A study of the impact of applying EN50716 to EN50128 compliant Software engineering

Mohan Murari, Martin Hughes and Dr. Andrew Hussey
Hitachi Rail STS

mohan.murari@hitachirail.com, martin.hughes@hitachirail.com,
andrew.hussey@hitachirail.com

Abstract

The EN50128 standard is widely applied and accepted in the rail industry for the development of software for programmable electronic systems for use in safety related railway control and protection (signalling) applications.

Previous studies have confirmed the effectiveness of this standard in improving the overall software quality [Hussey 2023].

In parallel, the EN50657 standard has been in use for the development of software for programmable electronic systems for rolling stock applications.

More recently, the EN50716 standard has been developed, with the aim to replace both EN50128 and EN50657 with a common standard for railway software applications.

To date, there are few studies of the differences between the new EN50716 standard and the preceding EN50128 and EN50657 standards.

In this paper, we address this topic, from the perspective of EN50128:2011, and analyse the impact on 50128:2011 compliant process of changes introduced by the transition to the new EN50716 standard. We classify the changes according to a classification scheme, including whether the change has major (semantic) impact or not and group the changes into categories (new, modified and deleted) and derive keywords for each change. We consider significant impact arising from the major changes, and how they will affect typical railway software development enterprises. Minor changes are not explicitly dealt with in this paper.

Keywords: Software Systems, Railway, Signalling, Rolling Stock, EN50128, EN50716

1 Introduction

The CENELEC standard EN50128 has been commonly applied to railway signalling Software Engineering as part of the overall CENELEC lifecycle, in conjunction with EN50126 and EN50129. The objective of the EN50128 standard is to provide a framework for the Software process, based on the SIL of the Safety Functions implemented by that Software. Based on the application of the EN50128 process, a corresponding Functional Failure Rate may be claimed for the Software in respect of its Safety Functions. Implicitly, the application of EN50128 should lead to more robust Software with fewer faults.

However, in parallel with the EN50128, for rolling stock SW engineering, the standard EN50657 has been applied. The existence of the two standards EN50128 and EN50657 has led to the anomalous situation whereby different processes are applied according to whether the onboard equipment on a train is considered part of the signalling or part of the rolling stock application. To address that concern, the new standard EN50716 has been developed.
In this paper, we consider the impact of applying EN50716 for a SW system that has previously been developed under EN50128 and which is subject to major change. We consider the impact of EN50716 from two perspectives:

- Whether the change had only a syntactic impact or whether the change was semantic;
- Where the changes were semantic, which was the topic being addressed.

We derive a list of eight (8) key changes that have major impact for anyone who would update their SW process from EN50128 to EN50716.

The key contributions and additions to current learning discussed in this paper are as follows:

1. a comparison of EN50128 to EN50716;
2. The categorisation of those changes;
3. The impact assessment for those changes.

2 Acronyms, Abbreviations, and Definitions

2.1 Acronyms and Abbreviations

Acronym	Description
AI	Artificial Intelligence
CENELEC	European Committee for Electrotechnical Standardization
ML	Machine Learning
PM	Project Manager
RAMS	Reliability Availability Maintainability & Safety
SIL	Safety Integrity Level
STS	Hitachi Rail STS
SW	Software
SysML	Systems Modelling Language
UML	Unified Modelling Language

2.2 Definitions

Requirement

(1) A condition or capability needed by a user to solve a problem or achieve an objective.

(2) A condition or capability that must be met or possessed by a product, service, or product component to

satisfy a supplier agreement, standard, specification, or other formally imposed documents.

(3) A documented representation of a condition or capability as in (1) or (2).

Refer to the IEEE Standard Glossary of Software Engineering Terminology [IEEE90].

System

The abstraction level related to the Scope of Work under analysis. The System is composed of a set of Subsystems and Interfaces.

3 Literature Survey

Because the EN50716 standard is very new, there are still currently few studies comparing EN50716 with EN50128. High level overviews of the major differences from EN50128 are given in [Prover 2024a] and in [CSA 2024]. However, the differences highlighted in these two overviews themselves differ, possibly due to the different perspectives of the authors. From [Prover 2024a], [Prover 2024b] there is raised:

- Formal Methods
- Tool diversity
- Annex C
- Software management and organization

Whereas [CSA 2024] focuses on:

- Basic Integrity
- Removal of the Integrator role
- Development of application data or algorithms: systems configured by application data

A brief overview by [Gencer 2023] also notes the topics of alignment with EN50126 and EN50129 and AI integration.

Our survey of the literature confirms the need for a comprehensive assessment of the differences between EN50128 and EN50716, along with their impact for SW development that is EN50128:2011 compliant.

4 Method

To analyse the differences between the two standards, we followed a simple three step approach:

- A difference check was made to determine which changes had occurred and the differences were summarized.
- Each difference was then categorized according to the following framework:
    - Whether it was a new clause, modified clause or deleted clause
    - The impact of the changes, as Minor and Major, where Minor changes tend to be of a syntactic nature, with little or no deep impact on the semantics of the software development process
    - Keywords were derived from the summary of the differences
    - A category was determined for the change, based on the keywords, these categories are listed in section 5.
- An analysis was made, recorded in this paper, for the Major Impact changes.

5 Outcomes

5.1 New clauses

Categories for new clauses,

1. Software Development Organisation: Where different parts of a specific software development are being undertaken by separate and distinct organizations there shall be a clear definition of the tasks being undertaken by each organization (refer to clause 6.5.4.18).
2. Suitable programming language: A suitable programming language shall be selected regarding the criteria in Table A.15.
3. Application Integration Test Report: An Application Integration Test Report shall be written, under the responsibility of the Tester, based on the Application Integration Test Specification (refer to clauses 8.4.5.6 & 8.4.5.7).
4. Application Release Note: An Application Release Note which accompanies the delivered application data shall be written, under the responsibility of the Designer (refer to clauses 8.4.7.7 to 8.4.7.9).
5. Lifecycle Models: Annex C.1 provides additional guidance on creating lifecycle models, especially with respect to possible phase sequences. EN50716 gives example of a linear “waterfall” model and the “V” model. For the purposes of annex C, the only difference between “Waterfall” and “V” models is that the “V” model shows dependencies of a given phase to all relevant preceding phases more explicitly, while the “Waterfall” model explicitly visualizes the dependency on the preceding phase only. Iterative lifecycle models are also discussed, but no specific development framework is provided.
6. Iterative lifecycle models: In iterative lifecycle models, activities from one or more of the phases are executed repeatedly to progressively cover the project scope (refer to C.1.3 in Annex C).
7. Software Modelling: Annex C.2 provides additional guidance on software modelling. Annex C.2 explains possible usage benefits of modelling approaches during development at software level as well as specific aspects to consider. It also provides guidance for a possible application of modelling in software development in compliance with the requirements of the standard.
   For example, semi-formal modelling notation includes Unified Modelling Language (UML), Systems Modelling Language (SysML)
   Modelling is specifically applicable to requirements and design as per Tables A.2 and A.4 and shall comply with the requirements of Table A.17. For example, formal methods of modelling become highly recommended for SIL1 & SIL 2 as well. However, the testing techniques in Tables A.5, A.6 and A.7 also apply to models as does the techniques for coding standards in Table A.12.

Example adaptations for modelling of Table A.4, A.5 and A.12 are given in Tables C.1, C.2 and C.3.

8. Artificial Intelligence and Machine Learning: Annex C.3 provides guidance on using Artificial Intelligence in Software solutions. Per EN50716, the term Artificial Intelligence (AI) covers a wide range of disciplines, research fields, applications, and techniques. Machine Learning (ML) is considered the most relevant of these areas for software and which is therefore within the scope of the EN50716 standard.

   EN50716 considers there to be four challenges of machine learning which are hard in the sense that it is not yet possible to be confident that they can be solved either by existing techniques or by techniques currently under development.

   1. Ensuring that the training data are sufficiently complete and accurate
   2. Verifying the trained software
   3. Validation of approximated functionality
   4. Adversarial attacks and missing causality

   For adversarial attacks, e.g. small deviations in the picture to be recognized by a ML algorithm where even small changes in the picture can completely change the categorization of the object. This is among others due to the fact that a ML algorithm does not provide causal relations between input and output, but merely statistical correlations.

   Further, these challenges must be overcome if AI/ML techniques are to be included as viable techniques for software development within EN 50716.

5.2 Modified clauses

Categories for modified clauses,

1. Application Integration Test: Additional new documents need to be produced for application data development which are, Application Integration Test Specification, Application Integration Test Report, & Application Release Note. This requires additional effort and requirement for application developer. Refer to clause 8.3 of EN 50716.
2. Basic Integrity: For Basic Integrity software, the software interface specification is only required for the boundary of the overall software. Refer to clause 7.3.4.18 of EN 50716.
3. Coding Standards and Style Guide: Additional requirements for improving the availability and reliability of the Coding Standards and Style Guide. Refer to clause D.15 of EN 50716.
4. Independence of roles: EN 50716 emphasis the independence of roles inside the SW development organization. Interestingly Assessors may be part of supplier, customer, or third-party organisations as per new standard. Refer to clause 5.1.1 & 5.1.2.5 of EN 50716.
5. Independence of roles, role definitions: Duplicate text in role definitions have been removed, segregated the roles of verifier & validator as different person, Assessor is removed for Basic integrity & the validator shall not report to PM for SIL3 and above. Refer to clause 5.1.2.10, 5.1.2.11 & 5.1.2.12 of EN 50716.
6. SW Assessment: For Basic Integrity, requirements of the EN 50716 standard shall be fulfilled but no assessment will be required. Refer to clause 6.4.1.2 & 8.4.6 of EN 50716.
7. Minor upgrade, minor maintenance: Application of EN 50716 standard document during upgrades and maintenance of existing software is advisable (Not highly recommended). Refer to clause 1.9 & 9.2.4.1 of EN 50716.
8. Non-safety: For non-safety related functions a software quality assurance process shall be used. Refer to clause 4.4 of EN 50716.
9. Release notes: Output documents like “Software Release and Deployment Plan” & “Release Notes” are no more required for Software deployment and maintenance as per clause 9.1.3 of EN 50716.
10. Role definitions: Role definitions have been updated for implementer & validator, additionally the integrator role is removed from the standard. Refer to Annex B of EN 50716.
11. Site testing: The consequences shall be analysed in order to evaluate what kind of tests have to be defined and performed in the real environment and which are still suitable to be performed on the test bench. Refer to clause 7.7.4.5 of EN 50716.
12. Software Quality Assurance Verification Report: Software Quality Assurance Verification Report document is removed from the new standard as per clause 6.5.3, 6.5.4.7 & 6.5.4.8 of EN 50716.
13. Software Validation Verification Report: Software Validation Verification Report document is removed from the new standard as per clause 6.3.3 of EN 50716.
14. Translator: The evaluation of a Translator may be performed for a specific application project, or for a class of applications. This is not required as per the new standard and related clause have been deleted. Refer to clauses 6.7.4.7 & 6.7.4.8 of EN 50716.

5.3 Deleted clauses

Categories for deleted clauses,

1. Non-Safety: Clause 1.3 of EN 50128 regarding the applicability of the standard only to systems with a safety impact has been removed. EN 50716 therefore applies Basic Integrity even to systems with no safety impact. Refer to Section 1 of EN 50716.
2. Role definitions: The role of Integrator has been removed and combined with Tester; Assessor is removed for Basic integrity. Refer to Annex B.4 of EN 50716.
3. Software Validation Verification Report: Software Validation Verification Report document is removed from the new standard as per clause 6.3.3 of EN 50716.
4. SW Assessment: Software Assessment Verification Report is removed from the new standard as per Table A.1. of EN 50716.

5. Generic Software: Clause 8.4.8 of EN 50128 regarding the development of generic software has been removed. Refer to clauses 7.3.4.13 & 9.1.4.9 of EN 50716. The same process applies for all Software. EN50716 assumes that continued development of Generic SW constitutes a SW change or maintenance activity.
6. Techniques: Many outdated techniques that appear in Annex D of EN 50128 have been removed from EN 50716 and replaced with generic clauses regarding technique selection e.g. Section D.9 for Common Cause Failure Analysis has been deleted. This particularly applies to programming languages and modelling diagrams as per clause 7.3.4.25 of EN 50716.

6 Future ideas

This paper has only considered the comparison of EN50128:2011 with EN50716:2023. Comparison of EN50657 to EN50716 is not in scope. Future work may be conducted to make this comparison to assist the transition of Rolling Stock RAMS projects to use the new EN50716 standard.

7 Conclusions

This paper has considered the differences between EN50128 and EN50716 and the impact of those changes for a system that is EN50128 compliant.

The key contributions and additions to current learning discussed in this paper were a detailed comparison of EN50128 and EN50716, which enabled us to derive:

1. The categories of difference between EN50128 and EN50716.
2. The impact of the differences, and in particular which changes can be considered significant.

Top impacts identified by our analysis include:

- The process for developing non-safety SW is now covered by Basic Integrity.
- There is more flexibility to define custom lifecycle models. Simpler to align a bespoke company process to the EN50716 process, e.g. agile process/methodologies.
- Obligations with respect to Application Data have increased e.g. Application Integration Test process and documentation including Application Integration Test Specification, Application Integration Test Report, & Application Release Note.
- The treatment of AI in Annex C.3 raises 4 key challenges but does not give specific methods or processes to follow to overcome those challenges.
- Verification reports have been simplified/rationalised i.e. the Software Quality Assurance Verification Report and Software Validation Verification Report are no longer required.
- Role descriptions have been simplified, specifically the Integrator role is removed and combined with Tester, Assessor is removed for Basic integrity.
- Generic SW is no longer specially treated, the same process now applies for all Software.
- Techniques have been updated and simplified e.g. Tables A3, A4, A5, A12, A13 and A15 are significantly changed along with many other changes in Appendix A and D and the addition of a new Annex C.2 for modelling.

Overall, the EN50716 standard makes various improvements and updates to the EN50128 standard, while retaining the basic safety assurance approach and methodology. In general we consider the EN50716 standard to be clearer and simpler to apply than EN50128 and recommend its adoption by projects even in advance of the last date (30 October 2026) by which EN50128 may be used.

8 References

Prover (2024a): *Exploring the Evolution of Railway Software Standards with EN 50716*. Available at: https://www.prover.com/quality/exploring-the-evolution-of-railway-software-standards-with-en-50716/

Prover (2024b): *A Guide on CENELEC EN 50716*. Available at: https://www.prover.com/guide/cenelec-en-50716/

C. B. Gencer (2023): *EN 50716: A New Era in Railway System Software Standards*. Available at: https://www.linkedin.com/pulse/en-50716-new-era-railway-system-software-standards-can-berk-gencer-p1d0c/

Andrew Hussey, Maria Hill, Martin Hughes and Lionel van den Berg (2023): *An empirical study of the practical application of EN50128 to SW engineering*. ASSC 2023.

CSA Engineering (2024): *New standard EN 50716:2023 "Requirements for software development" Part 1*, Available at: https://www.csa.ch/en/blog/new-standard-en-507162023-requirements-for-software-development-part-1