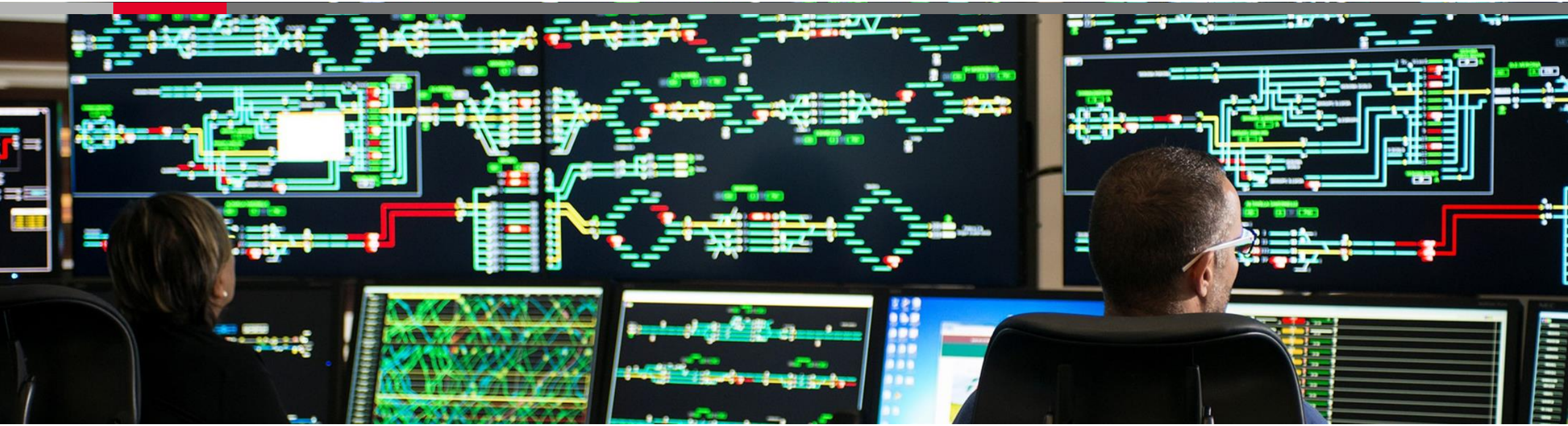
Applying EN50716 to EN50128 compliant Software

Mohan Murari, Martin Hughes & Andrew Hussey

October 2024



Agenda


HITACHI
Inspire the Next
DEFINING THE FUTURE OF RAILWAY SOFTWARE DEVELOPMENT

Introduction

EN50128 standard is widely applied and accepted

EN50716 standard replaces EN50128 and EN50657 for Rolling Stock

In effect since November 2023

Provide a framework for the Software process, based on the SIL of the Safety Functions

Implemented at national level by 2024-10-30

Conflicting national standards withdrawn by 2026-10-30

Motivation

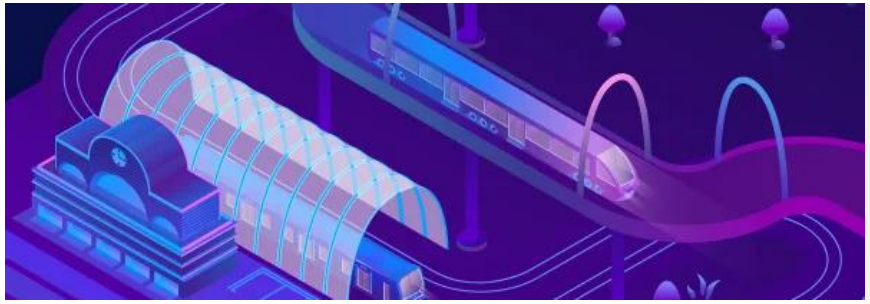
Why are we interested in studying this topic?

Different V&V regimes for Signalling vs Rolling Stock

– 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 are considered part of the signalling or part of the rolling stock application

Necessary for EN50126 safety process

– The V&V process in EN50716 is aligned with EN50126 to assure the safety for the potential systematic faults in software



EN50128 is widely used

– Major changes to existing EN50128 compliant software will need to follow EN50716 in the near future

Method

Difference check

– A difference check was made to determine which changes had occurred and the differences were summarized.

Impact Assessment

Each difference was then assessed according to the following framework:

– Whether it was a new clause, deleted clause or changed 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

Categorisation

Each impact was then categorised according to the following framework:

– Keywords were derived from the summary of the differences
– A category was determined for the change, based on the keywords

Analysis

An analysis was made, recorded in this paper, for the Major Impact changes, summarizing the impacts.

Top Impacts - New

What has been added that was not in EN 50128?

- The treatment of AI / Machine Learning in Annex C.3 raises 4 key challenges
  - Ensuring that the training data are sufficiently complete and accurate
  - Verifying the trained software
  - Validation of approximated functionality
  - Adversarial attacks and missing causality
- 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.
  - Iterative lifecycle models now explicitly included
- Obligations with respect to Application Data have increased – new documents:
  - Application Integration Test Report
  - Application Release Note
- Software Modelling guidance provided in Annex C.2

Top Impacts - New

HITACHI
Inspire the Next

Table C.1 — Architecture and design typical adaptation for modelling

TECHNIQUES / MEASURES OF TABLE A.4	TYPICAL ADAPTATION FOR MODELLING
1. Modelling	applicable as is
2. Structured methodology	applicable as is
3. Modular approach	A module is an element of organization of the source code to improve its understanding and separate the concerns. If the source code is not automatically generated, if it's modified after generation or if it's the input of manual analysis, this technique remains applicable.
4. Component	applicable as is
5. Design and coding standards	The design standard shall cover the modelling notation (i.e. it shall encourage good modelling practices and avoid poorly-defined features of the modelling language) and the structure, organization and hierarchy of the model. The coding standard is fully applicable when the source code is not automatically generated, when it's modified after generation or when it's input of manual analysis. Otherwise, the coding standard may be present as part of a validated translator, particularly the rules contributing to the avoidance of poorly-defined features of the programming language.
6. Analysable programs	If analysis are performed on the source code, the technique is applicable as-is. If all or parts of the analysis are made on the models, the models also need to be analysable.
7. Structured programming	This technique is used to limit the structural complexity of the source code. An equivalent technique need then to be similarly applied on models. The technique is also applicable to the source code if specific analysis are done on it.
8. Suitable Programming language	Criteria for programming languages are also applicable to modelling notations. If the source code is automatically generated but neither modified after generation nor the input of subsequent analysis and depending on the guarantees provided by the generator, some of the criteria can be useless.

Table C.2 — Component implementation and testing typical adaptation for modelling

SUBCLAUSES	TYPICAL ADAPTATION FOR MODELLING
7.5.4.2 The size and complexity of the developed source code shall be balanced.	7.5.4.2 The size and complexity of the developed model shall be balanced.
7.5.4.3 The Software Source Code shall be readable, understandable and testable.	7.5.4.3 The model shall be readable, understandable and testable.
7.5.4.4 The Software Source Code shall be placed under configuration control before the commencement of documented testing.	7.5.4.4 The model shall be placed under configuration control before the commencement of documented testing.
7.5.4.10 After the Software Source Code and the Software Component Test Report have been established, verification shall address a) the adequacy of the Software Source Code as an implementation of the Software Component Design Specification, b) the correct use of the chosen techniques and measures from Table A.4 as a set satisfying 4.8 and 4.9, c) determining the correct application of the coding standards, d) that the Software Source Code meets the general requirements for readability and traceability in 5.3.2.7 to 5.3.2.10 and in 6.5.4.14 to 6.5.4.17, as well as the specific requirements in 7.5.4.1 to 7.5.4.4, e) the adequacy of the Software Component Test Report as a record of the tests carried out in accordance with the Software Component Test Specification	7.5.4.10 After the model and the Software Component Test Report have been established, verification shall address a) the adequacy of the model as an implementation of the Software Component Design Specification, b) the correct use of the chosen techniques and measures from Table A.4 as a set satisfying 4.8 and 4.9, c) determining the correct application of the modelling standards, d) that the model meets the general requirements for readability and traceability in 5.3.2.7 to 5.3.2.10 and in 6.5.4.14 to 6.5.4.17, as well as the specific requirements in 7.5.4.1 to 7.5.4.4, e) the adequacy of the Software Component Test Report as a record of the tests carried out in accordance with the Software Component Test Specification

Table C.3 — Coding standards techniques / measures typical adaptation for modelling

TECHNIQUES / MEASURES OF TABLE A.12	TYPICAL ADAPTATION FOR MODELLING
1. Coding Standard	Modelling Standard
2. Coding Style Guide	Modelling Style Guide
3. Limited size and complexity of Functions, Subroutines and Methods	Limited size and complexity of model parts
4. Entry/Exit Point strategy for Functions, Subroutines and Methods	Not applicable since models generally don't have the concept of unconditional jump
5. Defined control of Global Variables	Defined control of input and output data in model parts (e.g. signals, information flows)

Top Impacts - New

HITACHI
Inspire the Next

Table A.17 — Modelling

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Data Modelling	D.65	R	HR	HR	HR	HR
2. Data Flow Diagrams	D.11	-	HR	HR	HR	HR
3. Control Flow Diagrams	D.66	R	HR	HR	HR	HR
4. Finite State Machines or State Transition Diagrams	D.27	-	HR	HR	HR	HR
5. Petri Nets	D.55	-	HR	HR	HR	HR
6. Decision/Truth Tables	D.13	R	HR	HR	HR	HR
7. Formal Methods	D.28	-	HR	HR	HR	HR
8. Performance Modelling	D.39	-	HR	HR	HR	HR
9. Prototyping/Animation	D.43	-	R	R	R	R
10. Structure Diagrams	D.51	-	HR	HR	HR	HR
11. Sequence Diagrams	D.67	R	HR	HR	HR	HR
12. Cause Consequence Diagrams	D.6	R	R	R	R	R
13. Event Tree Diagrams	D.22	-	R	R	R	R

Requirements:
1) For SIL 1-4, modelling guidance shall be defined and used.
2) For SIL 1-4, at least one of the HR techniques shall be chosen.

Top Impacts - Modified

Key topics modified with respect to EN 50128?

- Obligations with respect to Application Data have increased – additional test documents
- Role descriptions have been simplified,
  - Specifically, the Integrator role is removed and combined with Tester,
  - Independent Assessor is removed for Basic integrity
- Techniques have been updated and simplified
  - Tables A3, A4, A5, A12, A13 and A15 are significantly changed

Top Impacts - Modified

EN50128

BS EN 50128:2011+A2:2020
EN 50128:2011+A2:2020 - 70 -

Table A.3 – Software Architecture (7.3)

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Defensive Programming	D.14	-	HR	HR	HR	HR
2. Fault Detection & Diagnosis	D.26	-	R	R	HR	HR
3. Error Correcting Codes	D.19	-	-	-	-	-
4. Error Detecting Codes	D.19	-	R	R	HR	HR
5. Failure Assertion Programming	D.24	-	R	R	HR	HR
6. Safety Bag Techniques	D.47	-	R	R	R	R
7. Diverse Programming	D.16	-	R	R	HR	HR
8. Recovery Block	D.44	-	R	R	R	R
9. Backward Recovery	D.5	-	NR	NR	NR	NR
10. Forward Recovery	D.30	-	NR	NR	NR	NR
11. Retry Fault Recovery Mechanisms	D.46	-	R	R	R	R
12. Memorising Executed Cases	D.36	-	R	R	HR	HR
13. Artificial Intelligence – Fault Correction	D.1	-	NR	NR	NR	NR
14. Dynamic Reconfiguration of software	D.17	-	NR	NR	NR	NR
15. Software Error Effect Analysis	D.25	-	R	R	HR	HR
16. Graceful Degradation	D.31	-	R	R	HR	HR
17. Information Hiding	D.33	-	-	-	-	-
18. Information Encapsulation	D.33	R	HR	HR	HR	HR
19. Fully Defined Interface	D.38	HR	HR	HR	M	M
20. Formal Methods	D.28	-	R	R	HR	HR
21. Modelling	Table A.17	R	R	R	HR	HR
22. Structured Methodology	D.52	R	HR	HR	HR	HR
23. Modelling supported by computer aided design Table and specification tools	A.17	R	R	R	HR	HR

Requirements:

1) Approved combinations of techniques for Software Safety Integrity Levels 3 and 4 are as follows:
   a) 1, 7, 19, 22 and one from 4, 5, 12 or 21;
   b) 1, 4, 19, 22 and one from 2, 5, 12, 15 or 21.
2) Approved combinations of techniques for Software Safety Integrity Levels 1 and 2 are as follows: 1, 19, 22 and one from 2, 4, 5, 7, 12, 15 or 21.
3) Some of these issues may be defined at the system level.
4) Error detecting codes may be used in accordance with the requirements of EN 50159.

NOTE Technique/measure 19 is for External Interfaces.

EN50716

Table A.3 — Software architecture (7.3)

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Defensive Programming	D.14	-	HR	HR	HR	HR
2. Fault Detection and Diagnosis	D.26	-	R	R	HR	HR
3. Error Correcting Codes	D.19	-	-	-	-	-
4. Error Detecting Codes	D.19	-	R	R	HR	HR
5. Failure Assertion Programming	D.24	-	R	R	HR	HR
6. Safety Bag Techniques	D.47	-	R	R	R	R
7. Diverse Programming	D.16	-	R	R	HR	HR
8. Recovery Block	D.44	-	R	R	R	R
9. Backward Recovery	D.5	-	NR	NR	NR	NR
10. Forward Recovery	D.30	-	NR	NR	NR	NR
11. Retry Fault Recovery Mechanisms	D.46	-	R	R	R	R
12. Memorising Executed Cases	D.36	-	R	R	HR	HR
13. Artificial Intelligence and Machine Learning	C.3	-	NR	NR	NR	NR
14. Dynamic Reconfiguration	D.17	-	NR	NR	NR	NR
15. Software Error Effect Analysis	D.25	-	R	R	HR	HR
16. Graceful Degradation	D.31	-	R	R	HR	HR
17. Fully Defined Interface	D.38	HR	HR	HR	M	M

66

EN 50716:2023 (E)

18. Modelling	Table A.17	R	R	R	HR	HR
19. Structured Methodology	D.52	R	HR	HR	HR	HR

Requirements:

1) Approved combinations of techniques for software integrity levels 3 and 4 are as follows:
a) 1, 7, 17, 19 and one from 4, 5, 12 or 18;
b) 1, 4, 17, 19 and one from 2, 5, 12, 15 or 18.
2) Approved combinations of techniques for software integrity levels 1 and 2 are as follows: 1, 17, 19 and one from 2, 4, 5, 7, 12, 15 or 18.
3) Some of these issues may be defined at the system level.
4) Error detecting codes may be used in accordance with the requirements of EN 50159.
5) For Basic Integrity, fully defined interfaces are only highly recommended at the boundary of the overall software.

NOTE 1 Technique/measure 17 is for external interfaces.
NOTE 2 The techniques/measures are general and independent from processing technology (instruction set, parallelism/pipelining, cache levels, number of cores, …)

Top Impacts - Modified

EN50128

Table A.4– Software Design and Implementation (A2 7.3, 7.4, and 7.5 A2)

TECHNIQUE/MEASURE	Ref	A2 Basic Integrity A2	SIL 1	SIL 2	SIL 3	SIL 4
1. Formal Methods	D.28	-	R	R	HR	HR
2. Modelling	Table A.17	R	HR	HR	HR	HR
3. Structured methodology	D.52	R	HR	HR	HR	HR
4. Modular Approach	D.38	HR	M	M	M	M
5. Components	Table A.20	HR	HR	HR	HR	HR
6. Design and Coding Standards	Table A.12	HR	HR	HR	M	M
7. Analysable Programs	D.2	HR	HR	HR	HR	HR
8. Strongly Typed Programming Language	D.49	R	HR	HR	HR	HR
9. Structured Programming	D.53	R	HR	HR	HR	HR
10. Programming Language	Table A.15	R	HR	HR	HR	HR
11. Language Subset	D.35	-	-	-	HR	HR
12. Object Oriented Programming	Table A.22 D.57	R	R	R	R	R
13. Procedural programming	D.60	R	HR	HR	HR	HR
14. Metaprogramming	D.59	R	R	R	R	R

Requirements:

1) An approved combination of techniques for Software Safety Integrity Levels 3 and 4 is 4, 5, 6, 8 and one from 1 or 2.
2) An approved combination of techniques for Software Safety Integrity Levels 1 and 2 is 3, 4, 5, 6 and one from 8, 9 or 10.
3) Metaprogramming shall be restricted to the production of the code of the software source before compilation.

EN50716

Table A.4 — Software design and implementation (7.3, 7.4 and 7.5)

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Modelling	Table A.17	R	HR	HR	HR	HR
2. Structured methodology	D.52	R	HR	HR	HR	HR
3. Modular Approach	D.38	HR	M	M	M	M
4. Components	Table A.20	HR	HR	HR	HR	HR
5. Design and Coding Standards	Table A.12	HR	HR	HR	M	M
6. Analysable Programs	D.2	HR	HR	HR	HR	HR
7. Structured Programming	D.53	R	HR	HR	HR	HR
8. Suitable Programming Languages	Table A.15	R	HR	HR	HR	HR

Requirements:

1) An approved combination of techniques for software integrity levels 3 and 4 is 1, 3, 4, 5.
2) An approved combination of techniques for software integrity levels 1 and 2 is 2, 3, 4, 5 and one from 7 or 8.

Top Impacts - Modified

EN50128

BS EN 50128:2011+A2:2020

EN 50128:2011+A2:2020 - 72 -

Table A.5 – Verification and Testing (A2 6.2 and 7.4 A2)

TECHNIQUE/MEASURE	Ref	A2 Basic Integrity A2	SIL 1	SIL 2	SIL 3	SIL 4
1. Formal Proof	D.29	-	R	R	HR	HR
2. Static Analysis	Table A.19	-	HR	HR	HR	HR
3. Dynamic Analysis and Testing	Table A.13	-	HR	HR	HR	HR
4. Metrics	D.37	-	R	R	R	R
5. Traceability	D.58	R	HR	HR	M	M
6. Software Error Effect Analysis	D.25	-	R	R	HR	HR
7. Test Coverage for code	Table A.21	R	HR	HR	HR	HR
8. Functional/ Black-box Testing	Table A.14	HR	HR	HR	M	M
9. Performance Testing	Table A.18	-	HR	HR	HR	HR
10. Interface Testing	D.34	HR	HR	HR	HR	HR

Requirements:

1) For software Safety Integrity Levels 3 and 4, the approved combination of techniques is 3, 5, 7, 8 and one from 1, 2 or 6.

2) For Software Safety Integrity Level 1 and 2, the approved combinations of techniques is 5 together with one from 2, 3 or 8.

NOTE 1 Techniques/measures 1, 2, 4, 5, 6 and 7 are for verification activities.

NOTE 2 Techniques/measures 3, 8, 9 and 10 are for testing activities.

EN50716

EN 50716:2023 (E)

Table A.5 — Software component analysis and testing (6.2 and 7.4)

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Formal Proof	D.29	-	R	R	HR	HR
2. Static Analysis	Table A.19	-	HR	HR	HR	HR
3. Dynamic Analysis and Testing	Table A.13	HR	HR	HR	M	M
4. Metrics	D.37	-	R	R	R	R
5. Test Coverage for code	Table A.21	-	HR	HR	HR	HR
6. Performance Testing	Table A.18	-	HR	HR	HR	HR
7. Interface Testing	D.34	HR	HR	HR	HR	HR

Requirements:

1) For software integrity levels 3 and 4, the approved combinations of techniques are:

1, 3 and 5; or:

2, 3 and 5.

2) For software integrity levels 1 and 2, the approved combinations of techniques are:

2 and 5; or:

3 and 5.

3) For Basic Integrity, interface testing is only highly recommended for the boundary of the overall software.

Top Impacts - Modified

HITACHI
Inspire the Next

EN50128

Table A.12 – Coding Standards

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Coding Standard	D.15	HR	HR	HR	M	M
2. Coding Style Guide	D.15	HR	HR	HR	HR	HR
3. No Dynamic Objects	D.15	-	R	R	HR	HR
4. No Dynamic Variables	D.15	-	R	R	HR	HR
5. Limited Use of Pointers	D.15	-	R	R	R	R
6. Limited Use of Recursion	D.15	-	R	R	HR	HR
7. No Unconditional Jumps	D.15	-	HR	HR	HR	HR
8. Limited size and complexity of Functions, Subroutines and Methods	D.38	HR	HR	HR	HR	HR
9. Entry/Exit Point strategy for Functions, Subroutines and Methods	D.38	R	HR	HR	HR	HR
Text deleted						
10. Limited use of Global Variables	D.38	HR	HR	HR	M	M
Requirement: 1) It is accepted that techniques 3, 4 and 5 may be present as part of a validated compiler or translator.						

EN50716

Table A.12 — Coding standards

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Coding Standard	D.15	HR	HR	HR	M	M
2. Coding Style Guide	D.15	HR	HR	HR	HR	HR
3. Limited size and complexity of Functions, Subroutines and Methods	D.38	HR	HR	HR	HR	HR
4. Entry/Exit Point strategy for Functions, Subroutines and Methods	D.38	R	HR	HR	HR	HR
5. Defined control of Global Variables	D.38	HR	HR	HR	M	M
Requirements: 1) It is accepted that technique 3 may not be possible for complex components for which further modularization logically is not sensible as complexity depends on the problem.						

Top Impacts - Modified

HITACHI
Inspire the Next

EN50128

Table A.13 – Dynamic Analysis and Testing

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Test Case Execution from Boundary Value Analysis	D.4	-	HR	HR	HR	HR
2. Test Case Execution from Error Guessing	D.20	R	R	R	R	R
3. Test Case Execution from Error Seeding	D.21	-	R	R	R	R
4. Performance Modelling	D.39	-	R	R	HR	HR
5. Equivalence Classes and Input Partition Testing	D.18	R	R	R	HR	HR
6. Structure-Based Testing	D.50	-	R	R	HR	HR
Requirement: 1) The analysis for the test cases is at the sub-system level and is based on the specification and/or the specification and the code.						

EN50716

Table A.13 — Dynamic analysis and testing

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Test Case Execution from Error Guessing	D.20	R	R	R	R	R
2. Test Case Execution from Error Seeding	D.21	-	R	R	R	R
3. Structure-Based Testing	D.50	-	R	R	HR	HR
4. Test Case Execution from Cause Consequence Diagrams	D.6	-	-	-	R	R
5. Prototyping / Animation	D.43	-	-	-	R	R
6. Test Case Execution from Boundary Value Analysis	D.4	-	HR	HR	HR	HR
7. Equivalence Classes and Input Partition Testing	D.18	R	HR	HR	HR	HR
8. Process Simulation	D.42	R	R	R	R	R
Requirement: 1) The analysis for the test cases is at the relevant level and is based on the specification and/or the specification and the code. 2) The completeness of the simulation will depend upon the extent of the software integrity level, complexity and application.						
NOTE 1 Techniques/measures 1 and 2 are experience-based testing techniques/measures. NOTE 2 Techniques/measures 4, 6 and 7 are specification-based testing techniques/measures (Functional / Black-box testing). NOTE 3 Technique/Measure 3 is typically used at component level to complement Techniques/Measures 6 and 7 in order to achieve the required test coverage.						

Top Impacts - Modified

EN50128

BS EN 50128:2011+A2:2020
EN 50128:2011+A2:2020 - 76 -

Table A.15 – Textual Programming Languages

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. ADA	D.54	R	HR	HR	HR	HR
2. MODULA-2	D.54	R	HR	HR	HR	HR
3. PASCAL	D.54	R	HR	HR	HR	HR
4. C or C++	D.54 D.35	R	R	R	R	R
5. PL/M	D.54	R	R	R	NR	NR
6. BASIC	D.54	R	NR	NR	NR	NR
7. Assembler	D.54	R	R	R	R	R
8. C#	D.54 D.35	R	R	R	R	R
9. JAVA	D.54 D.35	R	R	R	R	R
10. Statement List	D.54	R	R	R	R	R

Requirements:

1) The selection of the languages shall be based on the requirements given in 6.7 and 7.3.

2) There is no requirement to justify decisions taken to exclude specific programming languages.

NOTE 1 For information on assessing the suitability of a programming language see entry in D.54 'Suitable Programming Languages'.

NOTE 2 If a specific language is not in the table, it is not automatically excluded. It should, however, conform to D.54.

NOTE 3 Run-time systems associated with selected languages which are necessary to run application programs should still be justified for usage according to the Software Safety Integrity Level.

EN50716

Table A.15 — Suitable Programming Languages

TECHNIQUE/MEASURE	Ref	Basic Integrity	SIL 1	SIL 2	SIL 3	SIL 4
1. Documented and defined operational semantics	D.54	R	HR	HR	M	M
2. Supports modular approaches	D.54, D.38	R	HR	HR	HR	HR
3. Supports commenting	D.54	R	HR	HR	HR	HR
4. Strong type system and checking	D.54	R	HR	HR	HR	HR
5. Supports static analysis	D.54, Table A.19	R	HR	HR	HR	HR
6. Supports Testing	D.54	R	HR	HR	HR	HR
7. Supports Test Coverage Metrics	D.54, Table A.21	R	HR	HR	HR	HR

Requirements:

1) The selection of the languages shall be based on the requirements given in 7.3.

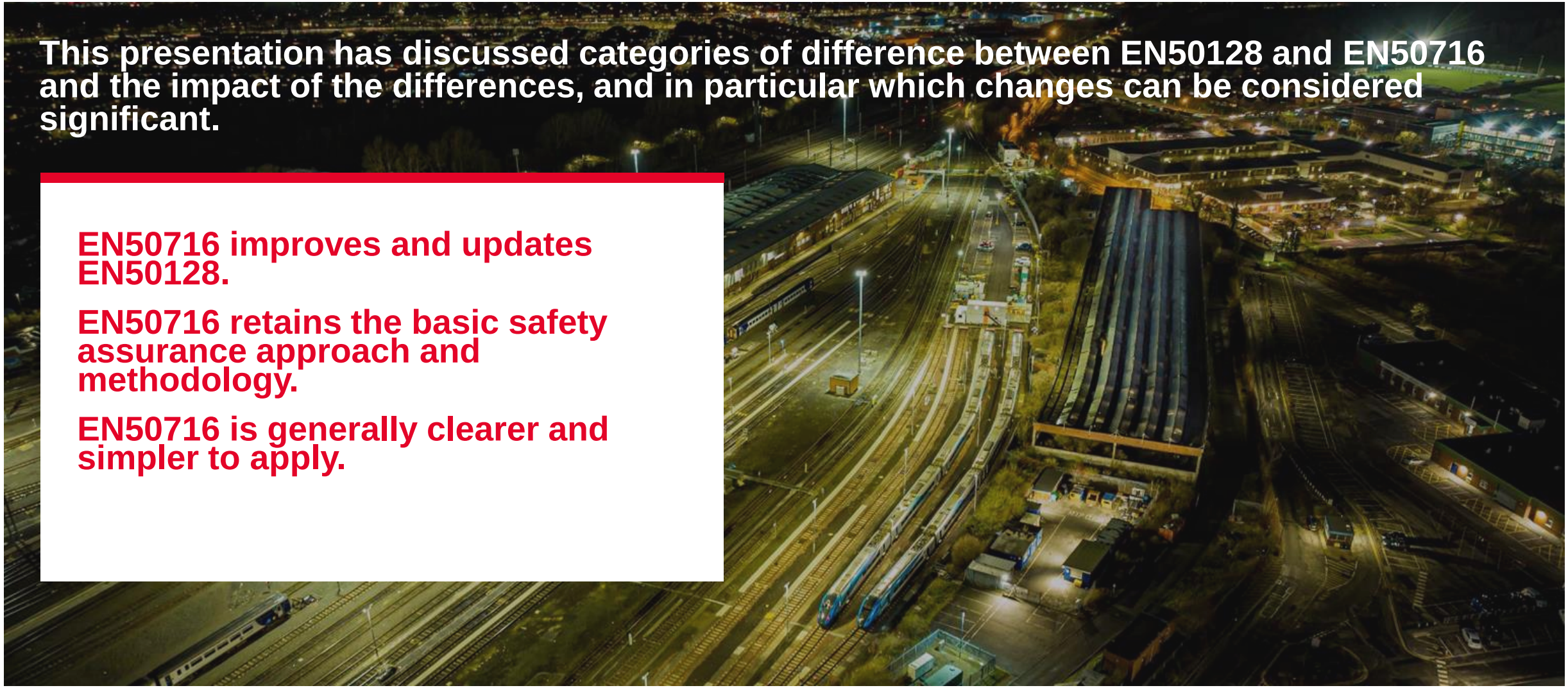
NOTE For information on assessing the suitability of a programming language see entry in D.54 'Suitable Programming Languages'

Top Impacts - Deleted

What was deleted from EN 50128?

- Basic Integrity or another suitable software quality assurance process must now be applied for non-safety-related software.
- The list of verification reports has been simplified
  - Software Validation Verification Report is removed
  - Software Assessment Verification Report is removed
- Generic software is now no longer a special case – the same process applies for all software
- Many outdated techniques from Annex D have now been removed and replaced with generic clauses in EN 50716

Conclusions



This presentation has discussed categories of difference between EN50128 and EN50716 and the impact of the differences, and in particular which changes can be considered significant.

EN50716 improves and updates EN50128.

EN50716 retains the basic safety assurance approach and methodology.

EN50716 is generally clearer and simpler to apply.

Future

What does the future hold?

Cutoff Dates

- EN50716 must be adopted by projects after 30th October 2026.
- EN50128 cannot be used after this date

- Projects starting before 30th October 2026 can use either standard.
- Projects changing existing software previously assured under EN50128 are permitted to keep using the old standard.

Adoption Into Companies

- Adapting company process to a new standard takes time
- EN50716 remains a complex standard to meet from scratch

- It is recommended that projects start adopting EN50716 early.
- Getting used to the updated requirements before October 2026 will make transition smoother.

Rolling Stock

- EN50716 is a combination of EN50128 and the equivalent rolling stock software standard EN50657

- Does this update standard affect rolling stock software process differently?
- While initial checks seem to show that EN50657 is similarly affected to EN50128, further comparison work is required in this area.

Thank you

Mohan Murari
Senior RAMS Engineer

Martin Hughes
Senior RAMS Engineer

Andrew Hussey
Director of RAMS - APAC

93 Francisco St,
Belmont
Perth 6104
Western Australia

www.hitachirail.com

HITACHI
Inspire the Next

Q&A