Validation and Verification of METEOR safety software

Jean-Louis BOULANGER

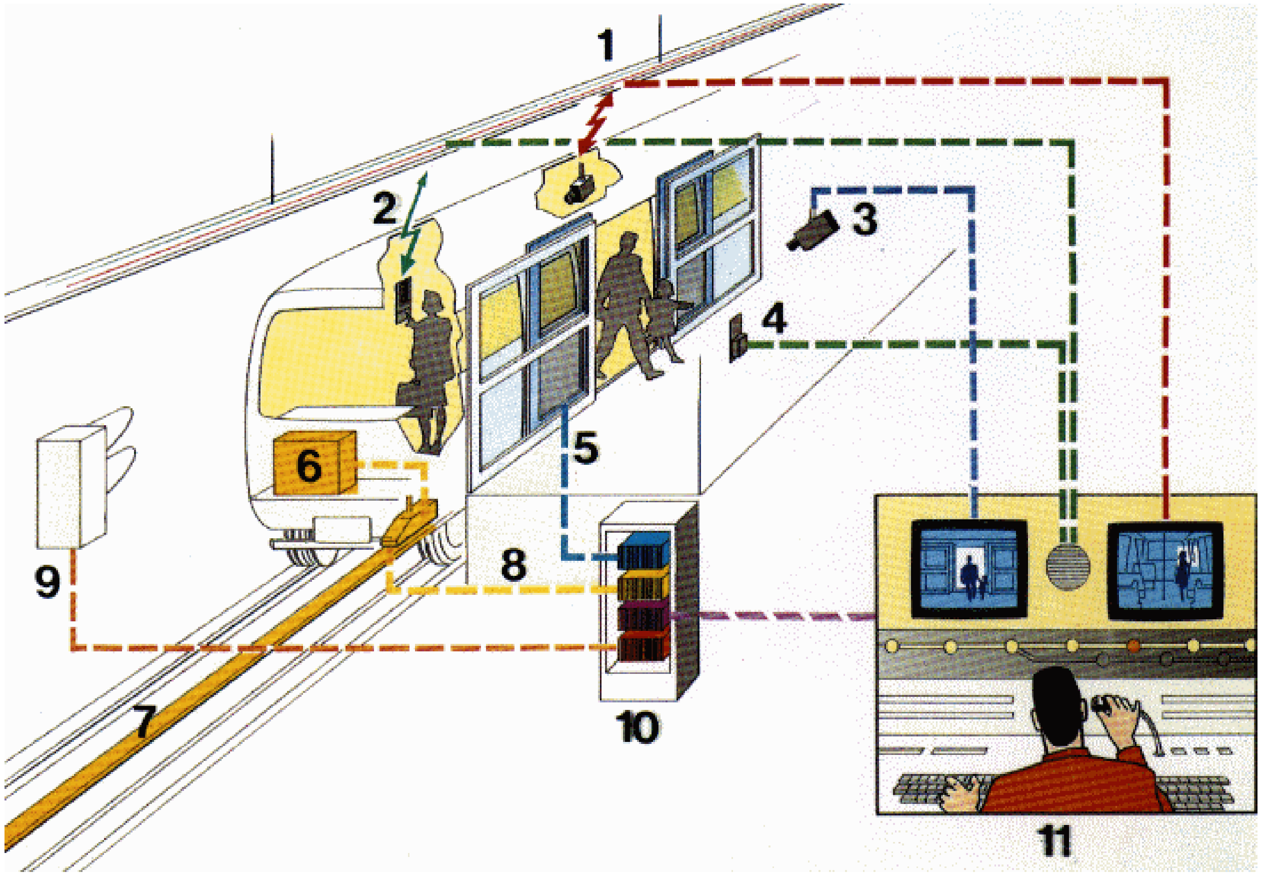
CERTIFER/SILAS

METEOR presentation

METEOR is also the line 14 of the Paris Métro

- Inaugurated the 15/10/1998
- 7.2 km, between “ Madeleine “ and “François Mitterrand Library“ for 7 stations
- 25 000 passengers by hour and by wayside
- 19 trains of 6 cars (extension to 8 cars possible)
- 40 km/h : Commercial speed
- 85 s : Interval in automatic mode

A complex automatism



1 - video in train

2 - inter-phone in train

3 - video in platform

4 - inter- phone in platform

5 - platform doors

6 -On Board control unit

7 - transmission

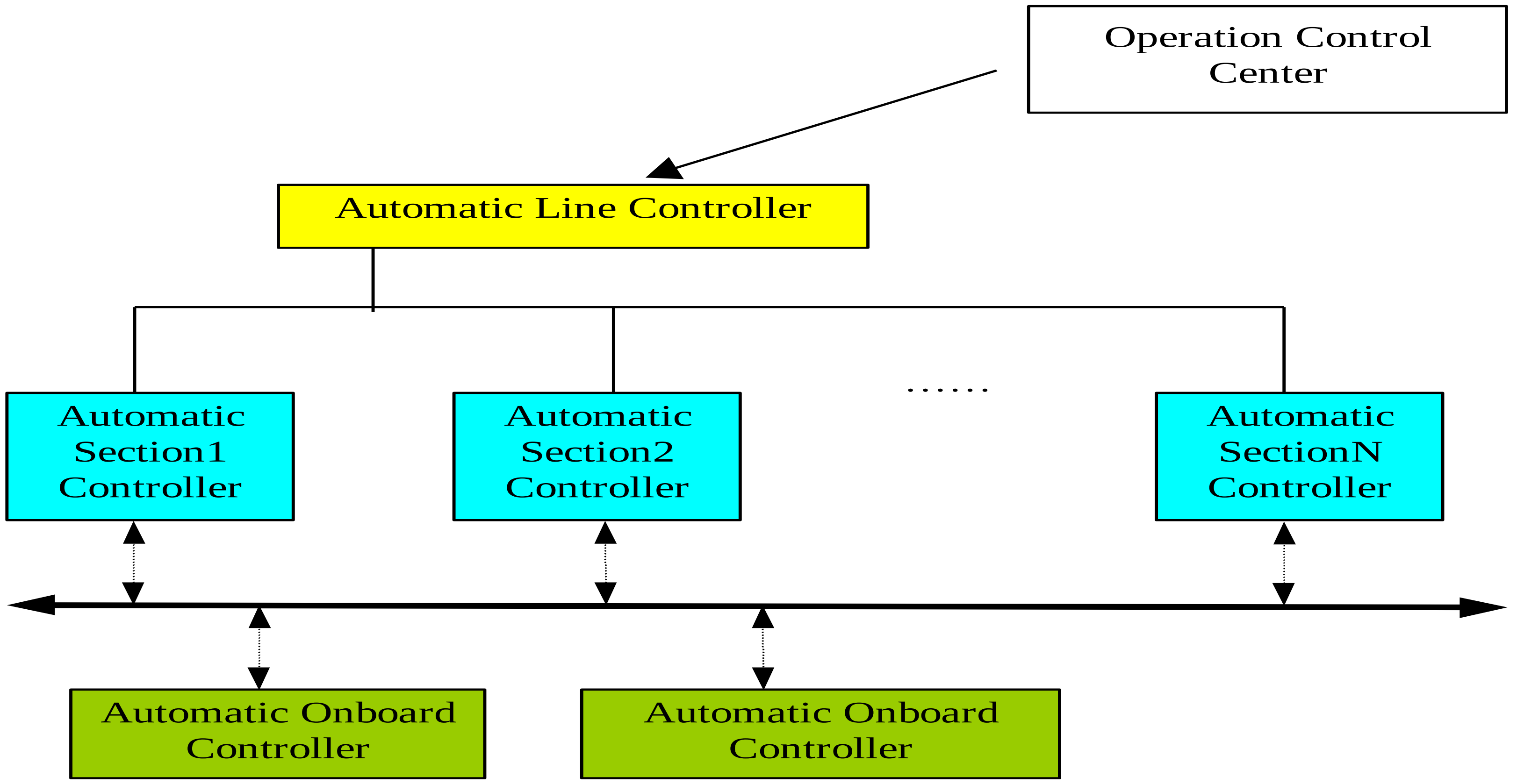
8 - transmission SWE-OBCU

9 - interlocking

10 -Sector Wayside equipment

11 -Operation Control center

A distributed architecture of redundancy calculators


Operation Control Center
Automatic Line Controller
Automatic Section1 Controller
Automatic Section2 Controller
......
Automatic SectionN Controller
Automatic Onboard Controller
Automatic Onboard Controller

METEOR, is ...

- Two kinds of trains :

  - equipped train

  - Unequipped train

- Two modes of running :

  - Automatic mode

  - Manual mode

Development process

Safety Calculator Architecture

- Standard VME with Safety Coded Processor technology
- A generic elementary calculator :

  - Specific treatment of safety coded processor,

  - Input acquisition,

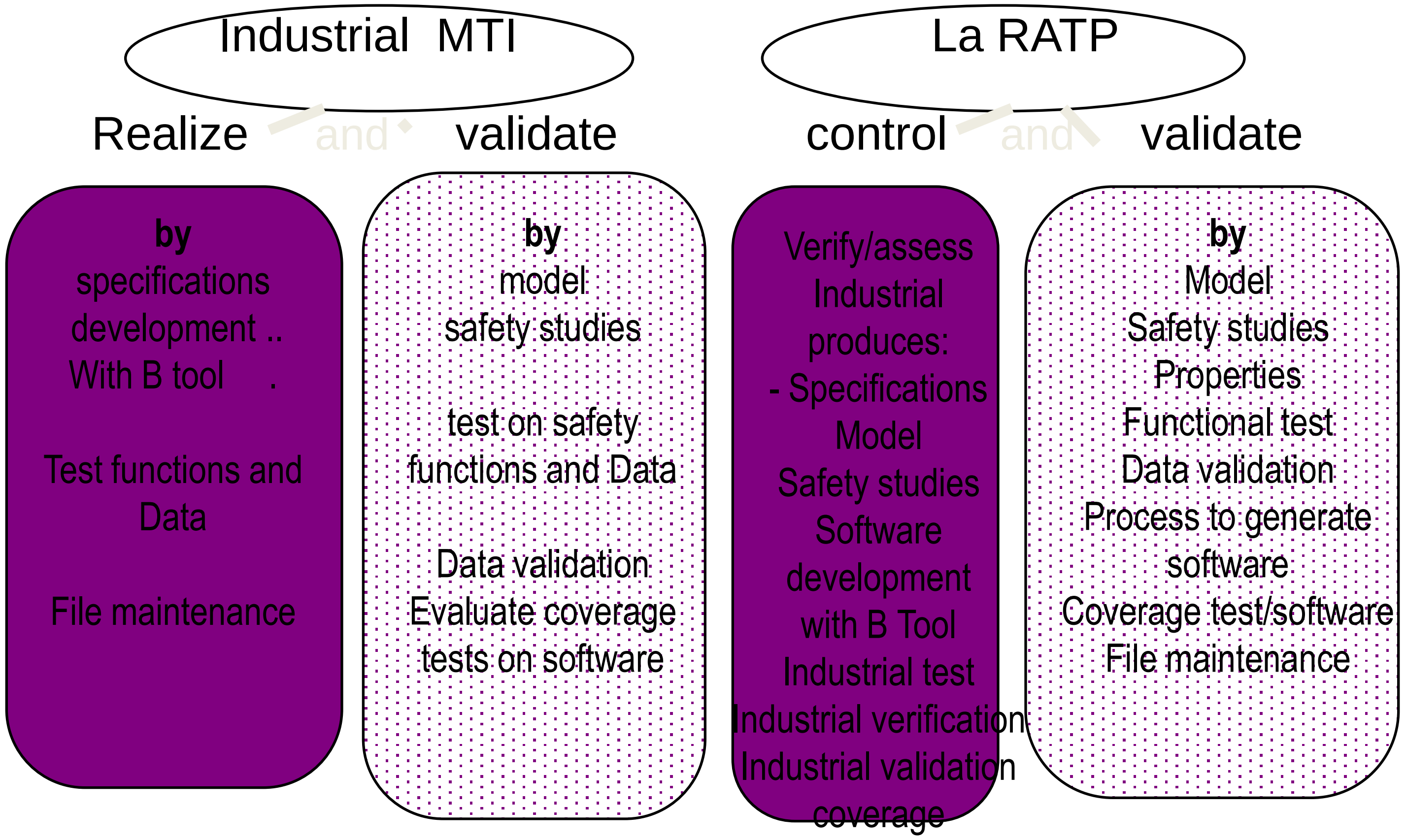
  - Management of the transmission, …
- 3 specific applications :

  - Safety application,

  - Functional application,

  - Transmission application

Actors and functions



Industrial MTI

Realize and validate

by
specifications
development ..
With B tool .

Test functions and
Data

File maintenance

by
model
safety studies

test on safety
functions and Data

Data validation
Evaluate coverage
tests on software

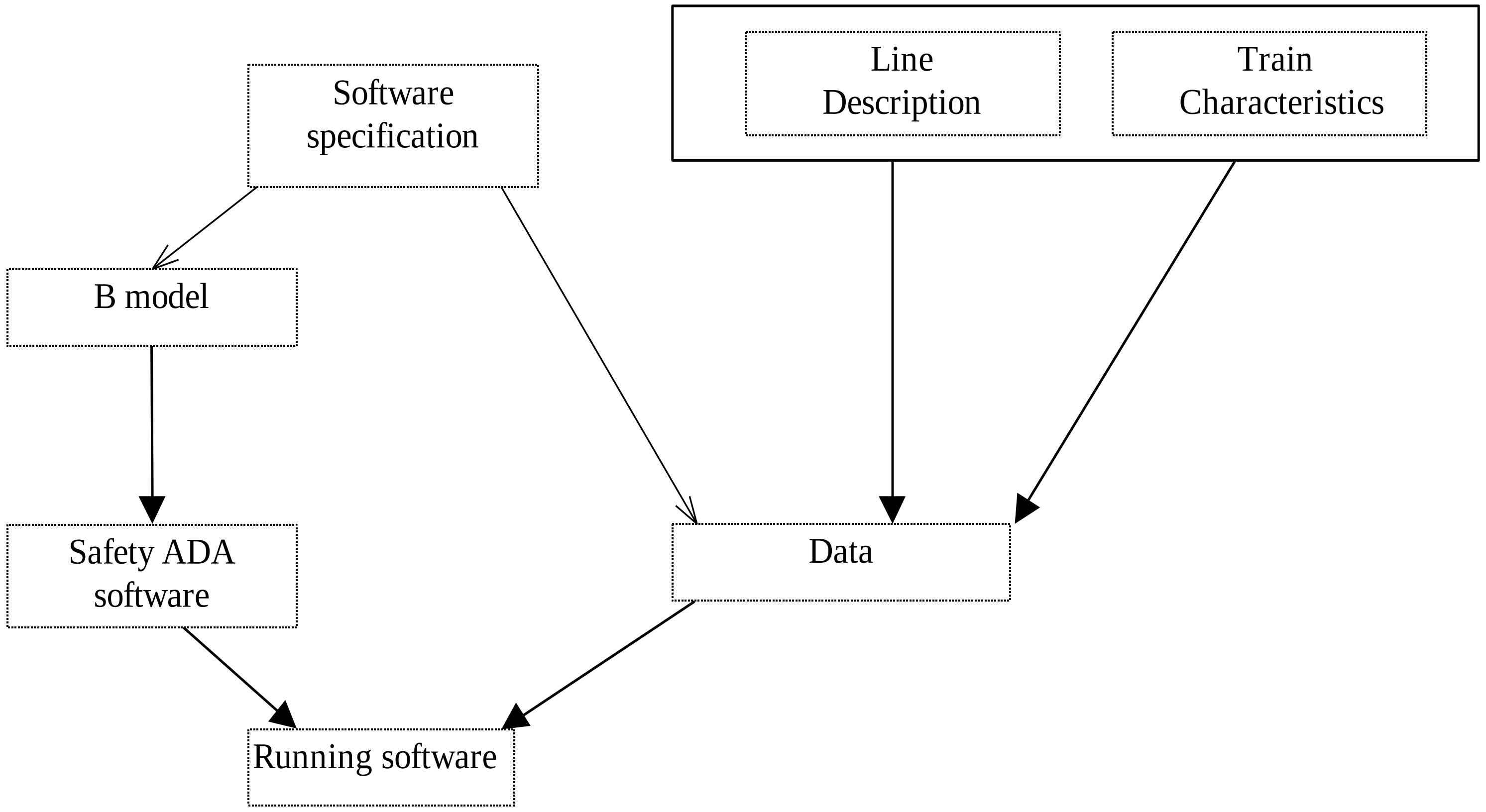
La RATP

control and validate

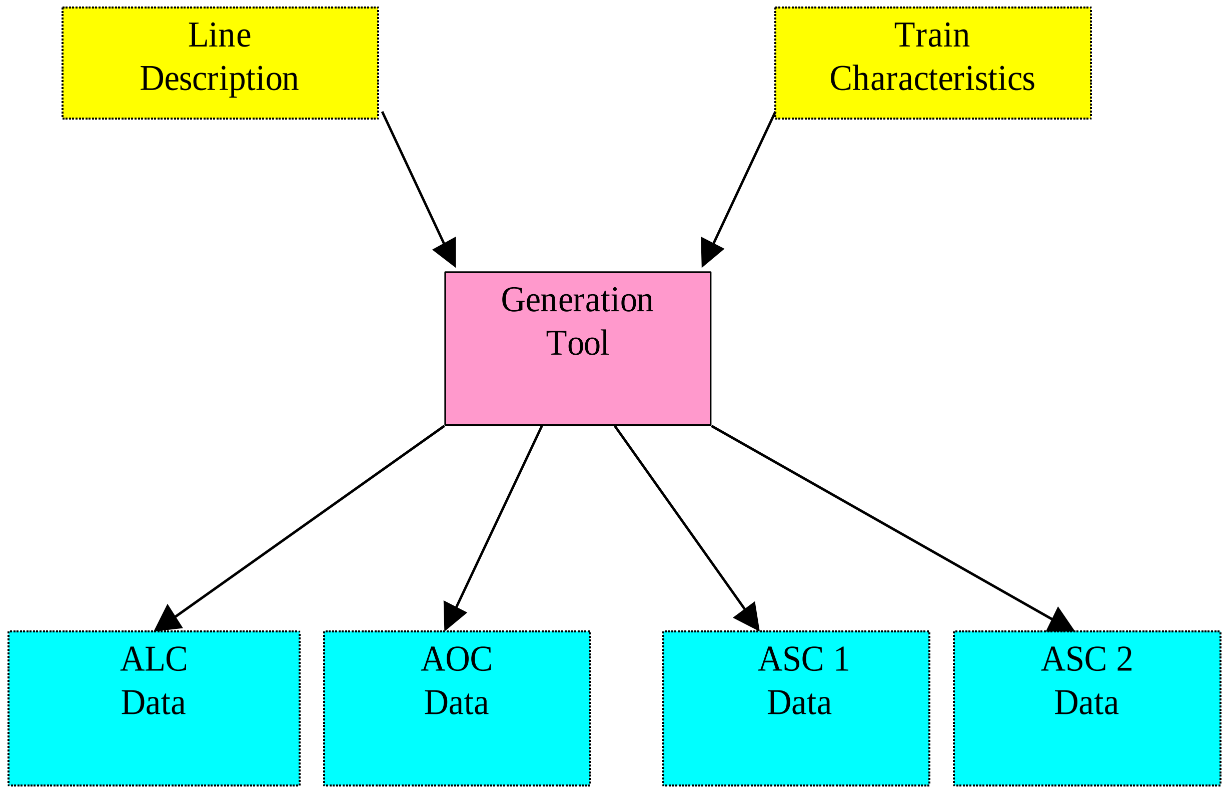
Verify/assess
Industrial
produces:
- Specifications
Model
Safety studies
Software
development
with B Tool
Industrial test
Industrial verification
Industrial validation
coverage

by
Model
Safety studies
Properties
Functional test
Data validation
Process to generate
software
Coverage test/software
File maintenance

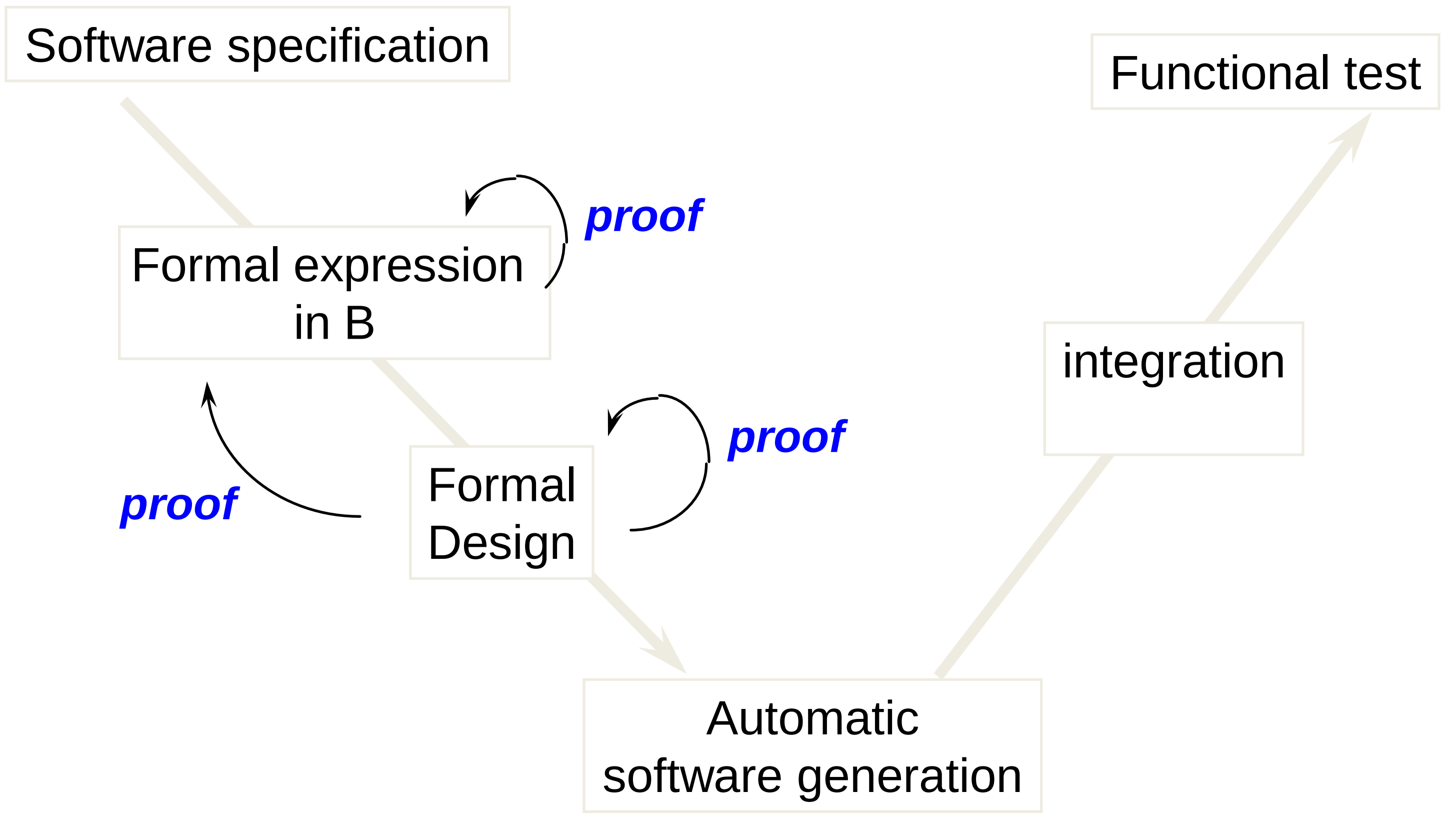
Software and Data



Process to develop data



Life cycle with B



RATP Validation process

RATP Process

3 Main activities :

- Validation of the elementary calculator
- Functional validation of the safety software
- Verification of industrial produces

Functional validation (1)

- Formalisation of activities and responsibilities in a Software Validation Plan
- Formalisation of the methodology to determine tests in a Tests Plan
- Formalisation of the methodology to validate the data in a Data Validation Plan

Functional Validation (2)

- Analysis of specification documents to :
  - get knowledge of the system,
  - produce a critical analysis,
  - produce a list of safety functions,
  - produce a list of safety properties,
- Produce a Principle functional book for each safety function

Functional Validation (3)

- Models of safety critical functions :

  Dynamic analysis used to

  - Validate specification

  - Verify safety properties respect

  - determine functional tests

- Tools used :

  - ELSIR

  - ASA+

Functional Validation (4)

- Test of safety critical functions :
  - Produce tests with models
  - Run tests on test benches
  - Verify test results
  - Verify properties on test results
  - determine test coverage/specification

Functional Validation (5)

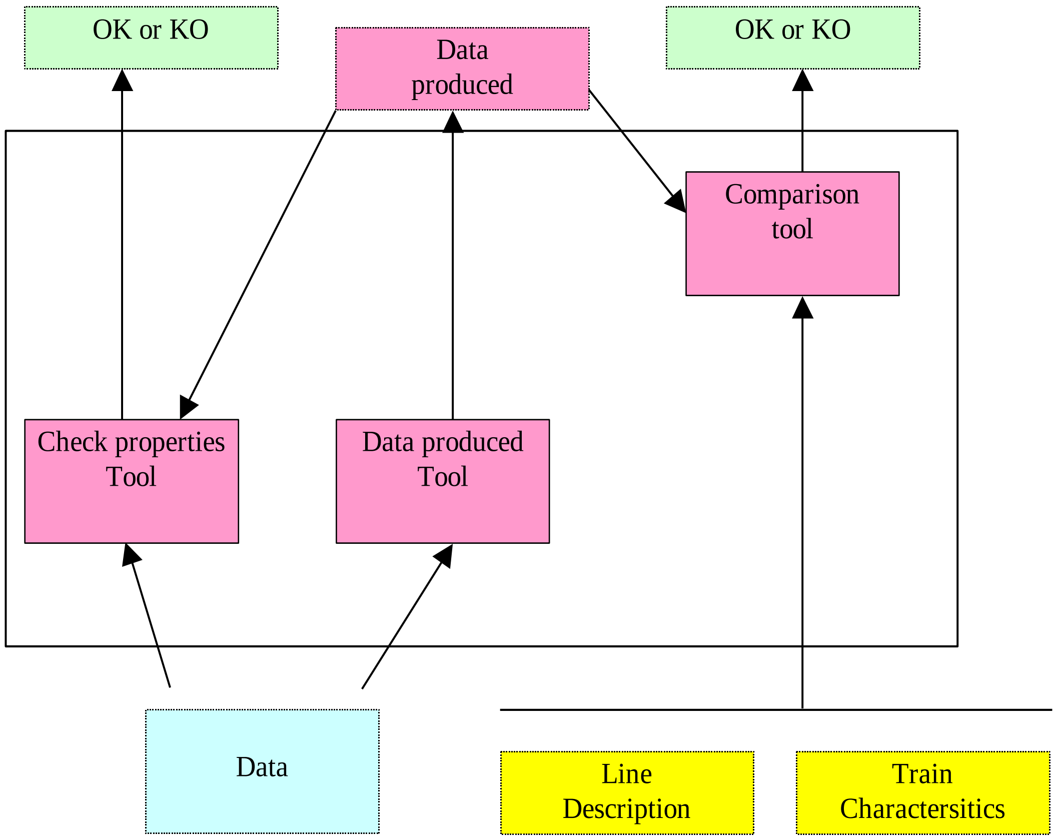
- Additional analysis for Distributed Functions :

  - Safety analysis to determine critical situation level sub-system,

  - Dynamic analysis to verify the timing of information exchange between calculators,

  - Determination of specific tests.

Data validation tool



Data properties
Example

- P4 :All track circuits are well chained in the line.
- P5 : Route is correct with switch position.
- P6 : Speed is correct with protected point.
- …

Conclusion

Some results

RATP process :

- 20 principle books on safety function
- 23 Models
- 30 tests books
- 5 000 tests on real time simulator.

Remarks / Anomalies

- 400 Safety remarks on software specifications

- 110 Anomalies on safety software

  Of course, all safety-critical anomalies were corrected before the latest version of the software was released.

METEOR results (1)

- **Since 1998, METEOR is running without problem**
- **Service Quality = 99.8**
- **Passengers by day = 130 000**
- **Satisfaction of the passengers**
- **Successful results**

METEOR results (2)

- COFRAC had accredited our laboratory on this process on 1999
- We were the first French Laboratory to be accredited by the COFRAC
- In 2000, accreditation was extended with B method and our accreditation was continued for 15 months

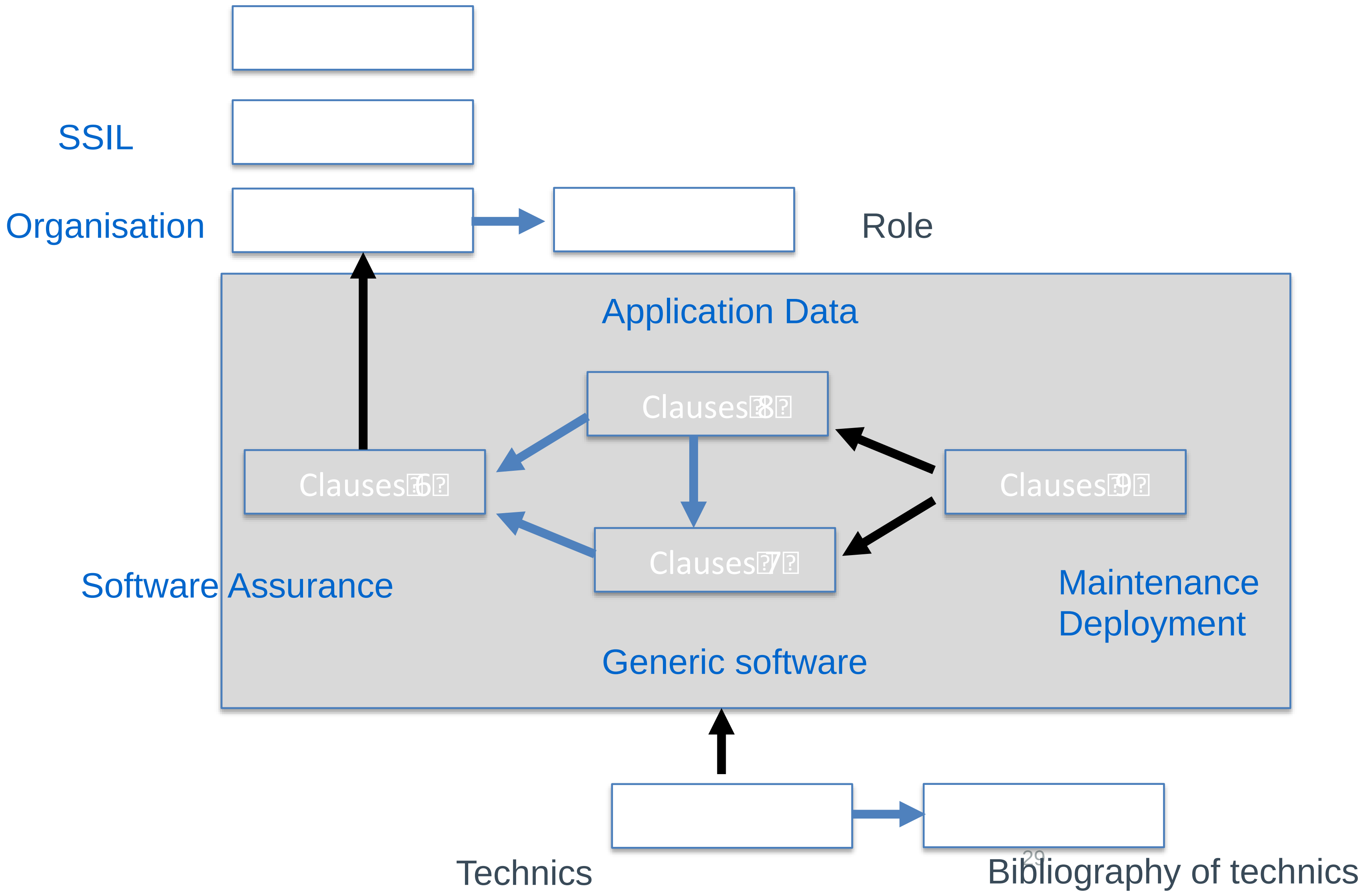
Safety properties
Example

- P1 :Only equipped train which is located and in automatic mode can have a target.
- P3 : The trains locations computed by the SWE must be correct with the actual trains locations on the line

The use of formal method for developing railway safety critical software in compliance with the CENELEC 50128:2011

Jean-louis BOULANGER
CERTIFER
jean-louis.boulanger@certifer.eu

CENELEC 50128:2011



Assurance software

- Assurance software :
  - Competency management;
  - Quality management;
  - V&V;
  - Assessment;
  - Tools qualification.

V&V

Formal proof

Using theoretical and mathematical models and rules it is possible to prove the correctness of a program or model without executing it.

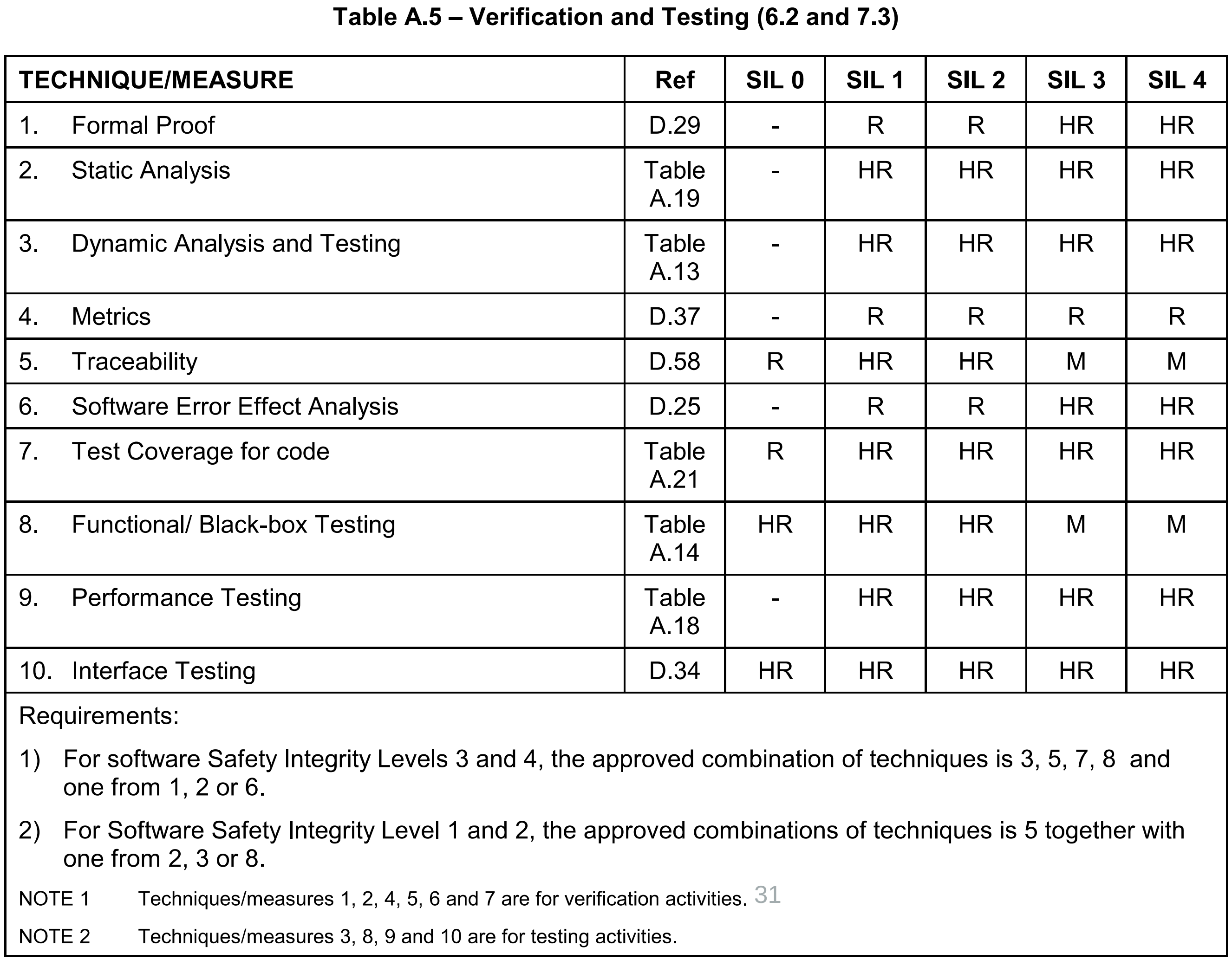


Table A.5 – Verification and Testing (6.2 and 7.3)

TECHNIQUE/MEASURE	Ref	SIL 0	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.

CENELEC EN 50128:2011

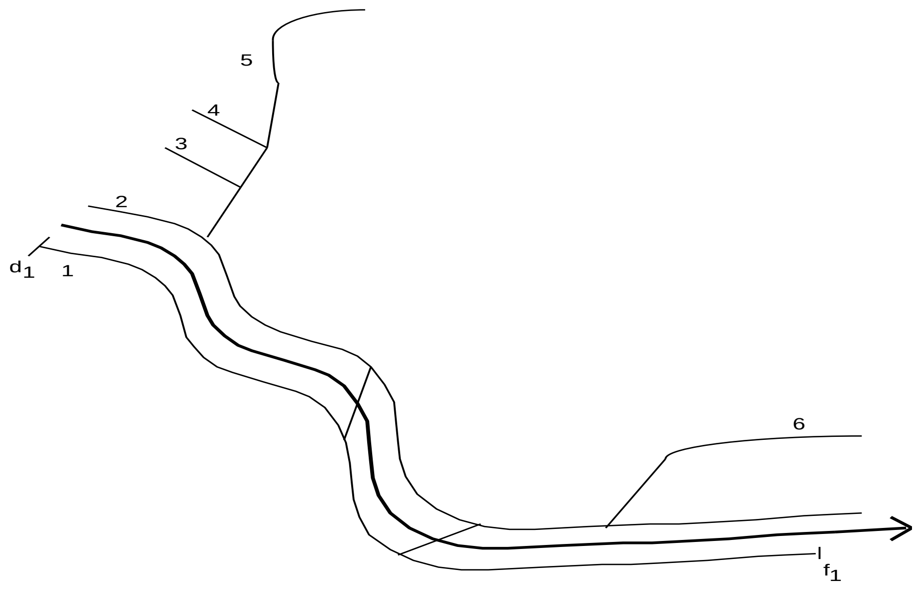
- In the CENELEC EN 50128, the formal method are used at different places
  - For realization (specification, …);
  - For verification.

- There 2 notions :
  - Formal model + Formal analysis :
    - SCADE, B method, etc.
  - Formal analysis:
    - Proof, model-checking, etc.

DATA

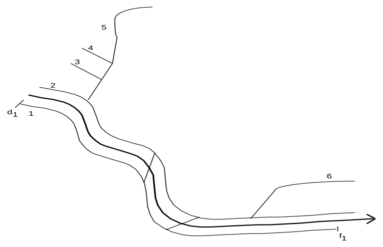
Data

A typical example is a system whose generic software is pre-configured for a generic railway application by a set of application algorithms, and which is then further configured to each specific installation by instantiation and interconnection of the application algorithms and by a set of configuration data. For instance, the signalling principles of an interlocking system (e.g. signal management, point management) may be implemented by a set of application algorithms.



Application data typically take the form of parameter values or descriptions (identity, type, location, etc.) of external objects. Application algorithms may take the form of e.g. function block diagrams, state charts and relay ladder diagrams, which determine the desired response of the system according to its inputs, its current state and specific parameter values. Application algorithms include logical connections and operations to be executed.

Data preparation



First step

DATA system
- Id
- Family
- Format
- Unit
- …

Third step
Data preparation

Second step

Generic Application

specify

DATA
- Id
- Family
- Format
- Unit
- …

Data preparation (3/3)

8.4.1 Application Development Process

- Application Preparation Plan
- Risk analysis on the application development process
- Application Data/Algorithms Verification Report
    - Consistency of the Application Preparation Plan
        - Traceability
        - Software assurance
        - …
    - Verify that the implementation is possible

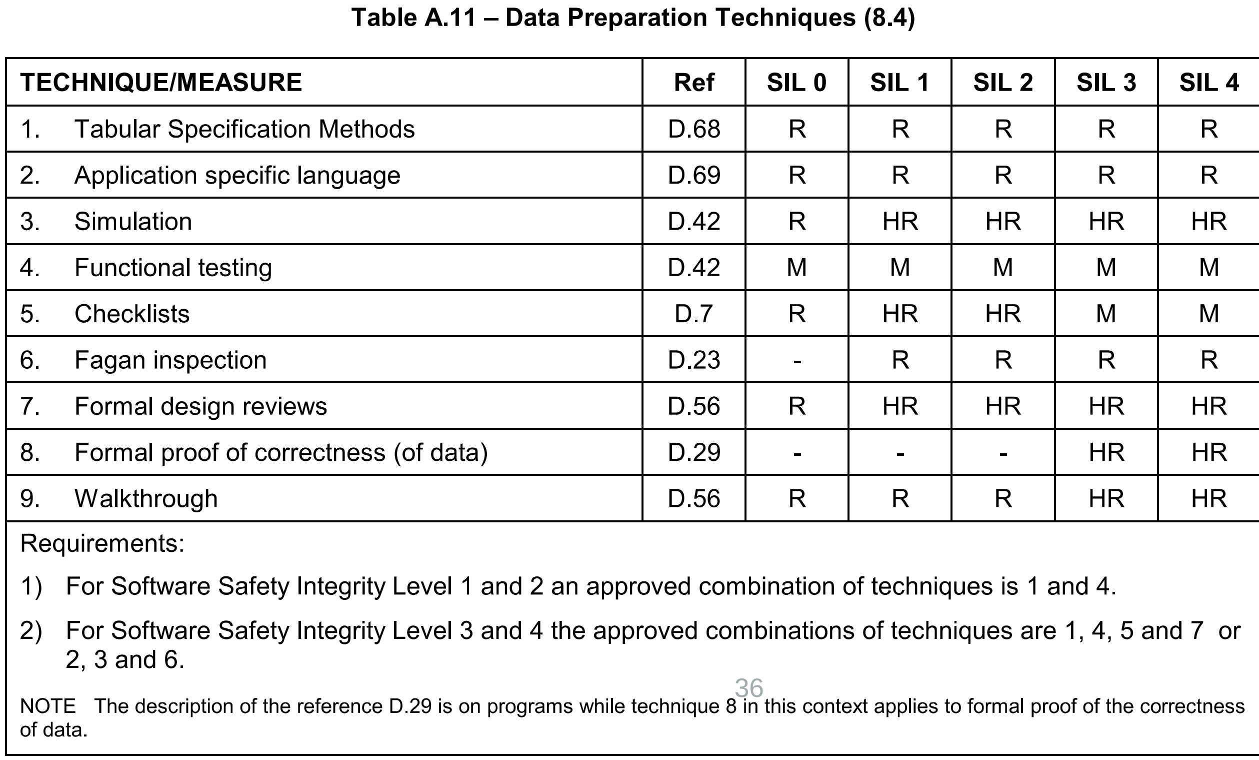


Table A.11 – Data Preparation Techniques (8.4)

TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Tabular Specification Methods	D.68	R	R	R	R	R
2. Application specific language	D.69	R	R	R	R	R
3. Simulation	D.42	R	HR	HR	HR	HR
4. Functional testing	D.42	M	M	M	M	M
5. Checklists	D.7	R	HR	HR	M	M
6. Fagan inspection	D.23	-	R	R	R	R
7. Formal design reviews	D.56	R	HR	HR	HR	HR
8. Formal proof of correctness (of data)	D.29	-	-	-	HR	HR
9. Walkthrough	D.56	R	R	R	HR	HR

Requirements:

1) For Software Safety Integrity Level 1 and 2 an approved combination of techniques is 1 and 4.

2) For Software Safety Integrity Level 3 and 4 the approved combinations of techniques are 1, 4, 5 and 7 or 2, 3 and 6.

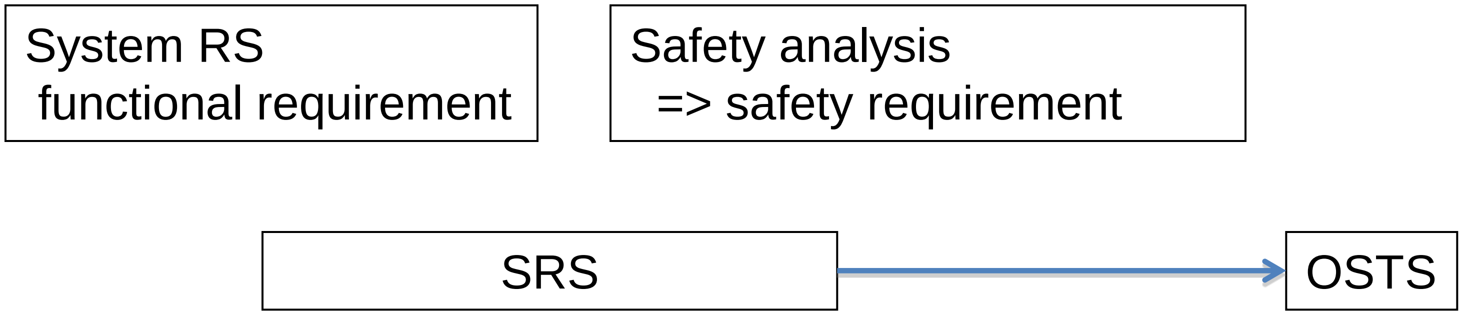
NOTE The description of the reference D.29 is on programs while technique 8 in this context applies to formal proof of the correctness of data.

Generic Software

Software requirement

The Software Requirements Specification is the first step.

In this phase, we produced two kinds of document
-The software requirement specification;
-The overall software tests specification.



Software Requirement

The Software Requirements Specification shall be:

i) complete,
ii) clear precise,
iii) unequivocal,
iv) verifiable,
v) Testable (not all),
vi) maintainable and
vii) feasible,
viii) traceable back to all input documents

Requirement and models

Table A.2 Software Requirements Specification (7.2)

TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Formal Methods (based on a mathematical approach)	D.28	-	R	R	HR	HR
2. Modelling	Table A.17	R	R	R	HR	HR
3. Structured methodology	D.52	R	R	R	HR	HR
4. Decision Tables	D.13	R	R	R	HR	HR

Requirements:

1) The Software Requirements Specification shall include a description of the problem in natural language and any necessary formal or semiformal notation.

2) The table reflects additional requirements for defining the specification clearly and precisely. One or more of these techniques shall be selected to satisfy the Software Safety Integrity Level being used.

"Formal Methods" refer to mathematically rigorous techniques and tools for the specification, design and verification of software and hardware systems.

Modeling

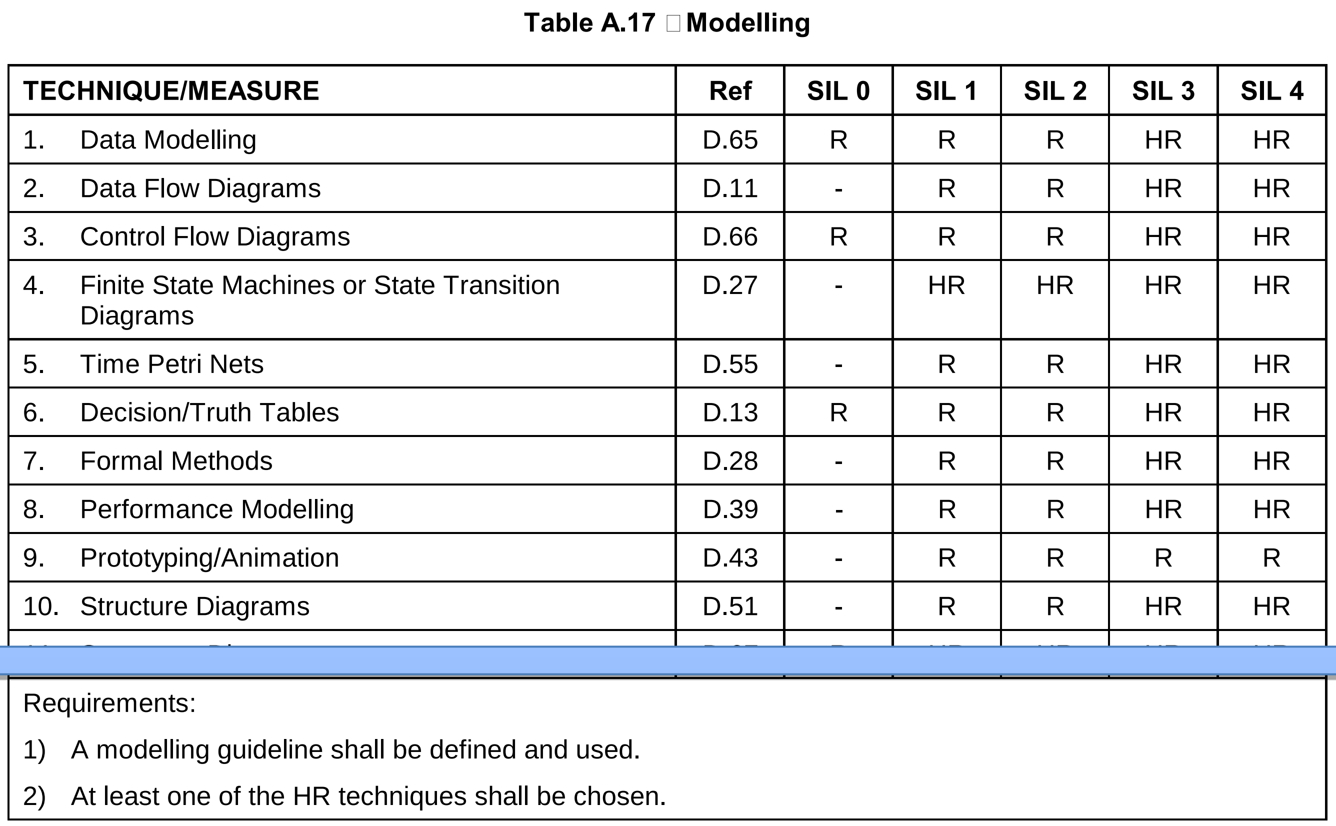


Table A.17 Modelling

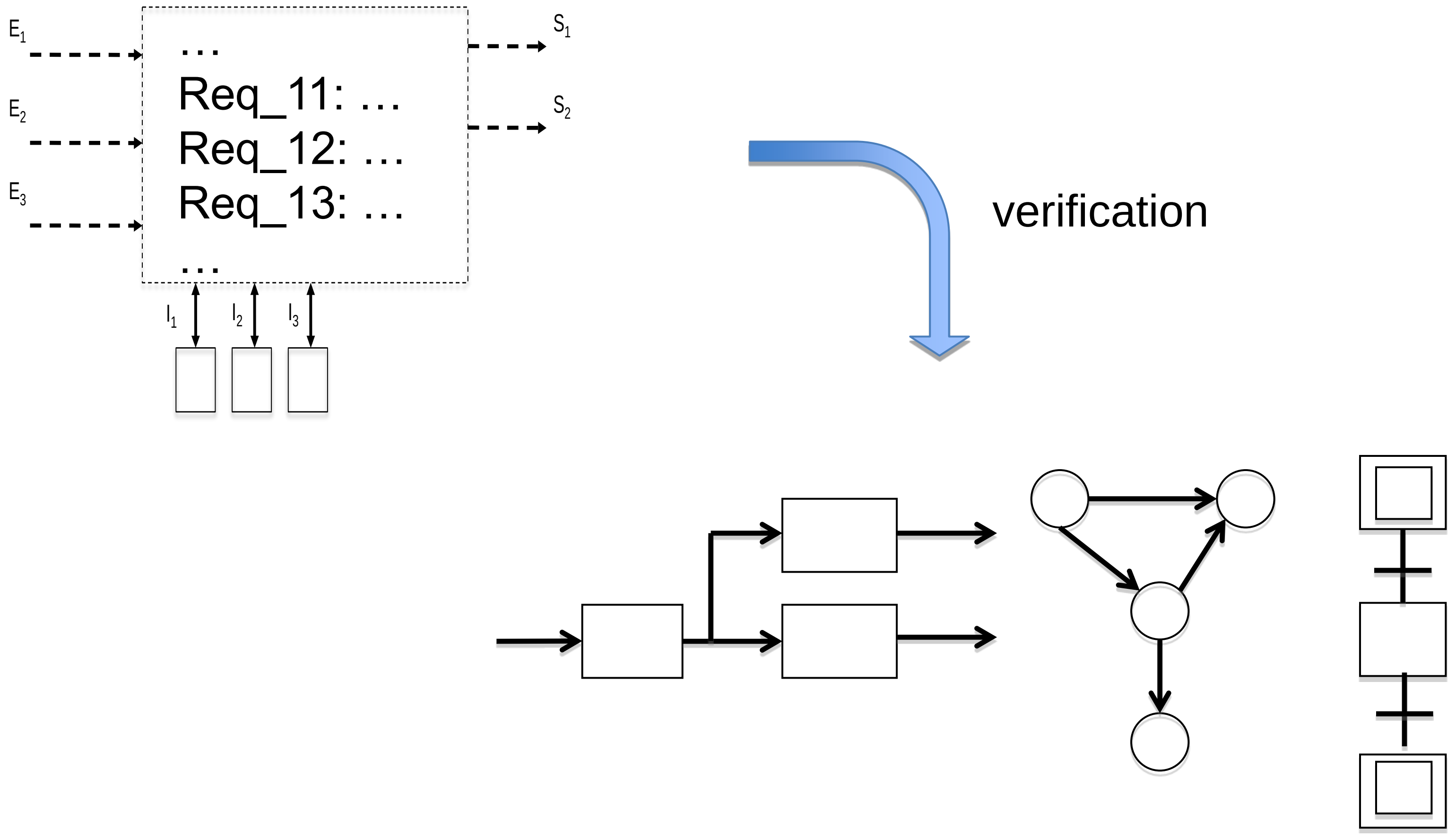
TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Data Modelling	D.65	R	R	R	HR	HR
2. Data Flow Diagrams	D.11	-	R	R	HR	HR
3. Control Flow Diagrams	D.66	R	R	R	HR	HR
4. Finite State Machines or State Transition Diagrams	D.27	-	HR	HR	HR	HR
5. Time Petri Nets	D.55	-	R	R	HR	HR
6. Decision/Truth Tables	D.13	R	R	R	HR	HR
7. Formal Methods	D.28	-	R	R	HR	HR
8. Performance Modelling	D.39	-	R	R	HR	HR
9. Prototyping/Animation	D.43	-	R	R	R	R
10. Structure Diagrams	D.51	-	R	R	HR	HR
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Requirements:

1) A modelling guideline shall be defined and used.

2) At least one of the HR techniques shall be chosen.

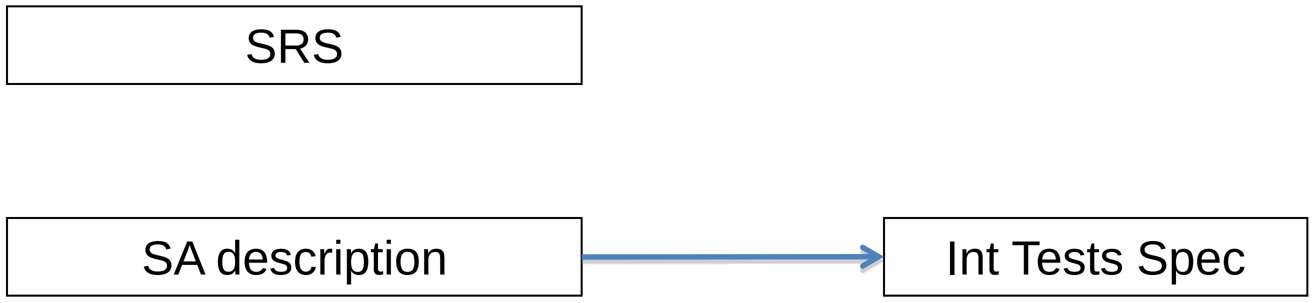
Model for verification



Software Architecture

In this phase, we produced two kinds of document

- The software architecture ;
- The integration tests specification.
    - Software / Software
    - Software / Hardware



Component

component

component is a constituent part of software which has well-defined interfaces and behaviour with respect to the software architecture and design and fulfils the following criteria:

 it is designed according to Components(see Table A.20);

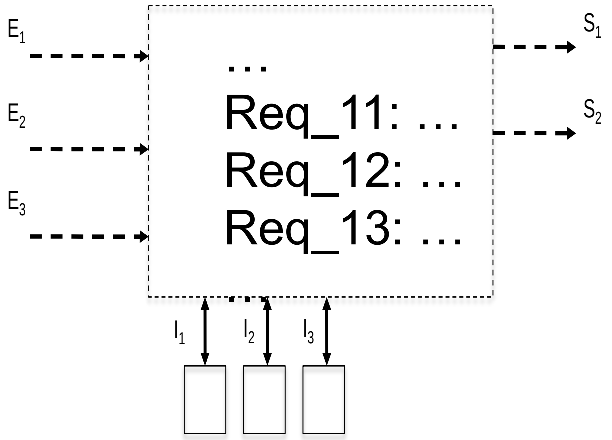
 it covers a specific subset of software requirements;

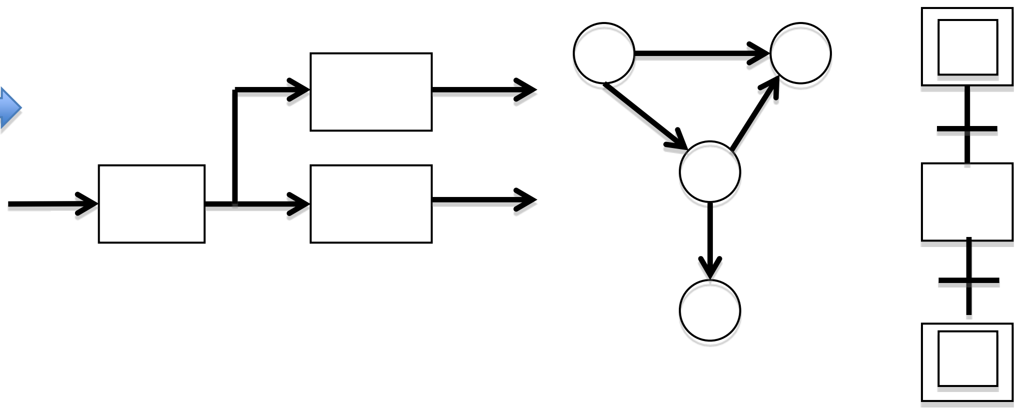
 it is clearly identified and has an independent version inside the configuration management system or is a part of a collection of components (e. g. subsystems) which have an independent version

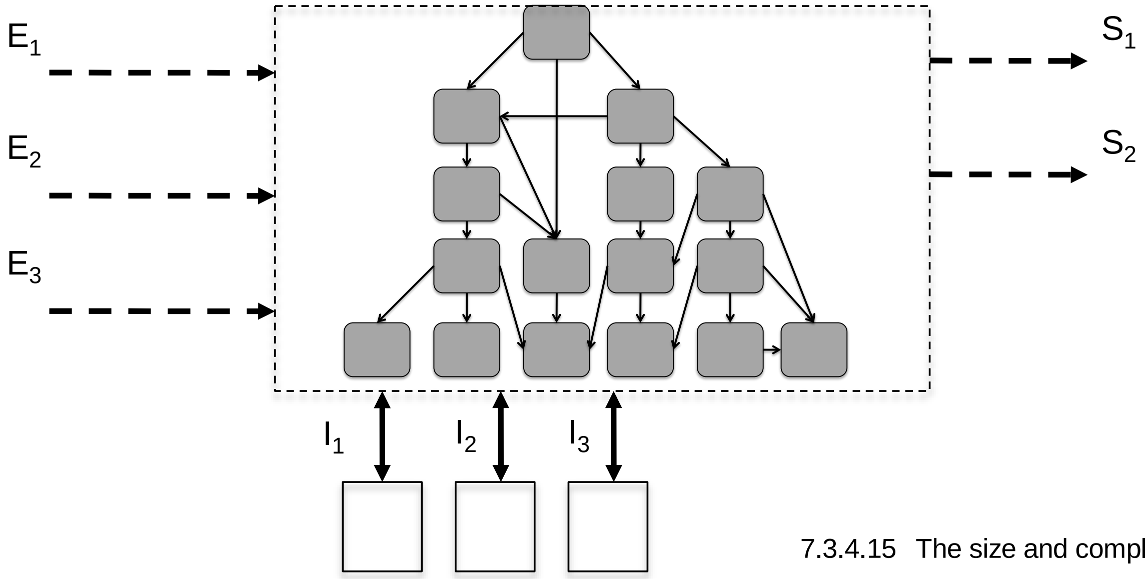
Table A.20 Components

TECHNIQUE/MEASURE	Ref	SIL 0	SIL 1	SIL 2	SIL 3	SIL 4
1. Information Hiding	D.33	-	-	-	-	-
2. Information Encapsulation	D.33	R	HR	HR	HR	HR
3. Parameter Number Limit	D.38	R	R	R	R	R
4. Fully Defined Interface	D.38	R	HR	HR	M	M
Requirements: 1) Information Hiding and encapsulation are only highly recommended if there is no general strategy for data access. NOTE Technique/measure 4 is for Internal Interfaces.						

From specification to architecture







7.3.4.15 The size and complexity of the developed software architecture shall be balanced.

Architecture

Table A.3 – Software Architecture (7.3)

TECHNIQUE/MEASURE	Ref	SIL 0	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 and specification tools	Table A.17	R	R	R	HR	HR

Requirements:

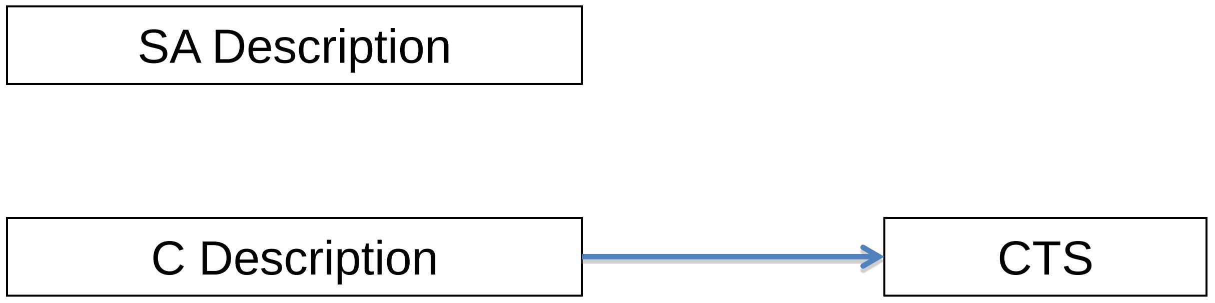
1) Approved combinations of techniques for Software Safety Integrity Levels 3 and 4 are as follows:
   a) 1, 7, 19, 22 and one from 4, 5, 12 or 21;
   b) 1, 4, 19, 22 and one from 2, 5, 12, 15 or 21.
2) Approved combinations of techniques for Software Safety Integrity Levels 1 and 2 are as follows: 1, 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 50159-1 and EN 50159-2.

NOTE Technique/measure 19 is for External Interfaces

Component Design

In this phase, we produced two kinds of document

- The Component description ;
- The Component tests specification.



Design

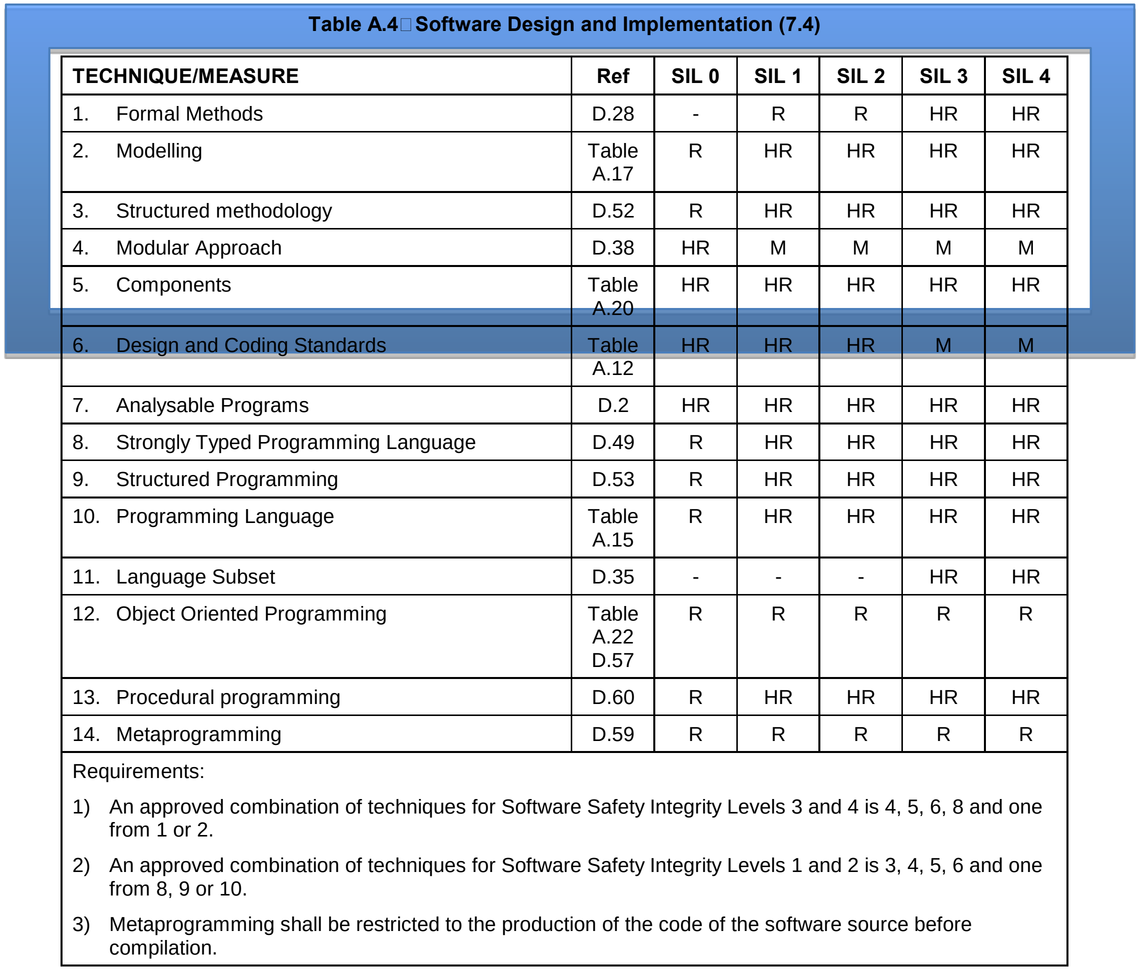


Table A.4□Software Design and Implementation (7.4)

TECHNIQUE/MEASURE	Ref	SIL 0	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

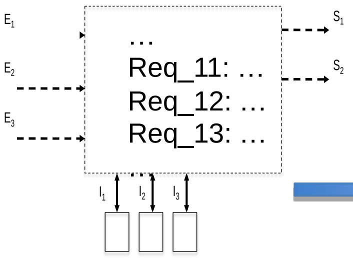
Requirements:

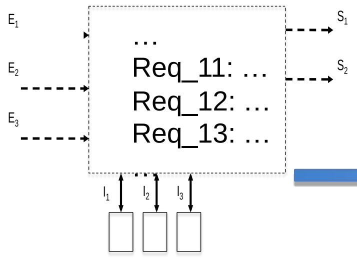
1) An approved combination of techniques for Software Safety Integrity Levels 3 and 4 is 4, 5, 6, 8 and one from 1 or 2.

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.

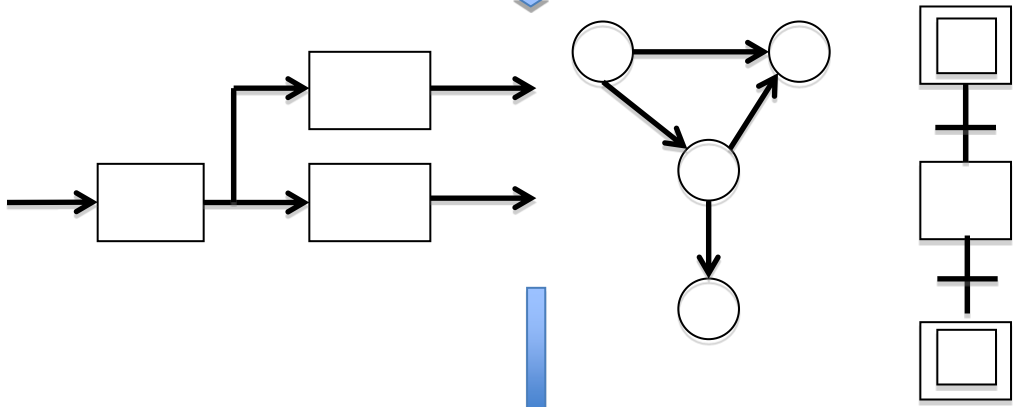
3) Metaprogramming shall be restricted to the production of the code of the software source before compilation.

Manual code vs code generation





Algorithms
Description
Pseudo-programming
If (xx) then …



```
int xx;
main ()
{
…
}
```

```
int xx;
main ()
{
…
}
```

Formal method used

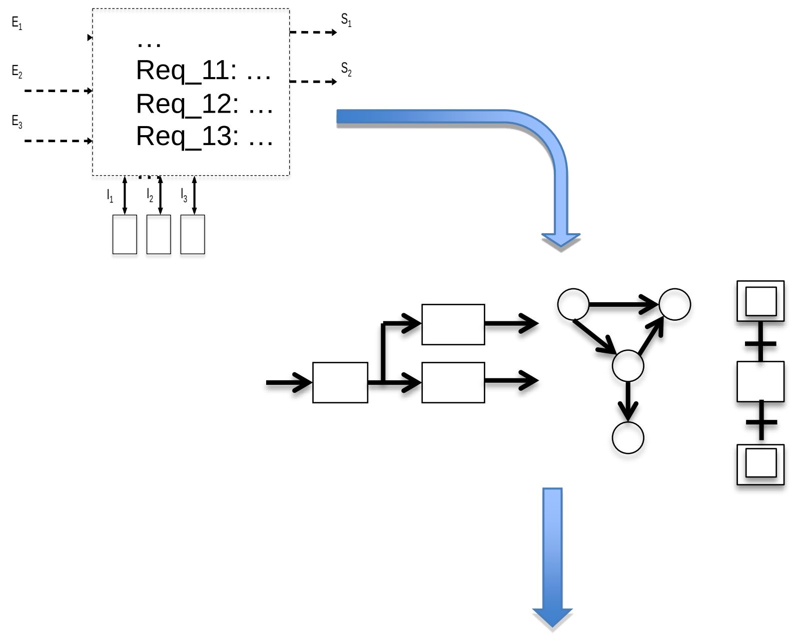
At specification level :
•Model(s) for verification of the completness and coherency

At architecture level :
•To introduce the architecture
•For the allocation of the requirement

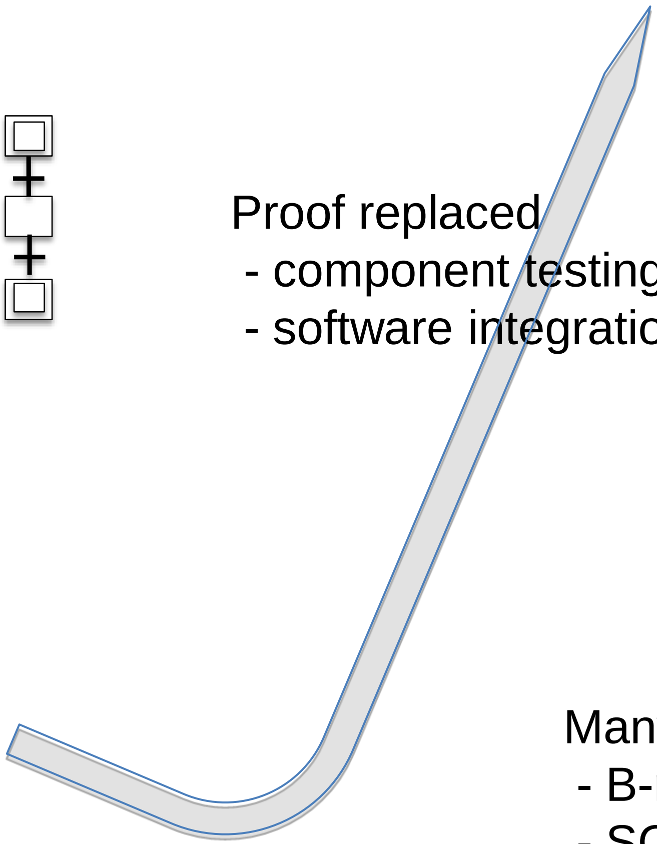
At composant design:
•To introduce the algorithm, the data management, etc.
•For the verification of the requirement

Examples of used

From specification to code



```
int xx;
main ()
{
…
}
```



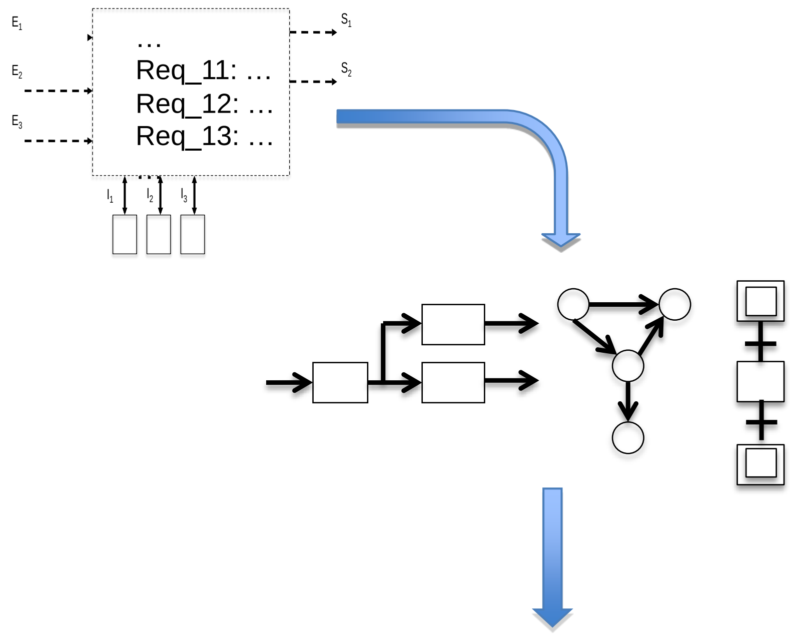
Overall software testing

Proof replaced
- component testing
- software integration testing

Many application
- B-method
- SCADE

CBTC
ERTMS
Train control
…

From specification to code



Overall software testing on target

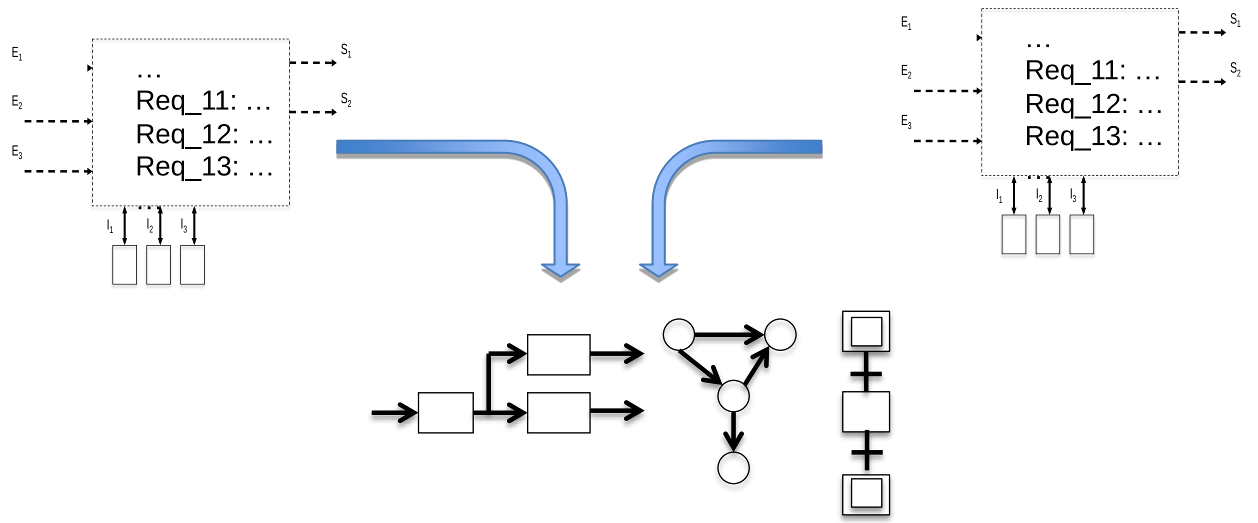
Simulation on model replaced
- component testing
- software integration testing

```
int xx;
main ()
{
…
}
```

CBTC
ERTMS
TCMS
Traction/braking unit

Many application
- SCADE
- Control-build

Verification of safety requirement

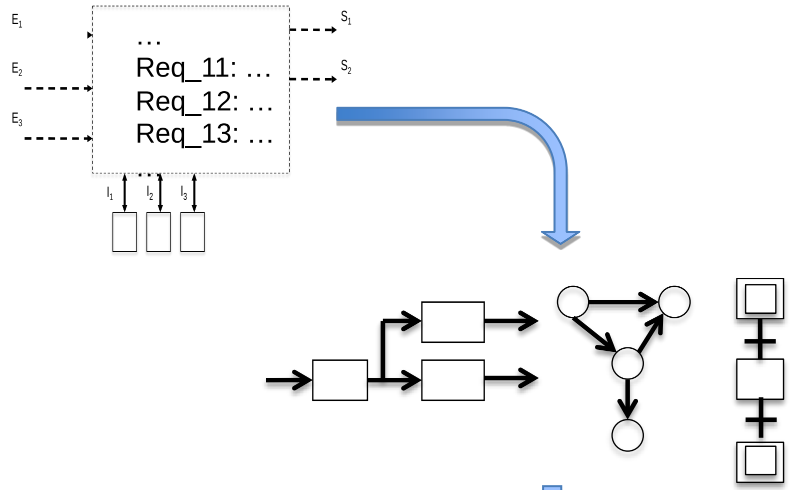


In addition to the overall software tests, we verified that some safety requirement identify at system level are respected by the software in a system environnement

Many application
- SCADE
- Prover
- B-event

Signaling system
CBTC

Proof on the code



Overall software testing

~~S/S Integration testing~~
H/S integration testing

~~Unit testing~~

```
int xx;
main ()
{
…
}
```

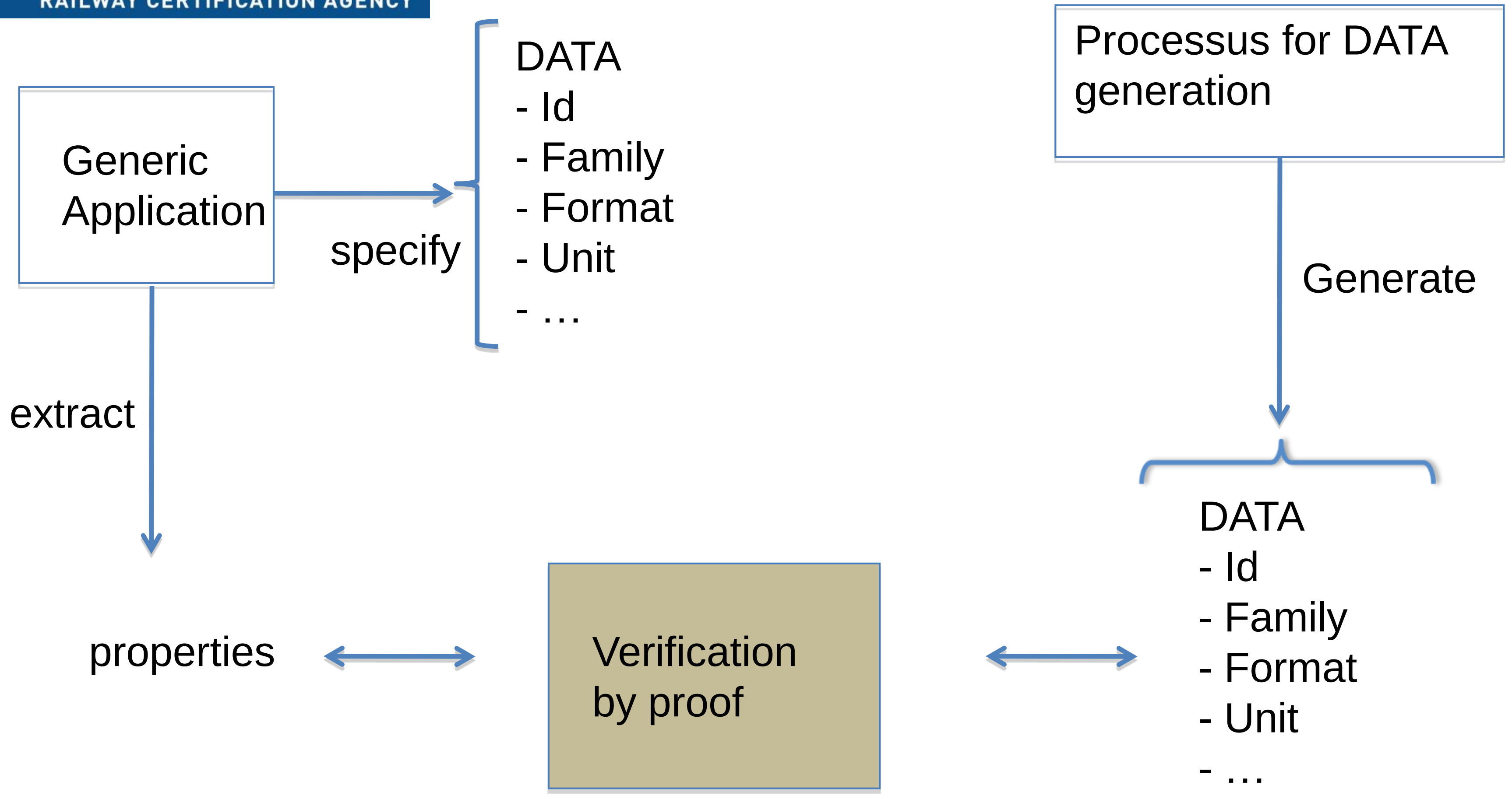
Signaling system

With Pre/Post

Verification of RTE
Properties verification by proof

- POLYSPACE

Proof for Data



Commercial tools : Atelier B, Provers certifier
Open source tools : rodin, ProB,
Specifics tools : OVIP, OVADO

Formal method used

Projet	**Onboard**	**Equipment Sol**	**Date**
SAET-METEOR	atelier B	Atelier B + OVIP	1998
ERTMS mode	SCADE		1999
….			
VAL-CdG (2 lines)	Atelier B	Atelier B	2007
OCTYS L3	Atelier B (Siemens)	SCADE–Proof Toolkit (Ansaldo STS)	2009
PAING		SCADE (ALSTOM)	2011
SAET L1	Atelier B (Siemens)	Atelier B (Siemens)	2011
OURAGAN L13	SCADE (Thales RSS)	SCADE + OVADO (Thales RSS)	2013
OCTYS L5	SCADE (Areva-TA)	Atelier B (Siemens)	2011
PMI L1, L12, L8, ..	N.A.	SCADE–Proof Toolkit (Thales)	2009 ; 2010
LYON	SCADE 5 (AREVA)	SCADE6 + Atelier B + CB (AREVA)	-

Tractions Applications (many, many applications)

MI09	CB (ALSTOM)	2012, 2013
REGIOLIS	CB (ALSTOM)	2013, 2014

Difficulties to fullfill the 50128

1. Documents production
    - How I produce a document ?
    - What is the content of the document ?
        - What is the information I need to introduce if I want to maintain my software during 40 years ?
        - A copy of the model ? More ?
    - The model can replace the documentation ?

2. Complexity management
    - What is a complexity metric for a drawing ?

1. Requirement Management from system to code.

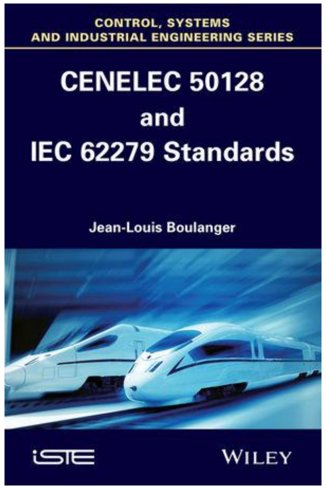
Difficulties to fullfil the 50128

4. Tests management
   - What is a component test ?
     - Tests on code ?
     - Tests on model ?
   - How I measure the coverage ?
     - Link between measure on code and measure on model
     - What is the coverage of the formal analysis ?
     - How combine the formal analysis and the test ?
     - **If proof replace test, the tests coverage is replaced by properties coverage ?**
       - **How I can guarantee that the set of properties cover all my model ?**
   - How I do the integration test ?
     - S/S : simulation, proof, tests
   - How I prepare the overall tests software ?
     - On the model and after on the target ?

Difficulties to fullfil the 50128

5. Competence management
   - Tools, technics, …

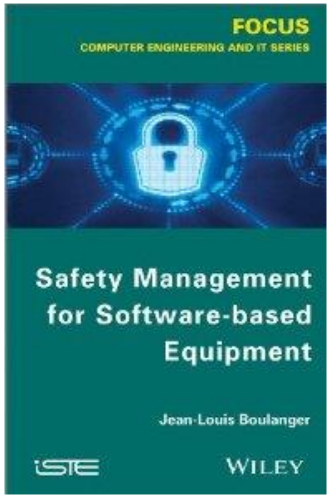
6. Tools qualification


CONTROL, SYSTEMS AND INDUSTRIAL ENGINEERING SERIES

CENELEC 50128 and IEC 62279 Standards

Jean-Louis Boulanger

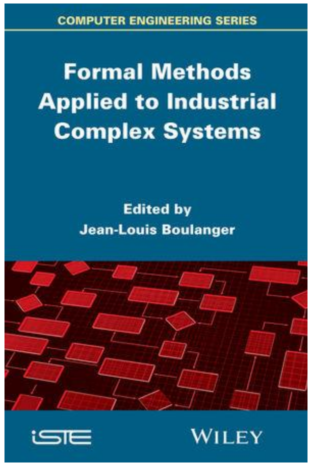
ISTE WILEY


FOCUS COMPUTER ENGINEERING AND IT SERIES

Safety Management for Software-based Equipment

Jean-Louis Boulanger

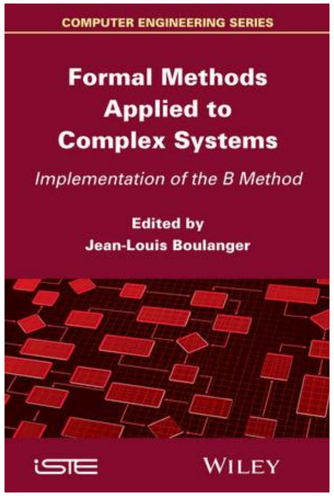
ISTE WILEY


COMPUTER ENGINEERING SERIES

Formal Methods Applied to Industrial Complex Systems

Edited by Jean-Louis Boulanger

ISTE WILEY

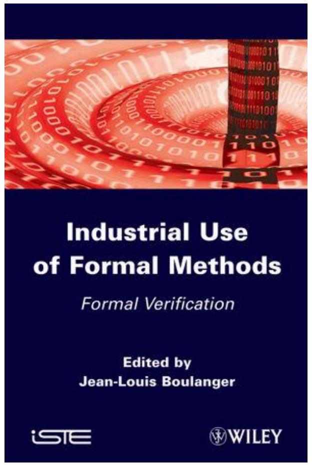

COMPUTER ENGINEERING SERIES

Formal Methods Applied to Complex Systems

Implementation of the B Method

Edited by Jean-Louis Boulanger

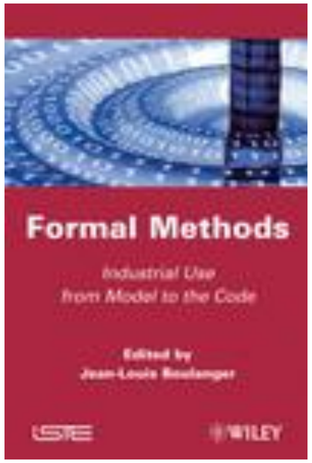
ISTE WILEY


Industrial Use of Formal Methods

Formal Verification

Edited by Jean-Louis Boulanger

ISTE WILEY


Formal Methods

Industrial Use from Model to the Code

Edited by Jean-Louis Boulanger

ISTE WILEY

Static Analysis of Software

The Abstract Interpretation

Edited by Jean-Louis Boulanger

ISTE WILEY

Safety of Computer Architectures

Edited by Jean-Louis Boulanger

ISTE WILEY