlined in sections 4.3 and 4.4 . Judgement will be required in determining the detailed scope and rigour of the elements (including applicable techniques and measures) as informed by the category of the functions implemented by the system, the systems' classification and its integrity requirements (ECS.1, ECS.2 and ECS.3). Guidance may be obtained from standards [8] on the techniques and measures applicable to defined integrity levels and this topic is discussed further in Appendix 3. The procedure should be underpinned by a comprehensive review of issues taking account of past precedents [5 and 6] (ERL.1).

8 - IEC 61508:2002. Functional safety of electrical/electronic/programmable electronic safety-related systems.

7 When considering past precedents assessors should note that the position of techniques and measures in the production excellence and confidence building legs is not immutable. Increasing the strength of one leg might lead to a reduction in requirements for the other; but both legs need to be sound.

8 Early agreement of the safety case elements including the applicable techniques and measures should be sought with the licensee. Following agreement the target should be to retain the fixed status of the agreed set for the projects' duration. Nevertheless, if some relevant technology were to emerge with significant potential to improve real plant safety then this would become subject to consideration, in terms of its 'reasonable practicability', for adoption.

4.3 Production Excellence

1 The multi-legged procedure requires a demonstration of production excellence, covering initial specification through to the finally commissioned system, comprising the following elements:

a) Thorough application of technical design practice consistent with current accepted standards for the development of software for computer based SSs e.g. [7] (ECS.1, ECS.2 and ECS.3);

7 - IEC 60880:2006. Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions.

b) Implementation of an adequate quality assurance programme and plan in accordance with appropriate quality assurance standards (ECS.3 and A1.27 to A1.29);

c) Application of a comprehensive testing programme formulated to check every system function (ECS.3), including,

- **prior to installation on site, the verification of all phases of the system production process and the validation of the integrated system against its requirements specification by persons qualified for the task and not involved in the specification and design activities,**
- **following installation on site, a demonstration that the safety system, in conjunction with the plant, performs to requirements, this demonstration being devised by persons other than the system specifiers, designers or manufacturers,**
- the use of prototypes as appropriate,

- **a programme of dynamic testing, applied to the complete system, which is capable of creating confidence that the system meets its reliability requirements.**

NB. Text in clauses a), b) and c) above that is identical to that contained in SAP paragraph 360 is identified by bold.

2 Excellence of production requires that the best practices current at the time be used to avoid errors, detect and remove those not avoided and provide in-built system tolerance to those not detected (ECS.3 and EDR.1). It is important that an unambiguous, comprehensive and verified specification of the system requirements, covering the full system life-cycle is available from the earliest stages of the production process.

3 Should the production excellence assessment identify weaknesses in the production process, compensating measures should be applied to address these. The type of compensating measures will depend on, and should be targeted at, the specific weaknesses found (ESS.27 paragraph 362). Compensating measures should not be confused with the confidence building activities discussed below. The confidence building activities should be determined at the outset of the project and applied as far as possible to the final production software (i.e. after the completion of any compensating measures).

4 Although not mandatory, the case for production excellence is greatly assisted by evidence of the systematic application of national and international e.g. [7] standards, coupled with a case by case justification of non-compliances.

5 It must be demonstrated that the integrated equipment is capable of satisfactory operation in situ through an extensive pre-operational test (A1.87). During this stage the equipment will be operational (with frequent testing) while being progressively integrated with other plant as part of commissioning (ECM.1).

6 It is recognised that the dynamic testing could be part of the confidence building measures if not undertaken by the manufacturer. The number of dynamic tests should be informed by statistical testing considerations. This might mean that tens of thousands of tests are required for confidence in the dependability of the system to be built up (e.g. of the order of 50,000 tests with no failures for a 1E-4 probability of failure on demand demonstration to 99% confidence). Consideration should be given to the feasibility of generating a statistical estimation of system reliability (as informed by research).

4.4 Confidence building

1 The confidence building leg provides an independent and thorough ‘reasonably practicable’ assessment of the SSs’ ‘fitness for purpose’ (ERL1), comprising the following elements:

a) **complete and preferably diverse, checking of the finally validated production software by a team that is independent of the systems suppliers, including**;

- **independent product checking providing a searching analysis of the product** including application of static analysis (A1.75) and other techniques such as reverse engineering and desk-top review,

- **independent checking of the design and production process including all activities needed to confirm the realisation of the design intention** such as interpretation and adequacy of the specification, application and compliance with design specification, methods and standards; and

b) **independent assessment of the test programme, covering the full scope of test activities** (e.g. verification, validation, commissioning and dynamic testing) including traceability of tests to specification and confirmation that the specification is met.

NB. Text in clauses a) and b) above that is identical to that contained in SAP paragraph 361 is identified by bold.

2 The independents should determine and justify their approach, the rigour of which should be proportional to the needs of the task. Wherever practicable the independent assessors should employ checking methods dissimilar to those used by the systems producer(s) (e.g. Dynamic Testing, MALPAS etc. as applied to the Sizewell B PPS).

3 The chosen confidence building techniques should be demonstrated to be adequate prior to application. New and unproven techniques will need especially comprehensive demonstration in order to provide confidence that the route chosen will be successful.

4 It is important to stress that the product checking should be applied only to the finally delivered product - that is after completion of the manufacturers verification and validation. The licensee may, however, be able to make a case for applying techniques to code that has completed the manufacturer's independent verification.

5 The timing of the confidence building activities should ensure that their application and correction of findings does not become entwined with the production process. The conduct of the initial proving of confidence building techniques should not compromise the assessment of the delivered product.

6 The confidence building measures should be conducted by suitably qualified and experienced persons who are independent, established in a working regime compatible with the objectives of the task and free from conflicts of interest.

7 The use of "independent" should be interpreted as meaning not connected with the systems' suppliers, either designer or manufacturer. The organisational arrangements including allocation of responsibility for confidence building tasks to the licensee's procurement group (i.e. the intelligent customer group for the system) and to independent assessors in a different line management chain should be justified.

8 The independents who are separate to the licensee's procurement group should have financial autonomy and managerial independence to reach their own conclusions on the adequacy of the system and its safety case. They should not be restricted in technical scope and have a frank reporting regime with a route to the licensee's main Board.

9 The independents should report findings and their reconciliation within the safety case. All reporting of confidence building measures including findings and their reconciliation within the safety case should be auditable.

4.5 Technical guidance

1 Technical guidance can be found in Appendix 1 which is a record of an earlier detailed technical assessment guide. The guidance provided in Appendix 1 should be used in conjunction with the later IAEA Safety Series [2] guidance, European Commission guidance document [3], the report "Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations – Revision 2007" and IEC standards as appropriate (see below). The term "protection systems" is used in Appendix 1, however, the guidance is also applicable to other SSs and SRI containing software falling within the scope of ESS.27.

3 - European Commission – Nuclears safety and the environment – Common position of European nuclear regulators for the licensing of safety critical software for nuclear reactors, European Commission's Advisory Experts Group, The Nuclear Regulators Working Group, Task Force on Safety Critical Software - Version 11. (May 2000)

2 Technical guidance and regulatory advice is available in the IAEA Safety Series [2], a European Commission guidance document [3] and the report "Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations – Revision 2007". In particular, they contain guidance on a number of topics not fully addressed or considered in the earlier assessment guide (Appendix 1), such as; use and validation of pre-existing software, computer security, use of tools, software diversity, safety demonstration and formal methods. Assessors should ensure that the recommendations and guidance of these documents [2], [3] and [13] have been considered. While they are directed at Nuclear Power Plants much of their content is applicable to other nuclear facilities.

3 Further technical guidance can be found in IEC standards such as IEC 60880 [7] (for software of SSs), IEC 61508 [8] (grading of techniques and measures for software development - see Appendix 3) and IEC 61513[11] (general requirements for systems). The IEC standards should be consulted for topics not addressed by the guidance provided by Appendix 1, the IAEA Safety Series[2], the European Commission guidance document [3,] or the report "Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations – Revision 2007".

11 - IEC 61513:2001. Nuclear power plants - Instrumentation and control systems important to safety – General requirements for systems

Appendix 1 Technical Guidance for Assessing Software Aspects of Digital Computer Based Protection Systems

Contents

General

A1.1 Computer systems provided as part of the protection system should not be used to perform non-protection functions. Specifically, control functions and the protective actions claimed against their failure should not employ common hardware or software.

A1.2 The single failure criterion should be applied to all parts of the protection system including both the systems and applications software of the computer based part. Where the application of this principle leads to a requirement for diverse software in an otherwise redundant system then arguments in support of applying a high quality single software solution can be considered, since high quality (see Para A3.4) means that the software is not the determinant of system reliability.

A1.3 Computer based protection systems should be specified, designed, manufactured, tested, installed, commissioned, operated and maintained in an orderly, disciplined and coherent manner. When carrying out the assessment clear evidence should be sought that best available practice has been used, e.g. structured design methods, software modularised, each phase verified, clear documentation, auditability and validation testing.

Organisation

A1.4 Only reputable companies should be used in all stages of the lifecycle of computer based protection systems. Each should have a demonstrably good track record in the appropriate field. Such companies should only use staff with the appropriate qualifications and training for the activities in which they are engaged. Evidence that this is the case should be provided.

A1.5 The licensee should ensure that the appropriate standards, procedures and personnel are used at all stages in the software lifecycle to produce the required integrity

for the computer based protection system. Those persons carrying out this Safety Authority role should be independent from the designers, implementors, the verifiers and validators.

A1.6 The Safety Authority should submit written proposals to ONR on the methods to be employed to demonstrate how the required level of integrity will be met. Regular progress reports should be submitted throughout the software development lifecycle.

A1.7 For all stages of the software lifecycle the verifiers should be independent from the designers and implementors. All communication between the two groups should be recorded in writing; unrecorded verbal communication should not be allowed. Such communications should be logged and archived to provide an auditable trail and ensure that system concerns, together with their resolution, are clearly and accurately documented.

A1.8 The validators should be independent from the team producing the requirements specification. All communication should be recorded in writing; unrecorded verbal communication should not be allowed. The validation team should review the specification for correctness and completeness, and should devise and implement the test strategy that demonstrates that the system meets its specification.

System Specification

A1.9 A requirements specification should be produced that specifies what is required of the system but does not contain any information on how the specification is to be implemented.

A1.10 Where practicable the requirements specification should be animated through the process of executing the specification. Animation aids the process of communicating ideas about a complex set of requirements which should be understood by all professionals involved in the protection system project.

A1.11 The specification should list standards to be used for all aspects of the software lifecycle. All non-compliances should be justified. The standards listed in the Paragraph A1.27 and Reference 7 are currently regarded as appropriate for the development of protection system software. Their exclusion from the specification should be justified.

A1.12 Credit can be claimed for the use of a formal mathematical specification language since this will increase confidence in the correctness of the design and also assist in the static analysis phase. The language used should have well defined syntax and semantics that allow logical reasoning about the specification. The specification should be validated by logical reasoning and peer review of that reasoning if credit is to be claimed for the use of a formal mathematical specification language.

A1.13 Timing budgets, volumes of data, data rates, peak data loads, maximum values of parameters and calculation accuracies should be specified.

System Design

A1.14 The system should be designed using a structured approach so that the design process is clearly visible. Designs that facilitate software analysis are preferred. For

example, the essential safety kernel should be securely separated from other features such as hardware failure diagnostics and operator communications. Concurrent interactions with other systems should be avoided. Also, the provision of software for the detection and reporting of hardware failure should not be allowed to complicate the system.

A1.15 For a project involving software of any large size or complexity, Computer Aided Software Engineering (CASE) tools should form an integral part of all stages of the software lifecycle.

A1.16 Due consideration should be given to the frequency of sampling of input parameters such that varying measurements are truly represented within the system and events are observed within sufficient time to take the necessary action.

A1.17 The system as a whole should be designed as far as practicable to reveal failures. Diagnostic software can be used to detect hardware failures (see section on "Hardware Fault Monitoring"). Defensive programming techniques and special hardware such as watchdogs, divide-by-zero interrupts etc. can be used to detect software failures. Hardware failures which are not self-revealing should be deemed to exist until the next proof test that would reveal them. These proof tests should be defined.

A1.18 Facilities should be incorporated for an operator to determine the state of each input variable and output function.

A1.19 The program should include a test arrangement to enable in-service functional checks of the system hardware to be made from the input variables to the output action. A frequency for applying this test should be proposed with a justification based on the type of faults detected and their rate of occurrence. The insertion or removal of any test equipment should not require manual action to reinstate the system; however, there should be confirmatory feedback to the operator.

A1.20 The program together with its stored constants, should be stored in a secure, reliable and permanent manner such that timely automatic restart is possible following power supply failure or any such detectable and recoverable failure.

A1.21 A system design aim should be to prevent the unintentional operation of a safety device due either to operator error or to a fault in the equipment.

A1.22 Unsolicited changes in plant status should be detected and where possible prevented.

A1.23 The need for manual intervention (e.g. for calibration or adjustment) should be kept to a minimum.

A1.24 The use of features not specified by the manufacturers (i.e. undeclared instructions or clock rates) should be prohibited unless their use is fully justified.

A1.25 A systematic method should be used to identify critical system modules. Examples of such techniques would be Failure Modes and Effects Analysis (FMEA) and Fault Tree Analysis (FTA).

A1.26 Failure of one part of a redundant scheme should not adversely affect other parts of the system.

Quality Assurance

A1.27 Quality Assurance should be at least to the standards and guidance defined by the following documents.

1 BS 5882:1980: Specification for a total quality assurance programme for nuclear installations.

2 IAEA Safety Series No. 50-C/SG-Q : Quality Assurance for Safety in Nuclear Power Plants and other Nuclear Installations

3 BS EN ISO 9001:1994 : Model for quality assurance in design, development, production, installation and servicing.

4 IAEA: Technical Report Series No. 282, Manual on quality assurance for computer software related to the safety of nuclear power plants.

A1.28 An agreed Quality Assurance Programme and Plan for the hardware and software, covering all stages of specification, design, manufacture, installation, commissioning, operation and maintenance, should be available. Attention should be given to organisational structure, error reporting and corrective action, computer media control (issuing and storage), testing, supplier control, record keeping and documentation. Standards and procedures should be available for these aspects.

A1.29 Periodic, independent audits should be conducted to ensure conformance to the QA Programme and Plan.

A1.30 Procedures should be in place to ensure that the effects of modifications on all parts of the system are fully assessed.

Software Configuration Management

A1.31 The software including all support software should be covered by a suitable, readily understood configuration management system (CMS) throughout the lifecycle. The system should be protected by a security system that prevents unauthorized access. Users should only be able to access parts of the CMS for which they have authority. The security system should be designed to a high standard and be regularly reviewed for possible breaches. The system should not be linked to an off-site network.

A1.32 During all stages of the software lifecycle software should be stored in a secure place and the media clearly identified as to its status. Strict procedures should be employed to ensure that the correct version is issued and that it has not been subjected to unauthorized modification either accidentally or intentionally. The issued version should be verified against a master copy.

A1.33 Version control and the issuing of correct versions requires the setting up of formal procedures. All software copies should be clearly and uniquely labelled with at least title,

version no. and creation date. Consideration should be given to the inclusion of the above data in the code so that this can be checked by the program.

A1.34 Program modifications should be controlled by change control procedures in which modifications are evaluated, approved and properly documented. Access to the software should be strictly controlled so that a programmer cannot make an unauthorised change at any stage in the lifecycle (especially post V&V).

Software

A1.35 Software reliability cannot be quantified with a high confidence level because it is associated with systematic events which do not lend themselves readily to statistical expression. Therefore, it is necessary for it to be produced to a standard such that it is considered not to contribute significantly to the overall unreliability of the system, allowing considerable margin for uncertainty. This means that only software produced to the highest standards should be employed. Unfortunately, the contribution that each development and testing technique makes to the overall reliability of the software cannot be quantified but some intuitive guidance is given in Appendix 3.

A1.36 Defensive programming techniques should be employed where known states and parameter ranges occur. For example, where the state of a valve can be either open or closed, a check should be made that, after a finite change-over time, one state or the other is achieved. This would be done by checking open AND NOT closed, or closed AND NOT open. In the case of parameter ranges, if the output of a temperature computation returns a negative result or an unrealistic answer then an error should be flagged.

A1.37 Assertions should be inserted into the code and the system should be made as fault tolerant as possible. For example, data errors occurring in inputs and on transfer between modules should be trapped and alarmed; instrument readings should indicate when they are off scale; correct output address selection pre- and post-operation should be verified.

A1.38 The system should be designed such that, where practicable, the correct operation of the output addressing system is verified. The design should also include means for checking that the final control element has achieved the desired state in the required time.

A1.39 If complicated calculated protection functions are required the program should be written so that a simple function(s) will provide an error check and back-up action.

A1.40 Because of the problem of unauthorised accessing via data links (hacking), computer based protection systems and all supporting systems should not be connected to off-site networks either directly or through another computer system unless an adequate safety case has been made to prove that there is no capability for unauthorised access to the protection system.

A1.41 It should be demonstrated that the protection system and all supporting systems are protected against unauthorised access. Password protection alone is insufficient.

A1.42 All support software and systems used in the production and testing of programs (compilers, assemblers, linkers, loaders, configuration control systems, static analysers,

test harnesses, coverage analysers, etc.) should themselves be qualified or otherwise suitably justified.

A1.43 Programs should be organised on a modular basis, in a manner which minimises the chance of errors, facilitates independent verification (e.g. restricting the design of a module to one clearly identified function that requires only minimum interaction with other functions) and minimises the impact of changes. Modules should not exceed a limit specified for the system (e.g. 50 or 100 statements, or the amount of coding which can be placed on one page) without written justification.

A1.44 The program should run in a fixed sequence pattern and should not generally employ interrupts. Where interrupts are used they should be justified in terms of demonstrably simplifying the program. Their usage and masking during time and data critical operations should be checked for correct operation in-service and be well documented. The hardware and software should be designed to detect both unserviced and missing interrupts. All interrupt vectors should be initialised whether used or not. Unused interrupts should point to an error reporting procedure and should initiate protective action.

A1.45 Appropriately engineered facilities should be provided to ensure that operation of any testing facilities, manually or automatically initiated, should neither trigger spurious operation of tests, nor protective action nor degrade protective system reliability. Where testing causes inhibition of protective action, testing facilities should be regarded as part of the protection system and should be designed and engineered to the same standards.

A1.46 The system should employ a hardware watchdog of demonstrably adequate reliability. This should be serviced by a program operating at the base level of priority so as to detect problems such as programs locked in loops or deadlocks.

A1.47 On detection of failure the system should adopt a safe state and as soon as practicable report the condition to the operator.

A1.48 Careful consideration should be given to the power fail/restart procedures since initial conditions and phasing may produce an unsafe state in redundant systems

A1.49 The compiler of the programming language used to write the software should be validated, its error detection capabilities defined and the code restricted to a safe subset. The same version of compiler should be used on all the software. All software must be retested if a new version of the compiler is used, unless it can be demonstrated that the machine code has not changed. Any compiler code optimisation should be simple and provably correct. Also, the same optimisation options should be used on all the software. Array bound checking should be included.

A1.50 Assembler language should be kept to a minimum and subject to the same strict controls as a high level language. Good practices that would be introduced by a compiler should be implemented in assembler language, e.g. array index checking.

A1.51 Any proposed operating system should be shown to comply with this assessment guide.

A1.52 Coding standards should be mandatory and should as a minimum cover naming conventions, house style, level of commenting and software production procedures

A1.53 A high level, strongly typed, programming language that prohibits the unsafe mixing of variable types is preferred. Variables and data should be explicitly declared and assigned. The program layout should aid understanding. Assembler and machine code inserts into high level language code, and commands that allow direct memory manipulation should not be allowed.

A1.54 The coding standards should prohibit the following practices:-

1 Tricks, recursion (unless it produces simpler code than by other means), re-entrancy and unnecessary code compaction - it should be noted that recursion is not amenable to static analysis;

2 Program halts - unless it is not possible to action (by bringing the plant to a safe state) and report the error;

3 Dynamic storage allocation - causes problems with static analysis, since memory overflow and overwriting of variables cannot be checked;

4 Dynamic instruction changes;

5 Multiple use of variables - variables should not be used for more than one function;

6 Default conditions in case statements - all cases should be covered, default should be failure condition;

7 Multiple module entry points - single entry and return points should be employed (other than error return where required): where a module is a subroutine or procedure, parameter passing should be the means of conditioning;

8 Branching into loops;

9 Complicated calculation of indexes;

10 Equivalence statements - especially referring to a COMMON or global area;

11 Procedure names used as procedure parameters;

12 Modification of loop control variables within the loop;

13 Similar names for variables and procedures;

14 Assembler and machine code inserts into high level language code;

15 Direct memory manipulation commands - for example, PEEK and POKE in BASIC;

16 Computed branching (note that good program structure such as use of IF-THEN-ELSE and CASE statements is required).

A1.55 The coding standards should encourage the following:-

1 Arithmetic expressions reflecting the equations they represent;

2 Rate of change checking, where possible;

3 Dynamic checking for overflow, underflow and compatibility between precisions;

4 Transfer of data by parameter passing - the unnecessary use of global variables should be discouraged;

5 Constraining of loop iterations to a declared maximum value (this ensures loop termination);

6 Explicit initialising of all variables;

7 Representing of constants by symbolic names;

8 Meaningful variable names and types;

9 Nesting limited to a maximum depth of four (4);

10 Minimisation of procedure and subroutine parameter numbers (ideally not to exceed four (4));

11 Careful use of indirect addressing so that the ultimate address is as clear as possible;

12 The performance of arithmetic operations in binary fixed-point arithmetic, unless the use of floating-point arithmetic can be demonstrated to contribute to safety - in the latter case, the hardware and software used to implement floating point functions should be suitably qualified;

13 Checking of computer array subscripts for dimensional range;

14 Detailed commenting (see Paragraph A1.90).

A1.56 Programs and fixed data (including operational data) should be held in read only memory so that they cannot be changed on-line either intentionally or due to a software error. Changes to in situ ROM should be subject to strict controls to avoid incorrect data entry (program logic should NOT be changed in this manner); consideration should be given to the use of special error checking software within the protection system, and within the modification device for off-line use, to check the correctness of any necessary fixed data changes.

A1.57 All messages should be tagged with a serial number to enable the loss of a message to be identified.

A1.58 The protection system should automatically record, in a secure manner, all fixed data changes within the system for subsequent analysis.

A1.59 Protection system loading should be computed on-line and be available without requiring special procedures.

A1.60 The software should be subjected to static analysis by the designers. The static analysis tool should be of proven design, fully qualified and diverse from the static analyser used by the independent assessors.

Hardware Considerations (Software Aspects)

A1.61 Since computer system components involve complex design with the potential for introducing errors, only proven components should be used. Proof of suitability should include either evidence of extensive installation years free of design faults or rigorous formal proof of the design plus comprehensive type testing.

A1.62 Where redundancy is claimed the hardware should be physically and electrically segregated.

A1.63 Information transmission between redundant parts of a protection system should be avoided. Where this is not fully practicable then the system should be arranged so that malfunction of any redundant part does not adversely affect the other part

A1.64 Redundant portions of the system should not be operated synchronously unless it can be established that there are safety advantages in so doing.

A1.65 Hardware modules should be clearly labelled with a unique part number and serial no. Also, the bin positions should be correspondingly labelled. Special precautions should be taken where modules are configurable; only reliable means of configuring should be used, e.g. soldering or wire wrapping. Features should be provided which detect (or prevent) the insertion of an incorrect module, absence of a required module or incorrect insertion of a correct module. Special attention should be paid to the prevention of the incorrect connection of cables due to either wrong orientation or location.

Hardware Fault Monitoring

A1.66 The memory should incorporate automatic (preferably hardware based) parity and checksum testing to detect bit change errors. Attempts to write to read only or protected memory should be detected.

A1.67 Where possible, operations should be timed and an error indicated when the pre-set time for the operation is exceeded.

A1.68 Correct operation of analogue-to-digital converters should be checked by reading reference voltages.

A1.69 Communications messages should incorporate checksums and message lengths.

A1.70 At initialisation the system should check its ability to write to and read from all installed memory and address all required hardware modules

A1.71 Where possible, power supply voltage levels should be checked. Also, the ability of a signal line to perform its specified function should be tested automatically.

A1.72 Software routines should be incorporated which check the integrity of standard arithmetic operations.

Verification and Validation

A1.73 The V&V team (independently from the designers) should verify that each stage of the software lifecycle is complete and correctly implements the requirements of the previous stage, in particular at the following stages:

1 Requirements specification,

2 Software and hardware specification,

3 Software and hardware design,

4 Software production and hardware manufacture,

5 Software and hardware off-site test,

6 Software and hardware on-site test,

A1.74 All verification and validation activities should be comprehensively documented to enable auditing. Only deficiencies should be indicated by the V&V team; they should not recommend design changes. The extent of V&V coverage should be declared and justified.

A1.75 As part of the validation process the software submitted by the supplier should be subjected to independent assessment using an acceptable Static Analysis method. An example of such a method is MALPAS.

A1.76 As a precaution against the introduction of erroneous or unwanted code, the machine code of the system should be translated into a form suitable for comparison with the specification (reverse engineering). Any errors should then be corrected. This process can be carried out either manually or using software aids. For example, the system machine code might be dissassembled using a software tool and the resulting code translated into, say MALPAS Intermediate Language (IL). This IL could then be compared with the IL generated from the source code by using the Compliance Analysis feature of MALPAS.

A1.77 The V&V team should check that the software standards have been correctly implemented and that as far as possible errors have not been introduced. In particular they should check for:-

1 A complete, unambiguous and correct requirements specification,

2 Correct implementation of all software modules,

3 Timing and data errors,

4 Good software structure,

5 Adequacy of commenting and documentation,

6 Appropriate use of fault tolerant and defensive programming features,

7 Adequacy of software Failure Modes and Effects Analysis or Fault Tree Analysis,

8 Adequacy of testing both off and on site,

9 Deadlocks,

10 Qualification of supporting software,

11 Correct and complete implementation of modifications,

12 Consistency throughout the whole lifecycle.

Testing

A1.78 The software should be qualified by an agreed comprehensive test programme which should encompass all modes of operation (particularly maximum loadings) and ranges of variables under conditions which are as representative of the operating environment as possible. This will require a range of diverse tests conducted both off-site and on-site.

A1.79 Testing should be conducted in a disciplined and orderly manner from module testing, through progressive integration and testing, to fully integrated system testing. During testing, the status of all outputs should be monitored to ensure that unintentional changes do not occur.

A1.80 Software should be tested by an independent team who independently devise and carry out the tests, and have no unrecorded verbal contact with the designers or implementors. Any errors should be documented and their correction tracked. Errors should be analysed for cause and lack of early detection.

A1.81 Modification to existing software to correct errors or change functionality could potentially introduce errors. Such changes will necessitate requalification (known as regression testing). The degree of retesting will depend upon the amount of change that has taken place, its complexity and the complexity of the system. As a guide, for a non-trivial change, all modified modules and their interfaces to other modules should be subjected to repeat tests. Any deviation from full path coverage of a module and interface should be justified on the grounds of non-interaction.

A1.82 Protection systems should be subjected to a wide range of tests. Methods such as random and back-to-back testing are powerful techniques for revealing many types of error.

A1.83 The proposed tests should exercise the software across the full range of the requirements specification paying particular attention to boundary conditions. Also, by inspecting the code, tests should be devised which seek to detect such errors as stack overflow, divide by zero, numerical overflow, timing conflicts including bus contention problems etc.

A1.84 Consideration should be given to cliff edge effects to ensure that the system does not generate unsafe conditions when taken marginally beyond the specification limits. Adequacy of spare CPU time, scan times and data transfer rates should be demonstrated.

A1.85 Off-site test coverage should be such that all lines of code and (at system integration) all module calls are exercised at least once; this to be confirmed by coverage analysis. Operations on data items should be exercised at least once unless a bounding case can be made. All time critical sections of code should be tested at least once under realistic conditions. Where calculations are performed in the software their result should be checked against pre-calculated values. The fault tolerance of the system should also be tested by subjecting the system to appropriate tests, including those that are only implied by the specification.

A1.86 Once installed a test program should exercise cyclically all aspects of the hardware (including inputs and outputs), over an extended period of time (typically 400 hours) to establish that the hardware is sufficiently reliable to commence plant commissioning.

A1.87 On-site testing of the full system (software and hardware) over an extended period (the duration will depend on system complexity, for example 12 months for a reactor protection system) will be required before active commissioning is permitted. Actual plant inputs should be used as far as possible; the validity of simulated inputs should be justified and tests should demonstrate the correct operation of simulated items. Test coverage analysis should be provided. The reliability of the software together with a confidence estimate should be computed during the on-site tests, although the value obtained should not be interpreted as a measurement of in service reliability since however well prepared the test input combinations will not represent in scope or distribution the in service combinations. This probabilistic software reliability value should be compared with that obtained by judgement in the application of deterministic development methods (see A3.4) and if not smaller in terms of failures per demand than the judged value, then it must be taken that the applied methods have failed to be effective. In such a case the judged value should be revised in the light of actual performance.

Documentation

A1.88 Documentation, in English, covering all aspects of the computer based protection system should be readily available (taking into account the need). It should reflect the as-built state, i.e. be correct, up-to-date and under a documentation management regime.

A1.89 The procedure for writing and quality control of all software should be set out in accordance with the quality assurance arrangements.

A1.90 Programs should be well commented with information on the basic program function in the header and more detailed comments for program specialists in the body of the listing.

A1.91 Sufficient documentation should be available so as to provide a complete audit trail for all stages of the system life cycle from conception to termination of use. All decisions should be documented. Documents should be signed-off as checked and approved by authorised personnel. Modifications should be signed-off and include date, details of change and reason for change.

A1.92 Sufficient documentation should be available to enable suitably qualified staff to understand and maintain the system. The following documents should be available as a minimum:-

1 Operators' Manual - details facilities and use of the system in a non-technical language. Normal and abnormal operation should be described.

2 Systems Description Manual - details in full the functions of the different parts of the system and the facilities offered to the operator and maintainer.

3 Standard Hardware Documents - handbooks, instruction manuals and drawings for all bought-in items.

4 Special Hardware Documents - full circuit diagrams, complete with component layouts and indications of normal signal levels plus a full parts list. Design principles, circuit operation, test procedures and fault diagnosis information should also be included.

5 System Hardware Documentation - details of equipment arrangement, first line fault diagnosis information including the use of any test procedures and module replacement procedures.

6 Standard Software - all software documentation for bought-in software such as compilers, linkers, utilities etc.

7 Systems and Applications Software - details of the operation of the software system and individual programs should be given. The content should be such that system maintainers can readily gain an appreciation of the overall system and individual programs. Descriptions of all functions, major task and programs together with their interrelationships. The following data should be provided with each program module as a minimum:-

i) Purpose of module,

ii) System linkage,

iii) Initialisation,

iv) Data and parameter input,

v) Data and parameter output,

vi) Data and files accessed,

vii) Data and files modified,

viii) Timing,

ix) Interrupts,

x) Hardware interfaces,

xi) Program structure,

xii) Functional description of the program,

xiii) Error returns,

xiv) Size of module,

xv) Any special considerations.

8 Detailed memory maps showing disposition of programs, data areas and spare memory.

9 Data areas:-

i) Details of purpose,

ii) Format,

iii) Type,

iv) Cross-references to programs using data,
v) size of file/array.

10 Listings - a complete listing of the source code fully commented.

11 A complete memory dump.

12 Control and data flow should be illustrated either in graphical or pseudo-code form for the overall system and for individual modules.

13 Software and hardware part numbers and versions in use.

14 Test results.

15 QA Plan Compliance Record.

16 List of software and hardware design standards.

17 FMEA/FTA for hardware and software.

18 V&V results.

Training

A1.93 All staff associated with the operation of the system should have received adequate training in their particular role and be able to demonstrate competence prior to being permitted access to the system.

Operation

A1.94 Formal procedures should be in place for the documenting and timely reporting of residual software errors during the operational stage. The safety implications of such errors should be reviewed by appropriately authorised personnel.

A1.95 The system and procedures should be arranged so that only an approved program as a whole can be loaded into the system.

A1.96 Suitable arrangements should be made for the secure storage of copies of the software that is to be loaded in the event of a system failure. Due regard should be given to all possible means of corruption of the software. Particular attention should be paid to the secure storage of pre-programmed memory devices if they are used. Means should be provided to verify that the software has not been corrupted prior to its use both before and after loading. Automatic means are preferred.

A1.97 All necessary hardware and software for maintenance of the system should be in place prior to active commissioning.

Modification and Maintenance

A1.98 No program modifications should be necessary as part of system operation and no facilities should be provided for this purpose. Protection system software should not be altered to overcome operational or maintenance difficulties, other than on an off-line system using agreed modification procedures. Engineered facilities should be provided where there is a need to change specific system parameters such as trip settings, calibration constants etc. for operational reasons. The available range should be limited to values supported by the safety case. A display of current values or states of such parameters should be provided.

A1.99 Modified software should be tested on a computer system identical to the installed system.

A1.100 Histories of all hardware modules should be maintained so that reliability claims can be monitored. Certificates of conformance should be available and a policy for faulty hardware modules of "repair and return" or "new for old" should be employed so that operating histories can be maintained, traced and properly interpreted.

A1.101 The repair and modification of both hardware and software should only be undertaken off-line. Such repair and modification should only be performed by personnel with proven competence in the relevant field and knowledge of that part of the system.

A1.102 Mean Time To Repair (MTTR) has an effect on total system reliability, therefore, there should be sufficient spares and test equipment, and adequately trained staff to

enable the proposed figure to be met. The maximum on-line repair time allowed before the reactor or process must be shutdown should be stated and justified.

A1.103 Modifications arising at any time should be subjected to a full V&V process with test coverage analysis.

Appendix 2 Definitions

A2.1 ANIMATION - A system for creating a simulation of a specification such that its behaviour can be observed in real time

A2.2 ASSERTION - A condition inserted into a program where failure to satisfy during execution or analysis indicates an error.

A2.3 ASSEMBLER LANGUAGE - A computer programming language based on mnemonics which have a one to one correspondence with the computer's instructions. Assembler language is the most difficult language in which to write computer programs.

A2.4 BIT - The binary digit 0 or 1 used by computers to represent data or instructions.

A2.5 CASE - Computer Aided Software Engineering is a generic term to describe software systems used to specify, design, code, test, modify and document computer programs. Software Engineering is the process of specifying etc. computer programs using formal procedures and techniques designed to reduce errors and improve program structure.

A2.6 CODE - The individual instructions or statements of a computer language.

A2.7 COMPILER - A computer program used to translate a higher order language program into its relocatable [i.e. containing addresses relative to the start of the module] or absolute [i.e. the actual addresses] machine code equivalent.

A2.8 COMPILATION - The process of translating a high level programming language into instructions that can run a computer.

A2.9 DIVERSITY [SOFTWARE] - The provision of dissimilar means of achieving the same objective. The introduction of diversity in any aspect of a system or its manufacture/ production will reduce the likelihood of common mode failures. Systems can be considered as having varying degrees of diversity according to the number of these different aspects which have been achieved by dissimilar means. Diversity in software covers the computer instruction set but also the programming language, support software, design, and all staff involved in the life cycle. Where a claim is made that very high reliability has been achieved through software diversity then it must be shown that dissimilar means have been employed in all aspects of the system and its manufacture/production. Any divergence from this should cause the claim to be down rated.

A2.10 DATA RATE - The speed with which information can be transmitted along the chosen path. For example, the data rate along a coaxial cable could be referred to as 10 Mbits per second.

A2.11 DEFENSIVE PROGRAMMING - Incorporation of mechanisms into a program that detects and respond in a predetermined safe manner to erroneous data values and control flow, i.e. program logic.

A2.12 EFFICIENCY - The extent to which a program performs its intended functions with a minimum of consumption of computing resources, including computing time. Efficient

use of computing time is an important consideration in systems responding to external processes and events.

A2.13 FAULT TOLERANT - The ability of a software system to operate in a predetermined safe manner in the presence of a limited number of hardware or software faults.

A2.14 FIRMWARE - computer programs and data loaded in a class of memory that cannot be dynamically modified by the computer during processing.

A2.15 HARDWARE. The physical components that make up a computer system.

A2.16 HARD WIRING. A system design method where each signal path is dedicated to a single parameter or function.

A2.17 HIGH LEVEL PROGRAMMING LANGUAGE - A set of instructions for a computer which are more closely related to English and therefore can be more easily understood. BASIC is such a high level programming language.

A2.18 IMPLEMENTORS - That organisation or part of an organisation, whose responsibility is to take a design specification and produce an operational finished article. As opposed to the Designers who use the requirements specification to produce a design specification.

A2.19 INTERRUPT - The process whereby a sequence of instructions is terminated and a another sequence is executed. Upon completion of the second set, control is returned to the first set at the point of interruption. This operation is regarded as dangerous in safety related software since parameters can be updated by the interrupting program only to be changed by the interrupted program.

A2.20 LOADER - A computer program that reads an object program or its data into the main storage area.

A2.21 LINKER - A computer program used to create one load module from one or more independently translated object modules by resolving cross references among the object modules and possibly relocating elements.

A2.22 PEAK DATA LOADS - The maximum rate of data production requiring handling by a system. For example, large amounts of data could require handling during an accident.

A2.23 PROGRAM - A set of computer instructions which perform a particular function.

A2.24 RE-ENTRANT ROUTINE - An interrupt called routine that is already running at the time of the interrupt. (A re-entrant routine is also recursive.)

A2.25 RECURSIVE ROUTINE - A routine that may be used as a sub- routine of itself, calling itself directly, or being called by another sub-routine, that it itself has called.

A2.26 SAFETY AUTHORITY - That organisation or part of an organisation, which is independent from all others involved in the software lifecycle, and is responsible for ensuring that the safety requirements of the design are achieved.

A2.27 SAFETY SYSTEM - A system which acts in response to a fault to prevent or mitigate a radiological consequence.

A2.28 SAFETY RELATED - An item important to safety that is not part of a safety system.

A2.29 SAFE SUB-SET - The sub-set of instructions from the total set of a programming language which can be regarded as amenable to static analysis and perform the same operation regardless of context.

A2.30 SOFTWARE LIFECYCLE. All stages from conception to final disposal through which a software product passes.

A2.32 TIMING BUDGET - The time allocated for the completion of a particular process or procedure.

A2.33 VALIDATION & VERIFICATION – Validation; The process of testing and evaluation of the integrated computer system (hardware and software) to ensure compliance with the functional, performance and interface requirements. Verification: The process of ensuring that a phase in the system life-cycle meets the requirements imposed on it by the previous phase.

A2.34 VALIDATABILITY - The extent to which a computer-based system can be shown to conform to its requirements specification. Consideration should be given to the facility with which analysis of the requirements to establish test criteria and evaluation against those criteria can be performed.

A2.35 VARIABLE - A parameter within a program whose value will be change by the action of the program. A declared variable is specified at the beginning of a program as being used. It is also characterised at this stage. This process reduces the likelihood of errors.

A2.36 VOLUME OF STORED DATA - The maximum amount of data to be stored over a system's operational life

Appendix 3 Software Reliability Claims

A3.1 As noted in the main text, reliability claims for computer based SSs to which SAP ESS.27 is applied can range from of the order of 1E-1 (interpreted herein as 0.3) to 1E-4 pfd. The following table provides indicative guidance on the reliability claim that ONR would expect to be applied, given application of the multi-legged procedure of ESS.27 including use of techniques and measures as outlined in the IEC 61508 standard (i.e. where the SS is implementing the safety function). ESS.27 need not be applied for claims higher than 0.3 pfd, though licensees should still apply appropriate safety standards such as IEC 61508 SIL 1, so far as is reasonably practicable.

Table 1 - Safety Integrity levels: target failure measures for a safety function and reliability claims for SSs

IEC 61508 Safety Integrity Level (SIL)	IEC 61508 Probability of failure on demand (pfd) Range	Minimum acceptable nuclear safety case probability of failure on demand (pfd) value
4	$\geq 10^{-5}$ to $< 10^{-4}$	10^{-4}
3	$\geq 10^{-4}$ to $< 10^{-3}$	10^{-3}
2	$\geq 10^{-3}$ to $< 10^{-2}$	10^{-2}
1	$\geq 10^{-2}$ to $< 10^{-1}$	10^{-1}

A3.2 The above table is in agreement with public statements on justifiable claims for computer based systems in nuclear reactors [2, 4, 9, 10 and 13]. It also represents a conservative and cautious approach which reflects the high standards expected within the nuclear sector. The maturity of current techniques, their practicability of application to real life SSs and other factors such as computer based system complexity, configurability, novelty and maintainability also have to be considered. Taking all of the above into account, a probability of 1E-4 failures per demand is considered to be the best that can justifiably be claimed for computer based SSs used in circumstances where the consequence in the event of failure of the SS could potentially involve very large releases of radioactive material. However, it is recognised that advances in system design and software engineering techniques might lead to a situation where a strong case could be made for a lower figure. Such a case would not then be ruled out of consideration.

A3.3 Assessors should consider applying similar reliability claim limitations (i.e. when referring to the levels stated in IEC 61508-1:1998 Table 3) to SRIs where they are subjected to ESS.27, for example, as shown in table 2 below. ESS.27 need not be applied for claims higher than 0.3 failures per year, though licensees should still apply appropriate safety standards such as IEC 61508 SIL 1, so far as is reasonably practicable.

Table 2 - Safety Integrity levels: target failure measures for a safety function and reliability claims for SRIs

IEC 61508 Safety Integrity Level (SIL)	Minimum acceptable nuclear safety case probability of dangerous failure per year (pdfy) value
4	10^{-4}
3	10^{-3}
2	10^{-2}
1	10^{-1}

A3.4 In some software development projects there might not be a significant difference between the techniques and measures adopted at the different IEC 61508 SIL levels. In addition, the selected confidence building measures might add significant weight to the safety case. Assessors should, therefore, consider the techniques and measures adopted by the system producer in relation to those of the next higher IEC 61508 SIL, the rigour with which they were applied and the strength of the independent confidence building measures. Following this assessment the assessor should determine whether the nuclear safety case reliability claim (see tables 1 and 2) of the next higher level could be substantiated. For example, an IEC 61508 SIL 1 system that substantially and rigourously adopts the requirements of SIL 2 combined with a rigourous confidence building programme might allow a nuclear safety case claim of 10-2 pfd.

A3.5 It should be noted that tables 1 and 2 above apply to specific types of systems important to safety and specific circumstances where the highest level of confidence of achievement of target reliability figures (pfd/pdfy target) that it is reasonably practicable to achieve is required. The factors affecting the decision to apply the tables include both:

1 circumstances where the consequence in the event of failure of the system important to safety could potentially involve very large releases of radioactive material, and

2 systems important to safety that are very complex (which would include all software based systems).

For those circumstances and situations falling outside (1) and (2) above, the adoption of the pfd/pdfy of the lower target failure measure at the boundary of the IEC 61508 SIL should be considered in respect of the effectiveness of what has been done to control or avoid systematic failures. However, the licensee should provide appropriate evidence that it had achieved this by effective and rigourous adoption of all appropriate techniques and measures for the SIL and that this is appropriate given the circumstances of the specific case.

A3.6 In assigning reliability values to a computer based system it is appropriate to consider the hardware and software aspects separately since their behaviour is quite different. Hardware failures are predominantly random, hence coincident failures have a low probability of occurrence unless occasioned by a common cause. Hardware reliability can, therefore, be improved by the use of simple redundancy, although a limitation is imposed due to the incidence of common cause failures. Software failures are due to systematic faults. Their occurrence depends upon the values of input and stored parameters causing paths containing faults to be executed. Here simple redundancy

gives no reliability improvement since each program sees the same input values. The software equivalent of hardware redundancy is achieved by software diversity, since only by such means can coincident failures be rendered less likely. Where a claim is made that very high reliability has been achieved through software diversity then the assessor should consider the guidance provided in Appendix 4 and the document "Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations"

A3.7 The overall system reliability in terms of failures per demand is the sum of the separate software and hardware contributions.

A3.8 Where credit is claimed for self-revealing fault detection and automatic testing in reliability calculations, the contribution attributable to system and operator response, and MTTR should be specified and justified. Failures which are not self-revealing should be deemed to exist until the next test that would reveal them.

A3.9 The claimed effectiveness of the fault detection system should be justified. A claim of 100% should be rejected.

A3.10 When evaluating the numerical reliability claimed for the hardware of a computer based system, limited credit (depending upon the software production method employed) can be claimed for the diagnostic software designed to detect hardware faults.

A3.11 Licensee numerical reliability claims would be enhanced by the application of statistical testing techniques. An important consideration here is whether the system facilitates statistical testing and the generation of direct evidence to support reliability claims. A review of research into statistical testing should be undertaken so as to inform the steps needed to generate a convincing statistical reliability claim. Note that the number of required tests (representative operational transients randomly selected from the input space) could run to tens of thousands (e.g. of the order of 50,000 tests with no failure for a 1E-4 probability of failure on demand demonstration to 99% confidence).

Appendix 4 Use of Diverse Computer Based Systems Important to Safety

A4.1 A major issue on Sizewell B was justifying the high level of risk reduction provided by the Reactor Protection System (RPS) comprising the Primary Protection System (PPS) and Secondary Protection System (SPS). The risk reduction requirement for the RPS was 10-7 pfd (e.g. typically 10-3 pfd from the PPS combined with 10-4 from the SPS to give 10-7pfd). The justification relied on a claim that the SPS was a simple hardware based system.

A4.2 Demonstrating that two complex computer based protection systems are "independent" (e.g. would not tend to fail on the same demands) and hence the reliability numbers for each can be multiplied together remains an open question despite significant research (e.g. undertaken by City University and Bristol University). Hence, where a high level of risk reduction is required that is greater than the accepted common cause cut-off limit for a single computer based SS (i.e. 10-4 pfd for a software based SS where the consequence in the event of failure of the SS could potentially involve very large releases of radioactive material) then our current expectation is that ideally a simple hardware based secondary SS should be provided.

A4.3 However, ONR would consider the use of two diverse computer based safety systems to implement a safety function requiring high reliability (e.g. such as a reactor protection system comprising primary and secondary systems) provided the guidance in SAP ERL.1 paragraph 177 is followed. This “special case” procedure should also be considered for application where high reliability is required from a combination of a computer based safety system and a computer based safety related system. In this context “high reliability” is taken as a reliability requirement for a safety function that is lower than the common cause cut-off limit (i.e. 10-4 pfd/pdfy for a computer based system important to safety where the consequence in the event of failure of the system could potentially involve very large releases of radioactive material). For ease of reference the content of SAP ERL.1 paragraph 177 is repeated below:

> **“177 Where reliability data is unavailable, the demonstration should be based on a case-by-case analysis and include:**
>
> **a a comprehensive examination of all the relevant scientific and technical issues;**
>
> **b a review of precedents set under comparable circumstances in the past;**
>
> **c an independent third-party assessment in addition to the normal checks and conventional design;**
>
> **d periodic review of further developments in technical information, precedent and best practice.”**

A4.4 Each of the diverse computer based systems important to safety would need to meet the requirements of SAP ESS.27. With regard to the implementation of SAP ERL.1 in the context of diverse computer based systems important to safety the case should include:

- application of best technical design practice (e.g. functional and equipment diversity),
- adoption of appropriate nuclear standards[e.g. 7, 12,13 and 14] for the production and assessment of diverse computer based systems,
- an independent assessment of all factors that could lead to CCF,
- examination and implementation of relevant research.

12 – IAEA Safety Standards Series, Safety Guide No.NS-G-1.3 – Instrumentation and Control Systems Important to Safety in Nuclear Power Plants. (2002).
14 – IEC 62340 Ed. 1.0 2007-12 – Nuclear Power Plants – Instrumentation and control systems important to safety – Requirements to Cope with Common Cause Failure (CCF).

A4.5 The section on diversity contained in the document "Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations" is particularly relevant (e.g. providing recommendations on functional diversity, use of current best practice, simplicity of software design, analysis of CCF within the safety demonstration, use of dissimilar means throughout the development lifecycle and conservative reliability claims etc.).

Appendix 5 WENRA Reference Levels

A5.1 There is one section of the WENRA Reactor Safety Reference Levels (Ref.[16]) that is directly relevant to the use of computer based safety systems (i.e. Appendix E – Issue: Design Basis Envelope for Existing Reactors, clause 10.10). The software aspects of clause 10.10 are implemented through application of SAP ESS.27 and the guidance shown in this TAG (other SAPs address hardware aspects). Clause 10.10 from Ref.16 (January 2008 revision) is shown below and references into the main body of this TAG are provided in Bold text.

> "10.10 Computer based systems used in a protection system, shall fulfil the following requirements:
>
> - the highest quality of and best practices for hardware and software shall be used; **[4.3]**
>
> - the whole development process, including control, testing and commissioning of design changes, shall be systematically documented and reviewed; **[4.3]**
>
> - in order to confirm confidence in the reliability of the computer based systems, an assessment of the computer based system by expert personnel independent of the designers and suppliers shall be undertaken; **[4.4]** and
>
> - where the necessary integrity of the system cannot be demonstrated with a high level of confidence, a diverse means of ensuring fulfilment of the protection functions shall be provided. **[4.1.1, 4.2.1 and Appendix 4]**"

References

1 Safety Assessment Principles for Nuclear Facilities. (HSE 2006 Edition)

2 IAEA Safety Standards Series, Safety Guide No.NS-G-1.1 - Software for Computer Based Systems Important to Safety in Nuclear Power Plants. (2000)

3 European Commission – Nuclears safety and the environment – Common position of European nuclear regulators for the licensing of safety critical software for nuclear reactors, European Commission's Advisory Experts Group, The Nuclear Regulators Working Group, Task Force on Safety Critical Software - Version 11. (May 2000)

4 The Tolerability of Risk From Nuclear Power Stations (HSE 1992) ISBN 0-11-886368-1.

5 Software-based protection for Sizewell B: the regulator's perspective. Nuclear Engineering International, September 1991.

6 Software-based protection for Sizewell B: the regulator's perspective. Proceedings of the International Conference on Electrical and Control Aspects of the Sizewell B PWR, September 1992.

7 IEC 60880:2006. Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions.

8 IEC 61508:2002. Functional safety of electrical/electronic/programmable electronic safety-related systems.

9 BS IEC 61226:2005. Nuclear power plants - Instrumentation and control systems important to safety – Classification of instrumentation and control functions.

10 The use of computers in safety-critical applications - Final report of the study group on the safety of operational computer systems, (HSE 1998) ISBN 0-7176-1620-7

11 IEC 61513:2001. Nuclear power plants - Instrumentation and control systems important to safety – General requirements for systems.

12 IAEA Safety Standards Series, Safety Guide No.NS-G-1.3 – Instrumentation and Control Systems Important to Safety in Nuclear Power Plants. (2002).

13 Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organisations – Revision 2007.

14 IEC 62340 Ed. 1.0 2007-12 - Nuclear power plants - Instrumentation and control systems important to safety – Requirements to Cope with Common Cause Failure (CCF).

15 Managing competence for safety-related systems (HSE 2007).

16 WENRA Reactor Safety Reference Levels, January 2008