III INTERNATIONAL CONFERENCE
TRANSPORT SYSTEMS TELEMATICS TST`03

ZESZYTY NAUKOWE
POLITECHNIKI ŚLĄSKIEJ 2003
TRANSPORT z.51, nr kol. 1608

safety-related systems,
software validation, system tests

Jerzy M. SKRYPKO[1]

SOFTWARE VALIDATION OF THE SAFETY-RELATED SYSTEMS AS AN EXAMPLE SOFTWARE VALIDATION OF THE RAILWAY CONTROL SYSTEMS

This work deals with the tests needs for software validation process for safety-related systems. There is example the railway control systems. The test methods are shown with according to European railway standards. At last there is comparison these railway standards to safety-related systems.

WALIDACJA OPROGRAMOWANIA SYSTEMÓW BEZPIECZNYCH NA PRZYKŁADZIE SYSTEMÓW STEROWANIA RUCHEM KOLEJOWYM

Referat zajmuje się testami wymaganymi dla walidacji oprogramowania systemów bezpiecznych. Podano przykład systemów sterowania ruchem kolejowym. Metody testów są przedstawione zgodnie z kolejowymi normami europejskimi. Na zakończenie podano porównanie tych norm kolejowych z normami systemów bezpiecznych.

1. INTRODUCTION

Author wants to present requirements for software validator with accordance with the electrotechnical standards: IEC 61508 [8] "Functional Safety of electrical / electronic / programmable electronic safety-related systems" and PN-EN 50 128 [6] "Railway applications –Communication, signalling and processing systems – Software for railway control and protection systems". These standards specify procedures and technical requirements for the development of programmable electronic systems. The standard PN-EN 50 128 specifies procedures and technical requirements for the development of programmable electronic systems for use in railway control and protection applications. Both standards specify software validation procedure and the required techniques for the software validation.

Analysing and testing are the main validation activity. The required techniques depend on the software safety integrity level.

[1] Bombardier Transportation (ZWUS) Polska, Modelarska 12, 40-019 Katowice, jerzy.skrypko@pl.transport.bombardier.com

2. SAFETY RELATED SYSTEMS AND STANDARDS

Safety related systems are electrical and/or electronic and/or programmable electronic systems, those are used to perform safety functions. The railway control and protection systems are used to perform safety functions, too.

The standards: PN-EN 50 126 [5], EN 50 129 [7], PN-EN 50 128 [6] set out a generic approach for all safety lifecycle activities for safety-related programmable electronic systems for railway signalling. The standard PN-EN 50 126 is the most overall specification of Reliability, Availability, Maintainability and Safety (RAMS). The standard EN 50 129 defines requirements for the acceptance and approval of safety-related electronic systems. The standard PN-EN 50 128 specifies software requirements. You can find the software validation process description in that document.

The standard IEC 61 508 [8] sets out a generic approach for all safety lifecycle activities for safety related systems. This standard has seven parts. The third part [10] specifies software requirements. You can find the software validation process description in that document, too.

The factors influencing railway safety are:

- systematic failures
- random failures

Software is also source of systematic failures, but software has the influence the time in the system detection and reaction against random failures.

The objective of the software validation process is to ensure that the integrated system complies with specified requirements for software safety at the intended safety integrity level.[10] In the another way: The railway standard [6] specifies more in detail:

- to ensure by analysing and testing,
- to ensure with particular emphasis on functional and safety aspects.

3. REQUIRED VALIDATION TESTS

The main validation method for software shall be testing.[10] The railway standard [6] specifies the main validation activity shall be analysing and testing.

Simulation (animation) and modelling may be use to supplement the validation process.

3.1 REQUIRED TECHNIQUES OF SOFTWARE VALIDATION TESTS

Standard PN-EN 50 128 [6] specifies the following software validation techniques:

1) Probabilistic testing
2) Performance testing
    a) Avalanche/stress testing
    b) Response timing and memory constraints
    c) Performance requirements
3) Functional and Black-box testing
    a) Test case execution from cause consequence diagrams
    b) Prototyping/animation
    c) Boundary value analysis
    d) Equivalence classes and input partition testing
    e) Process simulation

4) Modelling
   a) Data Flow Diagrams
   b) Finite State Machines
   c) Formal methods
   d) Performance modelling
   e) Time Petri Nets
   f) Prototyping/animation
   g) Structure diagrams

Standard IEC 61 508-3 [10] specifies the following software safety validation techniques:

1) Probabilistic testing
2) Simulation/modelling
   a) Data Flow Diagrams
   b) Finite State Machines
   c) Formal methods
   d) Performance modelling
   e) Time Petri Nets
   f) Prototyping/animation
   g) Structure diagrams
3) Functional and black-box testing
   a) Test case execution from cause consequence diagrams
   b) Prototyping/animation
   c) Boundary value analysis
   d) Equivalence classes and input partition testing
   e) Process simulation

You can see that overall standard IEC 61 508-3 [10] does not specify Performance testing as validation technique. This technique is specified as one of two techniques for the HW/SW integration techniques in both standards. The second technique is the functional and black-box testing.

You can find the differences in recommendations among these standards:

Table A.7

Standard IEC 61 508-3 [10]

Technique	SIL1	SIL2	SIL3	SIL4
1) Probabilistic testing	----	R	R	HR
2) Simulation/modelling	R	R	HR	HR
3) Functional and black-box testing	HR	HR	HR	HR

Table A.7

Standard PN-EN 50 128 [6]

Technique	SW SIL0	SW SIL1	SW SIL2	SW SIL3	SW SIL4
1) Probabilistic testing	----	R	R	HR	HR
2) Performance testing	----	HR	HR	M	M
3) Functional and black-box testing	HR	HR	HR	M	M
4) Modelling	----	R	R	R	R

You can see that signalling software with SW SIL 3 and with SW SIL 4 needs Performance testing and Functional and black-box testing as the validation techniques.

Detailed Table A.14 standard PN-EN 50 128 [6] specifies that the use of the "functional and black-box testing" technique is:

Table A.14

Standard PN-EN 50 128 [6]

Technique	SW SIL0	SW SIL1	SW SIL2	SW SIL3	SW SIL4
1) Test case execution from cause consequence diagrams	----	----	----	R	R
2) Prototyping/animation	----	----	----	R	R
3) Boundary value analysis	R	HR	HR	HR	HR
4) Equivalence classes and input partition testing	R	HR	HR	HR	HR
5) Process simulation	R	R	R	R	R

Detailed Table A.17 standard PN-EN 50 128 [6] specifies that the use of the "performance testing" technique is:

Table A.14

Standard PN-EN 50 128 [6]

Technique	SW SIL0	SW SIL1	SW SIL2	SW SIL3	SW SIL4
1) Avalanche/stress testing	----	R	R	HR	HR
2) Response timing and memory constraints	----	HR	HR	HR	HR
3) Performance requirements	----	HR	HR	HR	HR

You can see that signalling software with SW SIL 3 and with SW SIL 4 needs as the validation techniques:

- Boundary value analysis
- Equivalence classes and input partition testing
- Avalanche/stress testing
- Response timing and memory constraints
- Performance requirements

3.1.1 BOUNDARY VALUE ANALYSIS

The aim of the boundary value analysis is to remove software errors occurring at parameter limits or boundaries.

The input domain of the program is divided into a number of input classes. The tests should cover the boundaries and extremes of the classes. The tests check that the boundaries in the input domain of the specification coincide with those in the program. The use of the value zero, in a direct as well as in an indirect translation, is often error-prone and demands special attention:

- zero divisor
- blank ASCII characters
- empty stack or list element
- null matrix
- zero table entry

Normally the boundaries for input have a direct correspondence to the boundaries for the output range. Test cases should be written to force the output to its limited values. Consider also, if it is possible to specify a test case, which causes output to exceed the specification boundary values.

If output is a sequence of data, for example a printed table, special attention should be paid to the first and the last elements and to lists containing none, 1 and 2 elements.

3.1.2 EQUIVALENCE CLASSES AND INPUT PARTITION TESTING

The aim of the Equivalence classes and input partition testing is to test the software adequately using a minimum of test data. The test data is obtained by selecting the partitions of the input domain required to exercise the software.

This testing strategy is based on the equivalence relation of the inputs, which determines a partition of the input domain.

Test cases are selected with the aim of covering all subsets of this partition. At least one test case is taken from each equivalence class.

There are two basic possibilities for input partitioning, which are:

- Equivalence classes may be defined on the specification. The interpretation of the specification may be either input oriented, for example the values selected are treated in the same way or output oriented, for example the set of values leading to the same functional result; and
- Equivalence classes may be defined on the internal structure of the program. In this case the equivalence class results are determined from static analysis of the program, for example the set of values leading to the same path being executed.

3.1.3 AVALANCHE/STRESS TESTING

The aim of the avalanche and stress testing is to burden the test object with an exceptionally high workload in order to show that the test object would stand normal workloads easily.

There are a variety of test conditions, which can be applied for avalanche/stress testing. Some of these test conditions are listed below:

If working in a polling mode then the test object gets much more input changes per time unit as under normal conditions.

- If working on demands then the number of demands per time unit to the test object is increased beyond normal conditions.
- If the size of a database plays an important role then it is increased beyond normal conditions
- Influential devices are tuned to their maximum speed or lowest speed respectively.
- For the extreme cases, all influential factors, as far as is possible, are put to the boundary conditions at the same time.

Under these test conditions the time behaviour of the test object can be evaluated. The influence of load changes can be observed. The correct dimension of internal buffers or dynamic variables, stacks etc can be checked.

3.1.4 RESPONSE TIMING AND MEMORY CONSTRAINTS

The aim of the testing of response timing and memory constraints is to ensure that the system will meets its temporal and memory requirements.

The requirement specification for the system and the software includes memory and response requirements for specific functions, perhaps combined with constraints on the use of total system resources. An analysis is performed which will identify the distribution demands under average and worst case conditions. This analysis requires estimates of the resource usage and elapsed time of each system function. These estimates can be obtained in several ways, for example, comparison with an existing system or the prototyping and bench-marking of time critical systems.

3.1.5 PERFORMANCE REQUIREMENTS

The aim of the testing of performance requirements is to establish that the performance requirements of a software system have been satisfied.

An analysis is performed of both the system and the software requirements specifications to identify all general and specific, explicit and implicit performance requirements.

Each performance requirement is examined in turn to determine:

- the success criteria to be obtained;
- whether a measure against the success criteria can be obtained;
- the potential accuracy of such measurements;
- the project stages at which the measurements can be estimated;
- the project stages at which the measurements can be made.

The practicability of each performance requirement is then analysed in order to obtain a list of performance requirements, success criteria and potential measurements. The main objectives are;

1. Each performance requirement is associated with at least one measurement;
2. Where possible, accurate and efficient measurements are selected which can be used as early in the development process as possible;
3. Essential and optional performance requirements and success criteria are identified; and
4. Where possible, advantage shall be taken of the possibility of using a single measurement for more than one performance requirement.

3.2 THE WAY OF SOFTWARE VALIDATION TESTS

Test cases and their results shall be recorded in a machine readable form for subsequent analysis. If it is practicable, tests should be repeatable and be performed by automatic means.

The test results should allow to state that whole software on its target machine fulfils the requirements or does not fulfil the requirements set out in the Software Requirements Specification.

During validation tests validator have to remember about recording the identity and configuration the used hardware and software and the used equipment. The equipment should be calibrated.

4. COMPARE PL-EN 50 128 AND IEC 61 508

Standard PN-EN 50 128 requires to use validation techniques at Software Safety Integrity Level 4:

1. Performance testing as mandatory
2. Functional and Black-box testing as mandatory
3. Probabilistic Testing as high recommended
4. Modelling only as recommended

Standard IEC 61 508 requires to use validation techniques at Safety Integrity Level 4:

1. Functional and Black-box testing as high recommended
2. Probabilistic Testing as high recommended
3. Simulation/Modelling only as high recommended

The validation activities are described similarly in both standards. There are required the validation plan, the validation report with the documented results.

5. CONCLUSION

1. Standard PN-EN 50 128 [6] specifies requirements for Software Safety Integrity Levels 0 to 4, but standard IEC 61 508 [10] specifies requirements for Software Safety Integrity Levels 1 to 4.
2. Both standards specify SW/HW integration tests and system tests together as SW/HW integration tests and recommend these same techniques.
3. Standard IEC 61 508 [10] does not specify Performance tests as validation technique.
4. These same tests are high recommended for HW/SW integration and these same tests are mandatory for software validation, but any found discrepancies during validation tests should be in any release note, which accompanies the delivered software.[6] 13.4.13.

BIBLIOGRAPHY

[1] SKRYPKO J., Railway application - system dependability, Workshop Proceeding, Gliwice, 1996.
[2] CICHOCKI T., SKRYPKO J., Wprowadzanie wiarygodności i bezpieczeństwa systemów stosowanych na kolei zgodnie z wymaganiami norm Wspólnoty Europejskiej, Systemy czasu rzeczywistego '96, Wroclaw, 1996.
[3] SKRYPKO J., Lista zagrożeń, jako podstawa do specyfikacji wymagań, Materiały VI Konferencji Naukowej - Problemy niezawodności transportu, Ustroń - Jaszowiec, 1997.
[4] SKRYPKO J., Przekształcanie drzewa zagrożeń metodą „Zofii", Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim, CCATIE, Kraków, 1999.
[5] PN-EN 50 126: 2002 (U) Zastosowania kolejowe. Specyfikacja niezawodności, dostępności, podatności utrzymaniowej i bezpieczeństwa (Railway application – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)).
[6] PN-EN 50 128: 2002 (U) Zastosowania kolejowe. Łączność sygnalizacja i systemy sterowania – Programy dla kolejowych systemów sterowania i zabezpieczenia (Railway application – Communication, signalling and processing systems – Software for railway control and protection systems).
[7] EN 50 129 Railway application – Safety-related electronic systems for signalling.
[8] IEC 61 508-1 Functional safety of electrical / electronic / programmable electronic safety-related systems – Part 1:General requirements.
[9] IEC 61 508-2 Functional safety of electrical / electronic / programmable electronic safety-related systems – Part 2: Requirements for electrical / electronic / programmable electronic safety-related systems.
[10] IEC 61 508-3 Functional safety of electrical / electronic / programmable electronic safety-related systems – Part 3:Software requirements.
[11] IEC 61 508-6 Functional safety of electrical / electronic / programmable electronic safety-related systems – Part 6: Guide on the application of IEC 61 508-2 and IEC 61 508-3.
[12] IEC 61 508-7 Functional safety of electrical / electronic / programmable electronic safety-related systems – Part 7: Over view of techniques and measures.

Reviewer: Ph. D. Jerzy Mikulski