CNRA/CSNI WORKSHOP ON
LICENSING AND OPERATING EXPERIENCE OF COMPUTER-BASED I&C SYSTEMS
Hluboká nad Vltavou, Czech republic
25-27 september, 2001

Software Important to Safety: The New IAEA Safety Guide and The Common Position of European Nuclear Regulators [1]

Courtois Pierre-Jacques,
Association Vincotte Nuclear (AVN), Brussels

Abstract: An overview of some distinctive aspects of two international documents which provide guidance on the design and the licensing of computer based systems important to safety prepared by a contributor to both documents. The paper takes a look at their coherence and complementarities, at their strong and original points, and at the issues they leave open.

1. Introduction

In September 2000, two documents were published simultaneously:

the International Atomic Energy Agency (IAEA) Safety Guide " *Software for Computer based Systems important to Safety in Nuclear power Plants*", Safety Guide NS-G-1.1,

and the report EUR 19265 EN of the Nuclear safety, regulation and radioactive waste management unit of the European Directorate General for the Environment[2]: "*Common position of European nuclear regulators for the licensing of safety critical software for nuclear reactors* ", categorized as a consensus document.

For practical reasons, we will refer to the first document as **SG**, and to the second as **REG**.

These two documents are important, each in its own way. **SG** is a *new* safety guide of the Agency, the first of its kind to focus specifically on software. **REG** is a first consensus

[1] A version of this paper also appeared in Nuclear Engineering, vol 47, N°570, January 2002, pp.37-40, under the title : "Hard guidelines made for computer software".
[2] The activities of this unit are now within the Directorate General for Energy and Transport

document from nuclear regulators on licensing practices specifically addressing safety critical software and produced under the auspices of an international institution. Both documents have taken some innovative viewpoints, sometimes on thorny issues.

2. Background

Both documents have been the object of intensive work by experts and consultants, and the result of a long process of meetings and revisions.

The work on the **SG** was initiated as early as April 1991, when a group of distinguished international experts in software engineering – including Professors D.L. Parnas and N. Leveson - met in Vienna. They alerted the Agency that its current guidance did not address software issues –already considered as quite critical at the time - and they drew a list of topics for future technical reports. Their recommendations resulted in the publication, in 1994, of the technical report 367 [2]:" Software Important to safety in Nuclear power Plants" to which about fifteen experts actively contributed with papers and during lively meetings.
In April 1995, a group of four experts met again in Vienna to identify – on the basis of the technical report – the possible contents of a future safety guide. Their report advised the Agency to focus (i) on software issues, (ii) on the interface between the regulator and the licensee and (iii) on guidelines not on how to design but on what is needed to demonstrate adequacy of the design.
Then followed a series of alternate advisory group meetings, technical group and consultant meetings: October 95 and in particular November 1996, when 24 experts from 17 countries met for a week to review, comment and debate the current version of a safety guide. A fourth version of the document was submitted to the Agency Nuclear Safety Standards Advisory Committee (NUSSAC[3]) in October 1997 and accepted as a draft for a safety guide project. A subsequent version was then sent to Member States for comments. Fourteen Member States sent 465 comments which were dealt with in two consultant meetings. Two final consultant meetings took care of these comments, most very positive and constructive.
The version 7 of the guide was endorsed by the NUSSAC in its meeting of October 1999, by the Advisory Commission on Safety Standards (ACSS[4]) in December 1999, and published less than one year later.

The genesis of the **REG** document in many ways followed a similar pattern, albeit within the smaller community of the European nuclear regulators.
The 1995-2000 activity programmes of the Nuclear Regulator Working Group (NRWG) and of the Reactor Safety Working Group (RSWG) of the European Commission Directorate General XI (Environment, Nuclear safety and Civil Protection) were set up within the framework of the 1975 and 1992 resolutions of the Council of Ministers on the technological problems of nuclear safety.

[3] now NUSSC, which stands for Nuclear Safety Standards Committee
[4] now CSS, which stands for the Commission on safety Standards

In 1994, the NRWG and RSWG working groups launched a task force of experts from nuclear safety institutes with the mandate of "reaching a consensus among its members on software licensing issues having important practical aspects". From October 1994 to June 1997, the task force met three times a year. The task force selected a set of key issues, produced 64 contributions and made 7 revisions of a draft document which was eventually accepted by the NRWG/RSWG as a EC report [3] publicly available and open to comments (the report was also sent by the commission for comments to 30 prominent international experts). In March 1998, the project ARMONIA (**A**ction by **R**egulators to Har**mon**ize Digital **I**nstrumentation **A**ssessment) was launched with the mission to prepare a new version of the document which would integrate the comments received and would deal with a few software issues not yet covered. In May 1999, after 5 residential meetings of ARMONIA and 25 paper contributions, a revision 10 was submitted to the Task Force for comment and approval. Eventually, in May 2000, after two additional meetings, a revision 11 was presented and approved by the NRWG[5], provisionally classified under the category "consensus document". It was made available through the europa server:
http://www.europa.eu.int/comm/energy/en/nuclearsafety/reports.htm - Nuclear installation safety
and published as report EUR 19265 EN in September.

3. Why this guidance is useful

SG and **REG** are guidance documents that aim to meet specific needs, not met by other standards.

The eighties left the nuclear I&C community somewhat traumatized by several modernization projects involving safety critical software that had experienced abnormal delays and costs. The lack of experience, of practical methods and of interactions between the nuclear and other industrial and software engineering communities were probably some of the causes of these problems.

Two observations emerged from these experiences.
If guidance was available to *design* software based safety systems, little or none was available to address the specific issues raised by the *licensing* of highly critical software. As far as software was concerned, regulators and licensees were abandoned to improvisation.
The second observation – somewhat antinomic but salutary – was that software is not per se safe or unsafe. Software is only one component of the system. Checking the software is (i) not sufficient and (ii) is dependent on the environment. The notions of *computer safety case* and of *computer safety demonstration* resulted from this observation and received increased attention.

The section on background already pointed out that the former of these two observations was an essential motivation for launching the **SG** and the **REG** projects.
The second observation was also instrumental. Paragraph 1.5 of the **SG** states: "*The objective of this Safety Guide is to provide guidance on the collection of evidence and preparation of*

[5] The RSWG was discontinued in 1999. The NRWG is now made up of Nuclear Safety Authorities from the European Union countries as well as from candidate countries to the EU from Central and Eastern Europe.

documentation to be used in the safety demonstration of the software of computer based systems important to safety in nuclear power plants." The **REG** document also has an introductory section which addresses the safety plan, the safety strategy and demonstration: "*…All the subsequent recommendations contained in this report are founded on the premise that (such) a safety plan exists and has been agreed upon by all parties involved. The intent herein is to give guidance on how to produce the evidence and the documentation for the safety demonstration and for the contents for the safety plan."*

This intent to focus on the evidence required by the safety demonstration of software - makes **SG** and **REG** documents complementary to other guidance which – like the IEC 60880 - focuses on requirements for each stage of the software design, development, and V&V processes.

To sum up, in some of its more distinctive aspects, this guidance:

- Addresses regulator and assessor concerns, potential sources of conflict in licensor/licensee negotiations, and identifies grounds for mutual agreement,
- Addresses the safety demonstration (safety case) rather than the system design,
- Emphasizes the need for documentation (**SG**) and identifies sources of evidence (**REG**).

4. A same Scope…

Both documents address the software of systems important to safety as the IAEA guides define them, but focus on safety systems. Both documents recognize the difficulty of defining possible relaxations on requirements for safety related software based systems. However, whenever possible, both documents explicitly specify recommendations which apply only to safety systems and thus indirectly admit possible relaxations for safety related system software:

SG: in paragraphs 1.6, 3.15, 4.17, 5.19, 5.21, 5.35, 6.7,

REG: the clauses (more than 30) that apply to safety system software only are mentioned in a specific section (section 1.10), together with specific clauses for safety related systems and examples of relaxations for new and preexisting software.

The **SG** relaxations essentially concern security requirements against the external world, the nature of the independence required from the V&V teams, requirements on the specification of functional and non-functional safety requirements, requirements for statistically valid tests commensurate with the required reliability, and the dedication of safety systems to safety functions.

Moreover, **REG** admits additional relaxations on requirements for the assessment of pre-existing software (PSW), on dependability and documentation requirements for tools, on requirements for software produced by tools, on the required safety culture, on staffing levels, on computer system design (isolation, data protection,…), on programming and coding directives, on statistical testing, on software change control and maintenance, on calibration and testing requirements in operations.

5. … But Different Structures and Contents.

While the scopes of the documents are identical, their structures differ.
Their different structures reflect the fact that the **SG** is an emanation of designers, operators and regulators, while the **REG** gives a more focused regulator's common viewpoint.

The **SG** is organized in 15 sections (see appendix 1). The first four sections provide recommendations on preconditions of a software based system development project, on the management of safety, and on the planning of the project.
Sections 5 to 15 are dedicated to the individual phases of the development life cycle, up to post-delivery modifications. Each section is generally structured in the following pattern. In each section, under the heading "RECOMMENDATIONS" there is a set of recommended principles or concerns that should be addressed in this phase. Under the heading 'DOCUMENTS', there is a list of documents to be produced as an output from the phase and advice is provided concerning the contents of these documents. Also, some general recommendations are given concerning the attributes and presentation of the products of the phase. In all parts, the intent is not to provide an exhaustive description of all the material that will be needed for development purposes; instead, the intent is to summarize the principles, material and its attributes that are most important for the safety demonstration.

The **REG** document is organized around a selected set of technical issues which were considered difficult by the task force of regulators and of utmost importance to the licensing process. These issues cover a consistent set of licensing aspects right from the inception of the life cycle up to and including commissioning.
These issues were partitioned into two sets: "Generic Licensing Issues" and "Life Cycle Phase Licensing Issues". Issues in the second set are related to a specific stage of design and development process, while those of the former have more general implications and apply to several stages or to the whole system lifecycle. Each issue area is dealt with in a separate chapter of the report (see appendix 2).

Why two documents within the same scope?
The two documents have one part in common: the annex on preexisting software of the **SG** reproduces a section of the **REG**. Otherwise the contents are quite different. The **SG** is an inclusive account of all the aspects involved in the safety demonstration of a software based system, from the very initial phase before the start of a project up to and including the post-delivery modifications. The requirements and recommendations result from an agreement between experts representing different stakeholders (designers, utilities, regulators) and aim at completeness. They seek to establish an essential and comprehensive basis for the safety demonstration, assuming that more detailed requirements may need to be incorporated according to national practices, or on a case-by-case basis. In contrast, the REG focuses on a set of technical licensing issues only, for which it gives the common viewpoint of regulator's experts. The emphasis is on technical requirements, recommendations, and acceptance criteria, at a detailed level whenever necessary and possible.

6. Non - Prescriptiveness

None of these documents is of course legally binding.
Every IAEA guide foreword clearly states: *"The IAEA standards are not legally binding on Member States but may be adopted by them, at their own discretion, for use in national regulations in respect of their own activities."*

The **REG** executive summary is no less clear: " *While the Common Positions are intended to convey the unanimous views of the Task Force members on the guidance that the licensees need to follow as part of an adequate safety demonstration, it should be remembered that this guidance is non-prescriptive. Therefore, its specific application depends on each national regulatory authority. Throughout the document these common positions are expressed with the auxiliary verb "shall". The use of this verb for common positions is intended to convey the unanimous desire felt by the Task Force Members for the licensees to satisfy the requirements expressed in the clause. The Common Position requirements can be regarded as a common denominator of practices in the member states represented in the task force*."

It is the IAEA usage to use the verb "should" to express all recommendations in a safety guide, with the understanding that it is necessary to take the measures recommended or equivalent alternatives to comply with the requirements stated in the Safety Requirements publications. In the preparation of the **REG**, it was found useful to use, like the IEC standards, "shall" and "should" statements. "shall" statements are used in **REG** for expressing the common positions (as defined above), and "should" statements for *recommended practices.* Recommended practices are recommendations supported by most, but which may not be systematically implemented by all of the members states represented in the task force. In contrast, the set of requirements of a *common position* was – I believe - regarded as being "technically necessary" (the same technical necessity that would leave no choice to a railway safety guide but to require that the gates "shall" be closed before the train is on the railway crossing).

7. Some salient points

Both documents make steps forward by showing consensus on certain prevention or precaution measures to deal with software issues that either always proved difficult, or are new because they are engendered by new software practices. Below are a few examples, with no intent of being complete.

On automatic code generation
As far as licensing is concerned, there has always been much debate between proponents of systems generating code from application specifications and those more familiar with the classical development cycle. Here is the position of the **SG:**
Code can be produced from the system specifications in various ways that are essentially combinations of two distinct approaches: the classical development process through stages of specifications and design…, or the use of code generating tools which receive as input a high level language application-oriented description of the system. The choice between these two approaches depends on the tools and resources available to the parties involved in the project, and should, in

particular, take into account trade-offs between design and the demonstration of the dependability of tools. The recommendations of this Section (i.e. the section on software implementation) *apply to all possible combinations of the two approaches.(9.2)*

And on software requirements for code generated by tools (**SG**):
Software requirements are the subset of the computer system requirements that will ultimately be implemented as computer programs…The verification of the software requirements against the upper level requirements is an important step in the licensing process…(7.1)
If the computer system requirements are sufficiently detailed and their documentation is sufficiently formal, and if parts of the computer system design and of the code are generated by tools, then a separate software requirement document may be unnecessary for those parts. However, those parts of such computer system requirements from which code is generated or reused should be regarded as a statement of software requirements against which subsequent code should be verified. Also any separately compiled modules that are included by the code generator should be supported by separate documents for the software requirements (7.4).

On software and code hazard analysis
The application of hazard analysis to software is still barely dealt with by international guidance. The **SG** has several recommendations, e. g.:
…the computer based system and its interfaces to the plant should be evaluated at various phases of the development for potential contribution to hazards at plant level. (possible techniques are outlined in TRS 367 - Section 8.3.9). When such potential critical behaviours are identified, they should be traced into the Computer System Design, the Software Design and the code in order to identify parts of the design and of the software that require special design features. In addition, these hazards should be traced back into the requirements and should be incorporated into the plant safety analysis as appropriate(10.27).

…A documented demonstration should be provided that the Software Design addresses the hazards identified in previous analyses and the requirements that have been identified as important to safety.(8.14)

On pre-existing software (PSW)
The **SG** addresses the use of pre-existing or COTS (Commercial-Off-The-Shelf) software for safety functions (paragraphs 1.9, 2.11, 6.1, 6.40, 10.1, and annex).
The **REG** recognizes that licensees may wish to make use of such software given that appropriate assessment has been undertaken. Two of its sections deal with the issue: a specific one also reproduced as an annex in the **SG**, and another devoted to safety related systems. For safety systems, the **REG** is clear: *For safety systems (category one), the PSW shall be subjected to the same assessment (analysis and review) of the final product (not of the production process) as new software developed for the application. If necessary, reverse engineering shall be performed to enable the full specification of the PSW to be evaluated. (1.3.3.5)*

For safety related software, the **REG** recognizes that several possible sources of evidence may be exploited: : *Simplicity is required for safety systems. Safety related systems can be more complex. For these latter systems less information may be available on the development process and on the product. In certain cases, it might be possible to compensate for this lack of information - typical for pre-existing software (PSW) of category 2 - by using evidence provided by functional testing and adequate operational feedback. (1.10.1.3)*

Another source of evidence is suggested for safety related software: *In order to evaluate the possibility of relaxing certain requirements of the safety demonstration, as a minimum, the*

consequences of the potential modes of failures of the computer based system shall be evaluated. For instance, a failure mode analysis may show that certain relaxations are possible, when failures of the system can be anticipated and their effects can be detected and corrected in time by other means. (1.10.3.3)

On independent assessment

This is a difficult issue. In human societies, which draw their strength from interrelations and interdependencies, independence is somehow against nature, and often difficult to achieve. Besides, there are several types of independence, all of which are susceptible to make access to relevant information more difficult, and thereby affect assessors' competence. So, it is important to clarify what sort of independence is required and for what purpose.

The **SG** makes the following distinction**: …** *Independence includes:*

- *Technical independence: done by different people, preferably using different techniques and tools;*
- *Management independence: led and motivated by different people. The V&V team and the development team should have different management lines. Official communication between independent teams should be recorded;*
- *Financial independence: there should be a separate budget with restrictions on transfer of financial resources between development and V&V.(4.17)*

The **SG** also allows some relaxations:. *The amount and type of independent V&V should be justified with respect to the safety class of the system, e.g. financial independence may not be required for safety related systems.*

The **REG** emphasizes competence but does not go as far as strictly requiring (shall) financial independence: *The system and its safety demonstration shall be subjected to a documented review by persons who are*:

> *(a) Competent;*
>
> *(b) Organizationally independent of the supplier(s) of the system (and of its safety demonstration), and*
>
> *(c) Not responsible for or in the development, procurement and production chain of the system.(1.9.3.1)*

The **REG** also suggests that for safety related systems, independent validation only might be needed, in contrast to the requirements for independent verification (section 2.5), validation (section 2.6) and assessment (section 1.9) defined for safety systems.

On formal methods

The **REG** has 9 common positions on this difficult topic and 6 recommended practices. One of the key principles on which the common positions were founded is:

No credit can be taken in a safety demonstration for the use "per se" of a formal method without due consideration being given to the specific evidence brought in by this use, and to its contribution to the safety demonstration of the system. (1.8.3.1)

On documentation

The **SG** emphasizes documentation, and has a set of requirements on the documents to be produced in each section dealing with a stage of the development process. One general requirement is: *The set of documents should ensure the traceability of design decisions… Appropriate documents should be produced at each step of the development process. It is essential*

that documentation be updated throughout the iterative development including commissioning and ongoing maintenance processes. The documents available to the regulator should be identical to those used by the designers. The designer should be informed of this requirement early in the project.(3.35)

On determinism and interrupts

Both documents are not always fully aligned, but it is difficult to catch them out in blatant incoherence. For instance, the **SG** states: *The architecture chosen should be deterministic. A design should be selected that makes the operation of the software predictable in terms of response to inputs and the time to produce a response. A fixed, repeated sequence of operations (e.g. polling) may generally be used rather than interrupts. Communication protocols should be deterministic and should not depend on the correct operation of other, external systems (8.10)*.

The **REG** is somewhat less conservative**:** *…the code shall - as much as possible - run in a direct and fixed sequence pattern…Interrupts shall be avoided unless they lead to a significant simplification. Where interrupts are used, their usage and masking during time and data critical operations shall be proven correct and shall be well documented. The use of high-level synchronisation programming primitives shall preferably be used to deal with interrupts. The hardware and software shall be designed so that every interrupt is either serviced or explicitly masked. (2.4.3.3.2)*

On software reliability and demonstrable dependability

Here, both documents are more cautious than other international standards.

The **SG** emphasizes the issue of dependability, but avoids that of software reliability**:** *The system must not only be dependable, it must also be possible to demonstrate to the regulator that it is dependable. This safety guide is intended to guide licensees in how to achieve demonstrable dependability through design and qualification methods that improve traceability and through the production of adequate documents.(3.19)*

And later, in the section on software requirements, it explains why: *An overall software reliability target may be stated, but it should be understood that the achievement of such a target will be less demonstrable than the fulfillment of other types of requirements. It is extremely difficult to demonstrate that quantitative reliability requirements for software have been met. Currently available methods do not provide results in which confidence can be placed at the level required for systems of the highest importance to safety, and therefore this Safety Guide does not provide guidance on the use of software reliability models. If applicants propose the use of software reliability models for certification or commissioning, a rationale for the model should be included in the certification or commissioning plan and agreed with the regulatory authority. (7.11)*

The **REG** is clearly uncompromising. *It is recognised that the reliability of a computer-based safety system cannot be demonstrated by testing. Therefore, the demonstration of safety has to depend to some degree on the quality of the processes involved….(1.6.2.1).*

However, at the same time, it recommends that the level of reliability that would be required from the software be not left ignored: *The level of reliability required from the software should be explicitly stated, with the understanding that the achievement of a reliability level is less demonstrable than other requirements (2.3.4.1.4).* Retrospectively, one might wonder why this is a recommended practice, and not a common position (shall).

8. Recommendations for further work

Documents of this kind are never complete. Because they result from a consensus, they mark an important step forward, but need to be revised as knowledge and technology progress. When the **REG** neared completion, early in 2000, the members of the task force identified a few important areas where they agreed that more knowledge or experience was needed to establish useful guidance:

1. Diversity/Redundancy
   - Regulator positions requirements for diversity at architecture level;
   - Regulator positions on software diversity
2. Software Reliability
   - Methods to obtain quantitative estimations (numbers).
   - Regulator position to cope with situations where numbers cannot be obtained although quantitative objectives exist for plant operations.
3. Structure of Safety demonstration
   - Contents of a safety demonstration (safety case).
   - Organisation and structure (framework) for claims, sub-claims, arguments, proofs, ...
4. Criteria to rank software based systems in safety categories.
   - Criteria such as existence of redundant back-up, pure informative output or direct action, consequences of failure,..
5. Explicit requirements and acceptance criteria for distinct sorts of software:
   - Code produced by application oriented code generation tools (issues of validation).
   - Libraries,
   - Input/output drivers.
   - Run time and System software (operating systems), etc…

By way of independent confirmation, is was interesting to note a posteriori, that most of these topics were also included as research targets in the NRC proposed five year research plan for digital I&C technology, introduced by Steven Arndt at the Embedded Topical Meeting on Nuclear Instrumentation, Control and Human-Machine Interface Technologies, at the 2000 ANS/ENS International Meeting in Washington, D.C. [1]

9. Conclusions

The forceful value of the two documents lies in the consensus they achieve. Whatever the auspices are, consensus and common positions are always obtained at a given time, in a given context and on certain issues. They never dispense from adaptations and revisions. They are, however, the only way to make progress, especially in those cases where there is uncertainty or where some knowledge or operational experience is missing and a precautionary approach must be followed.

The work discussed above already proved useful in different respects:

- To share technical expertise among those who contributed,
- To support regulators in their national policies,
- To assist licensees in dealing with foreign manufacturers and suppliers

- To help designers produce systems that anticipate licensing requirements and are portable.

10. Acknowledgements

This paper gives a sketchy and personal viewpoint and is not an exegesis of the two guidance documents. As such, it does not do justice to the impressive work of the two teams of dedicated experts who produced them.
Thanks are due to J. Pachner, International Atomic Energy Agency, Vienna, J. Gomez, DG for Energy and Transports of the European Commission, Pierre Govaerts, AVN, and to Manfred Kersken, ISTec, Germany and Bob Yates, NII, UK, former members of those teams, for their useful comments on a previous version of this paper.
The European Commission Research Project "Cost Effective Modernisation of Systems Important to Safety (CEMSIS)" (project FIKS-CT-2000-00109) in part supported this work.

11. References

1. Licensing issues for advanced I&C technologies. Nuclear News, January 2001, 57-58.
2. Software Important to Safety in Nuclear Power Plants. IAEA Technical Reports Series 1994.. TRS N°367, 1994.
3. European nuclear regulators' current requirements and practices for the licensing of safety critical software for nuclear reactors. European Commission, DG Environment, Nuclear safety and Civil Protection, Report EUR18158 (revision 8), 1998.

Appendix 1 : Contents of Safety Guide NS-G-1.1

1. INTRODUCTION
2. TECHNICAL CONSIDERATIONS FOR COMPUTER BASED SYSTEMS
   - Characteristics of computer based systems
   - The development process
   - Safety and reliability issues
   - Organizational and legal issues
3. APPLICATION OF REQUIREMENTS FOR MANAGEMENT OF SAFETY TO COMPUTER BASED SYSTEMS
   - Requirements for management of safety
   - Design and development activities
   - Management and quality assurance
   - Documentation
4. PROJECT PLANNING
   - Development plan
   - Quality assurance programme
   - Verification and validation plan
   - Configuration management plan
   - Installation and commissioning plan
5. COMPUTER SYSTEM REQUIREMENTS
   - Recommendations
   - Documents
6. COMPUTER SYSTEM DESIGN
   - Recommendations
   - Documents
7. SOFTWARE REQUIREMENTS
   - Recommendations
   - Documents
8. SOFTWARE DESIGN
   - Recommendations
   - Documents
9. SOFTWARE IMPLEMENTATION
   - Recommendations
   - Documents
10. VERIFICATION AND ANALYSIS
    - Recommendations
    - Documents
11. COMPUTER SYSTEM INTEGRATION
    - Recommendations
    - Documents
12. VALIDATION OF THE COMPUTER SYSTEM
    - Recommendations
    - Documents
13. INSTALLATION AND COMMISSIONING
    - Recommendations
    - Documents
14. OPERATION
    - Recommendations
    - Documents
15. POST-DELIVERY MODIFICATIONS
    - Recommendations
    - Documents

ANNEX: USE AND VALIDATION OF PRE_EXISTING SOFTWARE

Appendix 2: Contents of Consensus Report EUR 19265 EN

Introduction

Background
Scope, Objectives and Implications
Safety Plan
Generic and Life Cycle Phase Licensing Issues
Recommendations

Part 1: Generic Licensing Issues

1.1 **Categorisation and Classification**
Rationale
Issues Involved
Common Position
Recommended Practices

1.2 **Applicable Standards**
Rationale
Issues Involved
Common Position
Recommended Practices

1.3 **Use and validation of Pre-existing Software**
Rationale
Issues Involved
Common Position
Recommended Practices

1.4 **Tools**
Rationale
Issues Involved
Common Position
Recommended Practices

1.5 **Organisational Requirements**
Rationale
Issues Involved
Common Position
Recommended Practices

1.6 **Software Quality Assurance Programme and Plan**
Rationale
Issues Involved
Common Position
Recommended Practices

1.7 **Security**
Rationale
Issues Involved
Common Position
Recommended Practices

1.8 **Formal methods**
Rationale
Issues Involved
Common Position
Recommended Practices

1.9 **Independent Assessment**
Rationale
Issues Involved
Common Position
Recommended Practices

1.10 **Requirements for New and Pre-existing Software (PSW) of Safety Related Systems**
Rationale
Issues Involved
Common Position
Recommended Practices

Part 2: Life Cycle Phase Licensing Issues

2.1 **Computer Based System Requirements**
Rationale
Issues Involved
Common Position
Recommended Practices

2.2 **Computer System Design**
Rationale
Issues Involved
Common Position
Recommended Practices

2.3 **Software Design and Structure**
Rationale
Issues Involved
Common Position
Recommended Practices

2.4 **Coding and Programming Directives**
Rationale
Issues Involved
Common Position
Recommended Practices

2.5 **Verification**
Rationale
Issues Involved
Common Position
Recommended Practices

2.6 **Validation**
Rationale
Issues Involved
Common Position
Recommended Practices

2.7 **Change Control and Configuration Management**
Rationale
Issues Involved
Common Position
Recommended Practices

2.8 **Operational requirements**
Rationale
Issues Involved
Common Position
Recommended Practices