CENELEC

IEC 62279: Railway applications – Communications, signaling and processing systems – Software for railway control and protection systems

(EN 50128)
March, 11th 2001.

IEC: International Electrotechnical Commission
CENELEC: Comité Européen de Normalisation Electrotechnique.

Contents

Background

- Approved by CENELEC as EN 50128 on 2000-11-01.
- Closing date for IEC voting – 2001-10-12.
- Key concept of the standard:
  - Levels of safety integrity
    - The more dangerous the consequences of a software failure, the higher the software integrity level will be.
    - Five integrity levels
      - From 0 (non safety-related) to 4 (very high)

6/22/2008 3

Scope

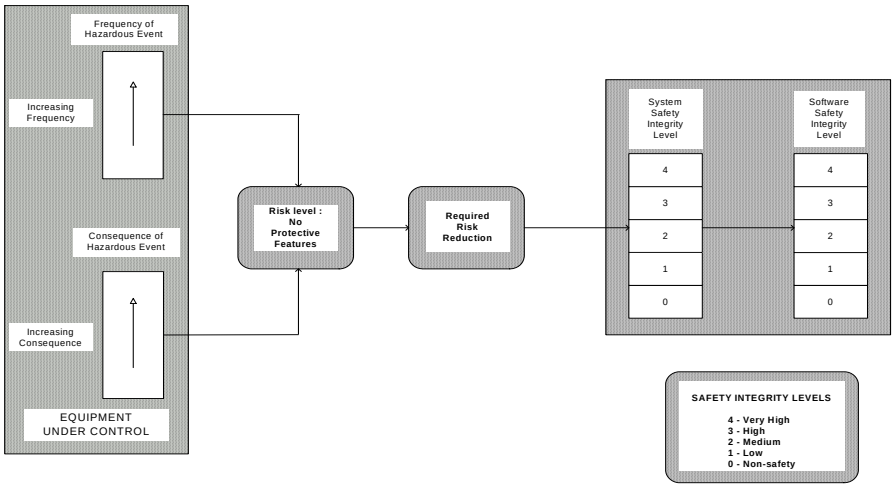
1. Specifies procedures and technical requirements for the development of programmable electronic systems for use in railway control and protection applications.
2. It is aimed at use in any area where there are safety implications.
   - These may range from the very critical, such as safety signaling to the non-critical, such as management information systems.
   - These systems may be implemented using dedicated microprocessors, programmable logic controllers, multiprocessor distributed systems, larger scale central processor systems or other architectures.
3. Applicable exclusively to software and the interaction between software and the system of which it is part.

6/22/2008 4

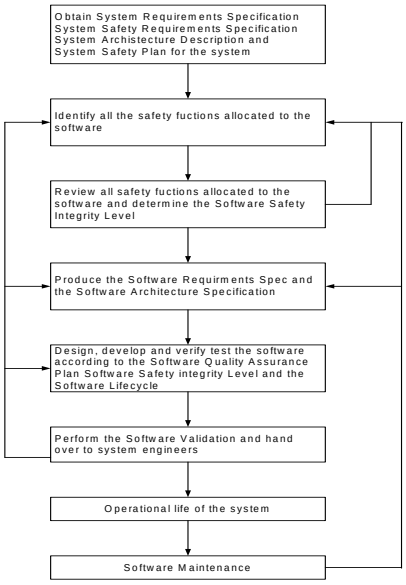
Scope

4. Software safety integrity levels above zero are for use in systems in which the consequences of failure could include loss of life.
   - Economic or environmental considerations, however, may also justify the use of higher software safety integrity levels.
5. Applies to all software used in development and implementation of railway control and protection systems including:
   - application programming;
     - e.g. Programmable Logic Controller ladder logic
   - operating systems;
   - support tools;
   - firmware.

Integrity Levels for safety-Related Systems



Software Safety Route Map



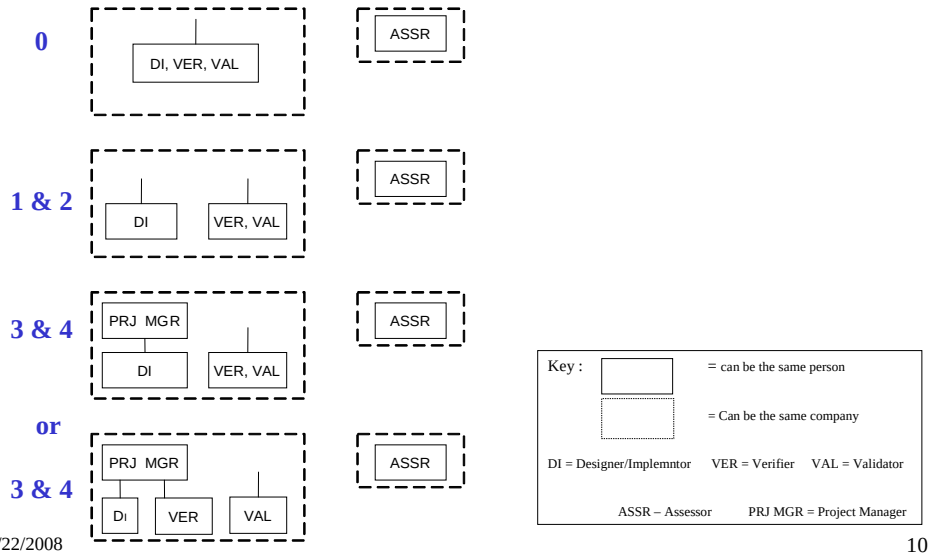
Personnel and Responsibilities

- **Objective**
  - To ensure that all personnel who have responsibilities for the software are competent to discharge those responsibilities.
- **Requirements**
  1. Implement the relevant parts of EN ISO 9001, in accordance with the guidelines contained in ISO 90003.
  2. Except at software safety integrity level zero, the safety process shall be implemented under the control of an appropriate safety organisation which is compliant with the "Safety Organisation" sub-clause in the "Evidence of Safety Management" clause of EN 50129.
  3. All personnel involved in all the phases of the Software Lifecycle, including management activities, shall have the appropriate training, experience and qualifications.

Personnel and Responsibilities

- **Requirements**
  4. An independent assessor for the software shall be appointed.
  5. The assessor shall be given authority to perform the assessment of the software.
  6. Throughout the Software Lifecycle, the parties involved shall be independent, to the extent required by the software safety integrity level, in accordance with Figure (next slide)

Independence Versus Software Integrity Level



Techniques and Measures- Requirements

- **'M': the use of a technique is mandatory;**
- **'HR': the technique or measure is Highly Recommended for this safety integrity level.**
  - If this technique or measure is not used then the rationale behind not using it should be detailed in the Software Quality Assurance Plan or in another document referenced by the Software Quality Assurance Plan;
- **'R': the technique or measure is Recommended for this safety integrity level.**
  - This is a lower level of recommendation than an 'HR' and such techniques can be combined to form part of a package;
- **'-': the technique or measure has no recommendation for or against being used;**
- **'NR': the technique or measure is positively Not Recommended for this safety integrity level.**
  - If this technique or measure is used then the rationale behind using it should be detailed in the Software Quality Assurance Plan or in another document referenced by the Software Quality Assurance Plan.

Assessment Techniques

ASSESSMENT TECHNIQUE	Ref.	SWS IL0	SWS IL1	SWS IL2	SWS IL3	SWS IL4
1. Checklists	B.	HR	HR	HR	HR	HR
2. Static Software Analysis	B. 14 B. 42 D. 8	R	HR	HR	HR	HR
3. Dynamic Software Analysis	D. 2 D. 3	-	R	R	HR	HR
4. Cause Consequence Diagrams	B. 6	R	R	R	R	R
5. Event Tree Analysis	B. 23	-	R	R	R	R
6. Fault Tree Analysis	B. 28	R	R	R	R	R
7. Software Error Effect Analysis	B. 26	-	R	R	HR	HR
8. Common Cause Failure Anaysis	B. 10	-	R	R	HR	HR
9. Markov Models	B. 41	-	R	R	R	R
10. Reliability Block Diagram	B. 51	-	R	R	R	R
11. Field Trial Before Commissioning	-	R	HR	HR	HR	HR

Requirement

1. One or more of these techniques shall be selected to satisfy the Software Safety integrity level being used.

Design and Coding Standards

TECHNIQUE MEASURE	Ref.	SWS IL0	SWS IL1	SWS IL2	SWS IL3	SWS IL4
1. Coding Standard Exists	B. 18	HR	HR	HR	HR	HR
2. Coding Style Guide	B. 16	HR	HR	HR	HR	HR
3. No Dynamic Objects	B. 16	-	R	R	HR	HR
4. No Dynamic Variables	B. 16	-	R	R	HR	HR
5. Limited Use of Pointers	B 16	-	R	R	R	R
6. Limited use of Recursion	B. 16	-	R	R	HR	HR
7. No Unconditional Jumps	B. 16	-	HR	HR	HR	HR

Requirement

1. It is accepted that techniques 3, 4 and 5 may be present as past of a validated compiler or translator.
2. A suitable set of techniques shall be chosen according to the software safety integrity level.

6/22/2008 13

Programming Languages

TECHNIQUE MEASURE	Ref.	SWS IL0	SWS IL1	SWS IL2	SWS IL3	SWS IL4
1. ADA	B. 62	R	HR	HR	R	R
2. MODULA-2	B. 62	R	HR	HR	R	R
3. PASCAL	B. 62	R	HR	HR	R	R
4. Fortran 77	B. 62	R	R	R	R	R
5. 'C' or C++ (unrestricted)	B. 62	R	-	-	NR	NR
6. Subset of C or C++ with coding	B. 62 B. 63	R	R	R	R	R
7. PLAM	B. 62	-	HR	HR	HR	HR
8. BASIC	B. 62	R	NR	NR	NR	NR
9. Assembler	B. 62	RR	R	R	-	-
10. Ladder Diagrams	B. 62	R	R	R	R	R
11. Functional Blocks	B. 62	R	R	R	R	R
12. Statement List	B. 62	R	R	R	R	R

Requirements

1. At Software Safety Integrity Level 3 and 4 when a subset of languages 1, 2, 3 and 4 are used the recommendation changes to HR.
2. For certain applications the languages 7 and 9 may be the only ones available.
   At Software Safety Integrity Level 3 an 4 where a highly recommended option is note available it is strongly recommended that to raise the recommendation to 'R' there should be a subset of these language and that there should be a precise set of coding standards.
3. For information on assessing the sutability of a programming language see entry in the bibliography for " Sutable Programming Language ", B. 62.
4. If a specific language is not in the table, it is not automatically excluded. It should, howover, conform to B. 62.

6/22/2008 14

Static Analysis

TECHNIQUE MEASURE	Ref.	SWS IL0	SWS IL1	SWS IL2	SWS IL3	SWS IL4
1. Boundary Value Analysis	B. 4	-	R	R	HR	HR
2. Checklists	B. 8	-	R	R	R	R
3. Control Flow Analysis	B. 9	-	HR	HR	HR	HR
4. Data Flow Analysis	B. 11	-	HR	HR	HR	HR
5. Error Guessing	B. 21	-	R	R	R	R
6. Fagan Inspections	B. 24	-	R	R	HR	HR
7. Sneak Circuit Analysis	B. 55	-	-	-	R	R
8. Symbolic Execution	B. 63	-	R	R	HR	HR
9. Walkhrough Design Reviews	B. 66	HR	HR	HR	HR	HR
Requirements A suitable set of techniques shall be chosen according to the software safety integrity level.						

Checklists

- **Aim**
  - To provide a stimulus to critical appraisal of all aspects of the system rather than to lay down specific requirements.
- **Description**
  1. A set of questions to be completed by the person performing the checklist.
     - Many of the questions are of a general nature and the Assessor must interpret them as seems most appropriate to the particular system being assessed.
  2. To accommodate wide variations in software and systems being validated, most checklists contain questions which are applicable to many types of system.
     - There may well be questions in the checklist being used which are not relevant to the system being dealt with and which should be ignored.
     - Equally there may be a need, for a particular system, to supplement the standard checklist with questions specifically directed at the system being dealt with.

Checklists

- **Description**
  3. The use of checklists depends critically on the expertise and judgment of the engineer selecting and applying the checklist.
     - As a result the decisions taken by the engineer, with regard to the checklist(s) selected, and any additional or superfluous questions, should be fully documented and justified.
       - The objective is to ensure that the application of the checklists can be reviewed and that the same results will be achieved unless different criteria are used.
  4. The object in completing a checklist is to be as concise as possible.
     - When extensive justification is necessary this should be done by reference to additional documentation.
     - Pass, Fail and Inconclusive, or some similar restricted set of tokens should be used to record the results for each question.
       - This conciseness greatly simplifies the process of reaching an overall conclusion as to the results of the checklist assessment.

Traceability

- **Aim**
  - To ensure that all requirements can be shown to have been properly met and that no untraceable material has been introduced.
- **Description**
  - Traceability to requirements shall be an important consideration in the validation of a system and means shall be provided to allow this to be demonstrated throughout all phases of the lifecycle.
  - Traceability shall be considered applicable to both functional and non-functional requirements and shall particularly address:
    1. Traceability of requirements to the design or other objects which fulfill them,
    2. Traceability of design objects to the implementation objects which instantiate them,
    3. Traceability of requirements and design objects to the operational and maintenance objects required to be applied in the safe and proper use of the system,
    4. Traceability of requirements, design, implementation, operation and maintenance objects, to the verification and test plans and specifications which will determine their acceptability,
    5. Traceability of verification and test plans and specifications to the test or other reports which record the results of their application.

Summary

1. **Background**
2. **Overview of the standard**
3. **Integrity Levels**
4. **Personnel and Responsibilities**
5. **Independence Versus Software Integrity Level**
6. **Checklists**
7. **Traceability**