Lukáš Špendla

Design for a Testing Model of a Communication Subsystem for a Safety-Critical Control System

Scientific Monographs in Automation and Computer Science

Edited by
Prof. Dr. Peter Husar (Ilmenau University of Technology) and
Dr. Kvetoslava Resetova (Slovak University of Technology in Bratislava)

Vol. 13

DESIGN FOR A TESTING MODEL OF A COMMUNICATION SUBSYSTEM FOR A SAFETY-CRITICAL CONTROL SYSTEM

Lukáš Špendla

Universitätsverlag Ilmenau
2013

Impressum

Bibliographic information of the German National Library
The German National Library lists this publication in the German national bibliography, with detailed bibliographic information on the Internet at http://dnb.d-nb.de.

Author's acknowledgement to Jana Green for translation.

This scientific monograph originated from the author's dissertation thesis defended at the Slovak University of Technology in Bratislava, Faculty of Materials Science and Technology in Trnava.

Reviewers:

Prof. Ing. Juraj Spalek, PhD.
Prof. Ing. Mária Franeková, PhD.
Prof. Ing. Alojz Mészáros, PhD.

Author's contact address:

Ing. Lukáš Špendla, PhD.
Slovak University of Technology in Bratislava
Faculty of Materials Science and Technology in Trnava

Technische Universität Ilmenau / University Library
Universitätsverlag Ilmenau
Postfach 10 05 65
D-98684 Ilmenau (Germany)
http://www.tu-ilmenau.de/universitaetsverlag

Production and delivery
Verlagshaus Monsenstein und Vannerdat OHG
Am Hawerkamp 31
D-48155 Münster (Germany)
http://www.mv-verlag.de

ISSN 2193-6439 (Print)
ISBN 978-3-86360-083-9 (Print)
URN urn:nbn:de:gbv:ilm1-2013100123

Titelfoto: photocase.com

Abstract

This monograph focuses on a proposal for a testing model in safety critical systems. Due to the large scope of these systems we have focused on the system testing and we have included requirements for testing the communication subsystem.

After establishing the theoretical background for testing, we have analysed the issues of safety critical systems. In this analysis we have also focused on industrial networks and their security. Subsequently we have defined the differences and specifics of traditional software systems and safety critical systems, based on standards and guidelines analysis for various safety critical systems. Requirements gained by this analysis have been generalized to use in any safety critical system.

The system testing that we are using is not clearly integrated into the process of design and development of safety critical systems. Therefore we have proposed basic steps of this process to which we have integrated system testing. To capture this process we have used appropriate UML diagrams.

Given the scope of the system testing we have decided to propose two testing models. We have focused on performance and step stress testing. These models implement requirements for testing of safety critical systems specified by us. To capture these models we have used appropriate UML diagrams.

To verify the proposed models, we have defined a metric. Based on its value we can determine whenever the proposed model meets testing requirements specified by us.

Key words

system testing, step stress testing, performance testing, safety critical system, UML

Scientific contribution

The main scientific contribution is the model design of the step stress and performance testing of safety-critical systems. The specification of requirements for safety-critical systems testing and the processing of various support activities were needed during the testing process execution.

In the first part we have analysed the relevant standards and guidelines for safety-critical systems. The analysed standards are focused mostly on the integration testing. The system testing is mentioned as one of the tests which must be executed. The concrete analysed requirements mentioned in the standards are specific so they can not be all included into a design model. Therefore we have specified the requirements considering the selected system testing based on the standard analysis. The requirements which must be considered by a system testing are not contained in any standard or guideline and therefore this specification can be understood as one of the monograph contribution.

Based on the analysed standards we have identified and proposed the basic steps of design and development of safety-critical systems. Every step of our designed model has defined the actions and requirements needed for a process. Then we have identified and specified the requirements and data necessary for a system testing of safety-critical systems. The designed steps as well as specified requirements and data have been integrated into a standard model of design and development of safety-critical systems. This integration of system testing into the process of design and development of safety-critical systems can be also considered as the monograph contribution.

After the specification of requirements for testing and the previous design we have started the model designs of a communication subsystem testing by safety-critical control systems. We have focused on two types of the system tests, specifically the step stress testing and the performance testing. These models come from the identical outputs while every one is focused on a testing of other system aspects. The designed models of testing can be named as the main contribution of this monograph.

The proposed testing models must be verified after their designing. The designed testing models were verified against our specified requirements for testing, considering the fact that the safety-critical systems are a specified area where it is a problem to gain the actual documents. To verify them we have suggested a metrics and we have defined its parameters. The proposed metrics for verification of the designed testing models can be considered as the last contribution of this monograph.

ABBREVIATIONS

ASIC	Application-specific integrated circuit
BK	Safety-critical
E/E/PE	Electrical/ electronic/ programmable electronic
E/E/PES	Electrical/ electronic / programmable electronic system
EUC	Equipment under Control
FAT	Factory Acceptance Test
HW	Hardware
RS	Control System
SAT	Site Acceptance Test
SIL	Safety Integration Level
SI/SIS	Safety Instrumented System
SRS	Safety Requirements Specification
SW	Software

INTRODUCTION

Newly developing environments, languages, tools and accesses of software are continuously being used to develop standard computer systems. New points of access do not guarantee that a created system will be without failures. The question of final product testing is still critical. Testing helps us to uncover errors. Testing of these systems is a relatively well studied area with an amount of general as well as specific accesses. These are applied and approved in practice while points of access for testing of standard systems are concretely defined.

The importance of testing is higher when we focus on the area of safety-critical systems. These require higher consistency in their specification and design as well as their testing. Their failure can have catastrophic results and can harm health, human lives or the environment. A simple unplanned event or event sequence can cause a failure and accident. An emphasis on testing of these systems is much greater than with standard systems, they require different methods, approaches and strategies than in testing of classic systems.

Safety-critical systems do not have only safety functions, but often also functions which do not relate to safety. We have to work with the requirements of testing standard as well as safety-critical systems by testing of these systems. Requirements of standard systems can be used as a basis for the concrete testing process; they must be completed with requirements for testing of safety-critical systems. Although testing of these systems is very important, requirements for testing are not defined.

Testing is mostly focused on standard systems; however testing of safety-critical systems requires a specific approach. Companies operating in this area use mostly internal directives orientated on a concrete area and specific systems. These internal directives are a part of company know-how and it is not possible to get an access to them. Only one available directive for testing of these systems is GAMP guideline which deals with safety-critical systems in the areas of food and pharmaceutical industries.

However there are no completely defined processes of testing in the area of safety-critical systems. Therefore it is necessary to deal with testing of these systems.

Definition of problematic area

There are many standards on the development of standard systems with accent on the importance of their testing. The most important are the standards IEC 12207, IEC TR 15271, IEC TR 16326, IEC 56/575/CD, IEC 14598, IEC 12119 and DIN 66272. However these are focused on testing of standard systems so they do not relate to testing of safety-critical systems. The area of testing of safety-critical systems is so important that it is required to deal with design and development of these systems.

Testing of these systems is a very wide area and it cannot be described completely in this monograph. For this reason we have focused on specific problems of a concrete area, on a communication subsystem and particular problems connected with communication. Based on the typical safety-critical system of a nuclear power plant we can apply our proposed testing model in the highlighted area in ***Fig. 1***.

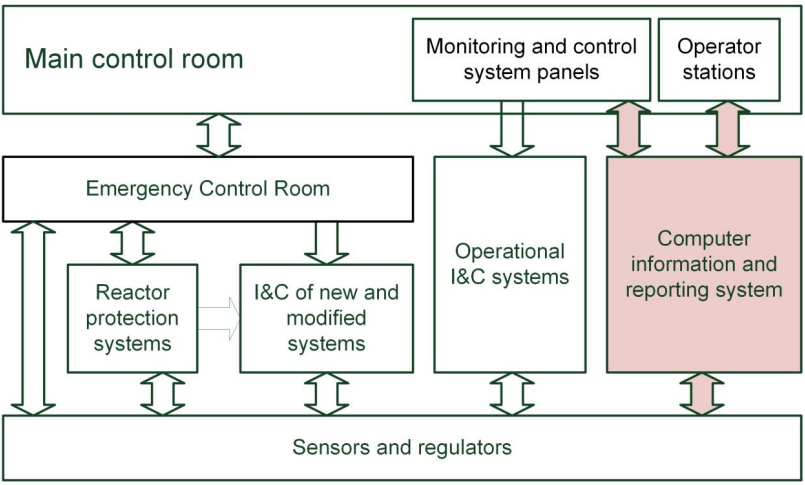

Fig. 1 *Monitoring and control system of a nuclear power plant*

The previous figure does not need to show clearly our communication level. If we had to define it more generally for example in the hierarchy of the industrial control system (1) in ***Fig. 2***, it would be on the Control level. This specification is important because every level contains specific requirements for data communication. Except for them there are various time requirements on communication in concrete levels.

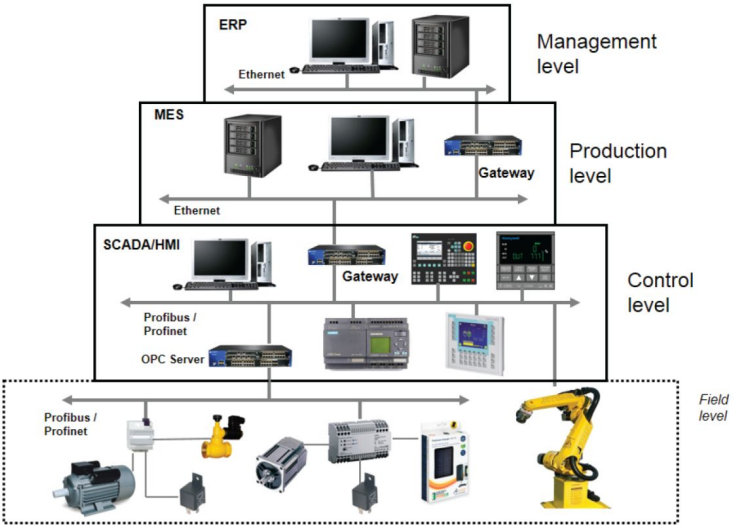

Fig. 2 *Hierarchy of control industrial system*

Because the testing area is also very wide it was necessary to focus on concrete tests which can be applied for testing of the communication subsystem of safety-critical systems. System testing was selected as a type of test which evaluates in a complex way a system behind the limits of its specified requirements or system sources. It must be processed on a completely integrated system and it must follow after integration testing. It is classified as non-functional testing, but it serves for testing of functional and non-functional system aspects. Due to the fact that the selected system testing tests the system as a complex the communication subsystem is one of the tested components.

Aims of the monograph

Regarding the specific problems in this area it is necessary to design a testing model of these safety-critical systems. We focused on specific requirements for testing communication subsystems. It is necessary to know specific aspects before the concrete design of a testing model. At first we had to define specific requirements for testing from the perspective of safety-critical systems. Therefore we have identified and analysed standards containing requirements for safety-critical systems which could provide us requirements for their testing. We have focused on various standards of safety-critical systems.

Concrete standards and directives for safety-critical systems are not clearly connected with a process of design for a particular system and they are focused on concrete safety requirements. Based on analysis of standards and directives we proposed a simple model of design and development of safety-critical systems. We have integrated system testing into the process and we have identified specific requirements which must be included.

Regarding the fact that the system testing includes more specific test types, we have decided to create two models of testing with a focus on different types of test. We have focused on stress and performance testing. Each of these tests presents a different perspective on system testing of safety-critical systems.

The designed testing models were verified after creation to confirm their quality. We have decided to propose a metric to verify; the metric determines correspondence of design to specified requirements.

The concrete aims of this monograph can be summarised into the following points:

1. Defining of differences and specific aspects of requirements on testing of safety-critical systems
2. Integration of system testing into the process of design and development of safety-critical systems
3. Design of performance testing model of communication subsystem testing for safety-critical control systems
4. Design of model- step stress testing of communication subsystem by safety-critical control systems
5. Design of metrics to verify proposed testing models

1. THEORY

There are various processes, methods, approaches for design and testing used in practice. Our selected system testing verifies the functionality of the system as a complex. Therefore they can be processed after integration tests (which means in later development phases) and they belong to the last test level which is made before submission to a customer. The first process step is therefore analysis and specification of the concrete properties of the system testing as well as the software as a complex.

From the perspective of software testing it is necessary to define important terms from this area. We focus especially on the design, testing and implementation into system testing. Because system testing is mostly included into each testing process, it is a permanent part of acceptance testing whether it is a factory acceptance test or site acceptance test. This definition and implementation of system testing is needed to simplify the identification of properties of safety-critical systems which must be included into the designed model of system testing.

It is important to also analyse appropriate types of system tests (or its more specific variant). Stress, performance and load testing belong to the most used aspects in the area of communication testing. This analysis is processed for more detailed specification of a completed test, while important properties can occur and they must be taken into account by its application.

This chapter provides material for selection of appropriate system testing. Requirements and properties will be determined which are a basis

for this class of test. Steps of defining the differences between traditional and safety-critical systems will be highlighted.

1.1 Testing of Software

It is necessary to define some basic terms from the area of software testing. Knowing these terms helps to understand properly the process of software testing.

Testing of software consists of a dynamic verification of programme functioning on the final group of testing scenarios. This is selected from a nearly infinite set of processing domains in regards to a specified expected process (2).

Failure is defined as a deviation of real functionality of SW system from the required functionality. This means we are talking about failure occurring when the tested programme does not react properly (3).

A **testing scenario** is characterised by the initial state of the tested object, testing inputs, conditions and expected results. The base of the testing scenario by specifically orientated testing is an action sequence needed to test certain functionality with a concrete SW. Input data are the basis of the testing scenario in implementation orientated testing. These data enter a certain code part to be tested (4).

A **test** can be defined as a programmed testing scenario which can be started in a testing environment. A test is therefore an executable presentation of the testing scenario. The test was successful when an expected result defined in the testing scenario was identical with a real result after initialization and evaluation of results. In the opposite case the

test was unsuccessful and we are talking about a failure in running of the tested SW (4).

1.1.1 Verification and validation

Verification of software proves if a developed system was correctly developed considering its specification. The process of verification uncovers failures in the SW system but not their absence or missing of some concrete functionality. Verification of the system can be static or dynamic (2).

Static verification is the static control of a proposed specification, the control of a programmed source code and others. The technique of static analysis is used for these purposes such as for example supervision of models, analysis of data flow and supervision of source code (2).

A dynamic process of a developed system is studied by dynamic verification and it is evaluated with regard to the required running of this system defined with its specification (5).

Validation deals with verification of the specification of a developing SW system, with consideration of the real expectations and requirements of the customer. The basic question in the process of validation is if a developing system is the one which we really want and have. Validation uncovers a poorly defined or completely missing functionality of system. This process is made also with the participation of the customer and analysis of user requirements, creation of prototypes and others are applied by it (4).

1.1.2 Elementary attributes of software quality

To estimate the quality of software systems many various attributes can be used. Many of them are considered as a system type – we can apply them by a whole system or they can be spanned through more parts of the software system. What defines an important set of qualitative attributes is dependence on objective perspective (6).

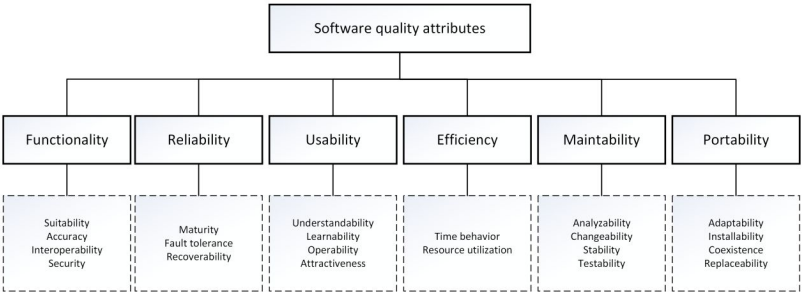

***Fig. 3** Attributes of software quality*

The most applied qualitative attributes are illustrated in ***Fig. 3***. These attributes present for example concrete spheres which are an object of system testing. The standard ISO 9126 defines various metrics for each of these spheres; the metrics can be applied to evaluate particular attributes. There is a close link between attributes of software quality and testing (6).

1.2 Acceptance testing

Acceptance testing is a type of testing processed to verify if a concrete product is developed according to standards and specified criteria and if it

follows all requirements of the customer. This testing is one of the types of testing with the method of a black box, when the user is not interested in the internal functionality/code of the system but he/she evaluates the complete functionality of the system and compares it with the user specified requirements (7).

Acceptance testing is also known as validation testing, final testing, QA testing, FAT, application testing, etc. In software engineering acceptance testing can be processed in two levels: one on the level of a system supplier and the second on the level of the final user (also called UAT, FAT or testing with a final user (7).

The output of acceptance testing can be expressed as a success or failure on critical operation conditions which the system has to follow successfully or not and a final user evaluation of a system (7).

1.2.1 Factory Acceptance Test (FAT)

The main aim of FAT is to test the security of safety-critical systems (solving logic with particular software). Tests are processed during the last phase of the design before final installation. FAT follows a control procedure of safety-critical systems and functions according to the specified requirements on safety (8), see ***Fig. 4***.

FAT contains appropriate general testing processes to verify the correct functionality of a safety-critical system. Testing activities are general methods. Therefore FAT can be applied in programmable as well as non-programmable safety-critical systems. The most important part of FAT deals with specification of testing cases; this means clear description of

testing cases, well-structured testing processes and relevant cases of testing (8).

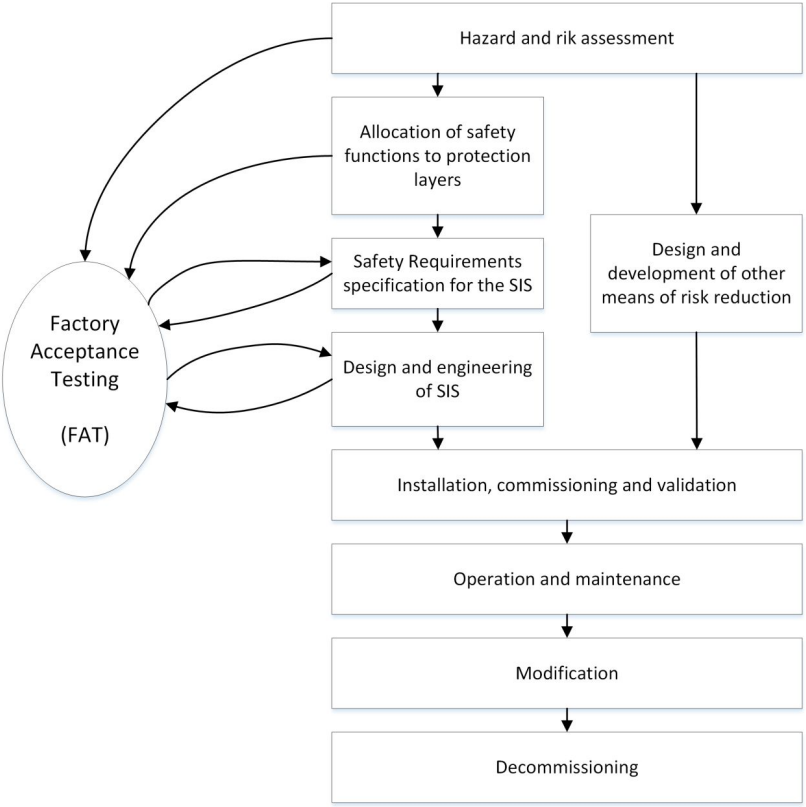

Fig. 4 *Factory Acceptance Testing*

Planning

Planning presents a set of appropriate tests which must be processed and which are responsible for development of testing cases. Appropriate levels of competencies and independence of evaluation are required.

Concrete test processing should be described as well as the responsibility of testers. The testing protocol is created during planning. It is necessary to specify who approves a testing protocol (8).

The process of FAT should be defined and documented with a correct way. Every testing process must be described with a logical sequence. This means how to test the application software and hardware. Necessary competencies of contributing staff are described during planning. It is recommended that staff members with appropriate experience in the process and safety-critical system should deal with the planning of FAT. Experience creating various spheres, such as for example the process of design, design of hardware and software, can contribute to FAT planning with relevant testing cases. The content of the FAT tests should be appropriate for the planned safety-critical system (8).

Planning contains processes for repair activities in the case of failure during testing. Planning of tests should contain also criteria used to evaluate tests. Physical allocation of safety-critical system and functional dependence on other systems are questions which must be specified during the phase of planning (8).

To develop integration tests it can be necessary to contact the supplier of the solution logic or other relevant suppliers for the used parts in safety-critical systems (8).

Testing activities

FAT is a standard process made with a producer on the place of manufacture (development). The producer verifies that the safety-critical system is running according to plan. This means it follows requirements

defined in specification of safety requirements (SRS). During the FAT the producer verifies if it is possible that (8):

- Applied equipment is in accordance with specifications (compatible HW and version SW)
- Applied equipment is installed according to the producer specification
- Inputs and outputs are installed according to plan
- Equipment is correctly calibrated
- Solving logic and appropriate software works according to the requirements in SRS
- Outputs and their actions follow SRS
- Resetting of functions work according to the SRS
- Alarms follow SRS
- Operator functions follow SRS
- Bypass functions work according to the SRS
- Functions for manual switch off work according to the SRS
- Diagnostic functions of alarm follow SRS

Outputs of safety-critical functions are observed during various testing cases. For example, a simulation of inputs is done to verify that safety-critical functions fulfil the requirements in SRS. If the FAT uncovers weaknesses in a phase of proposal and design, or in a phase of SRS, it is necessary to modify these phases according to the results of FAT. To check a modification, the safety-critical system must be tested again (8).

The FAT should be processed in defined versions of the logic of solutions and its configuration must be specified to gain relevant testing cases. To process the tests it is necessary to describe the required tools and

testing environment. Any modification or change should be an object of safety analysis to consider its influence spectrum on every safety-critical function. It is necessary to define and implement again a range of testing (8).

Test Result

All testing cases of FAT must be described in documentation and also if the aims and criteria of test were fulfilled or not. Discovered failures during test and failure causes are documented and supported with necessary activities to repair failures (8).

If the SI system was modified a safety analysis must be processed to find out if safety was influenced and if a new testing is required (8).

1.2.2 Site Acceptance Test (SAT)

This variant of acceptance test is usually processed at the operation place after installation. The customer can require a successful implementation of SAT before the real operation of the system. Demonstration that the system and application work properly with specific determination places plays an important role (9).

Testing activity of SAT can be identical or slightly different than the FAT activities. Verification of particular testing cases on compliance with requirements and criteria has the same role by both acceptance tests. The only one change is the place of system testing (9).

Activities during the SAT are (9):

- Verification of functions which could not be fully evaluated during system tests
- Control of local modifications / repairs
- Control of support systems
- Verification of documents / processes
- Enabling of process acceptance tests on key places with the user/developer
- Demonstration of contract requirements with a part of the product or the complete system.

1.3 System testing

System testing can be defined as a testing phase on the completely integrated system to consider the correspondence of the system with specified requirements. It is processed in the phase of testing of modules, components and integration testing (10).

A system is a complete set of integrated components which give functionality and properties to a product. If we want to test a whole system we have to understand the product function as a complex. System testing helps to uncover failures which cannot be directly associated to a module or a range. System testing shows problems which are at the level of design, architecture and code as a whole (10).

System testing is only one phase of testing which tests functional and non-functional aspects of a product. It is focused on the customer's use of a product and solutions with simulation of customer requirements. From the

perspective of the non-functional aspect it brings various types of testing (called quality factors). Here are some of them (10):

- **Performance/Load Testing**

  Performance testing evaluates the process time or time of system feedback, which is needed to precede required functions in comparison with various versions of the same product or products of different producers.

- **Stability Testing**

  Stability testing is used to find maximal abilities of system parameters which require an enormous number of sources.

- **Reliability Testing**

  Reliability testing serves to evaluate the ability of a system or an independent system part. Ability includes a repeated function process during a particular time (10).

- **Stress Testing**

  Stress testing evaluates a system behind the borders of its specified requirements or system resources (as it is a space on disc, memory, load of processor) to secure that system will be not abruptly damaged (10).

- **Interoperability Testing**

  This testing is processed to secure information exchange between two or more products, an information application and close cooperation (10).

- **Localization Testing**

  This testing verifies that a located product works properly in different languages (10).

The definition of system testing can be modified to cover wider aspects on higher levels while it is dependent on the context (10).

1.3.1 Stress Testing

Stress testing is one of the system tests. Stress testing represents testing of software or hardware on its efficiency in concrete conditions or sufficient performance under extreme and bad conditions. Stress testing helps to determine the level of robustness, reliability and balance or satisfactory performance in the system. This is also tested beyond the limits of normal system operation (software and hardware) (11).

The most important application of stress testing can be found in testing of software and hardware which should work also in critical situations. Stress testing with particular software is considered as an efficient process to determine the limit when the system/software/hardware has the property of robustness, it is always ready to fulfil tasks, to manage efficiently the load, and it shows an efficient administration of failures in extreme conditions (11).

Stress Testing (Fully Censored Test)

Standard stress testing is based on fulfilment or non-fulfilment of the required aim which presents reliability. The test is designed around hypothesis: a product has a given reliability with a given confidence. The hypothesis is tested by determination of how many parts must work without failure during the defined time by a certain severity. This test type is also called fully censored. If the test is successful the process of testing is

finished sooner than some component fails. We do not gain any relevant test results (12).

The precision of stress testing depends on two key factors (12):

1. Number of tested elements
2. Number of tested life cycles.

The greater number of tested elements, the more precise the test is. Increasing the number of tested life cycles declines the number of needed elements, but it is necessary to implement an estimation of the distribution in the amount of tested elements (12).

A fully censored test is a test of the type pass/fail. Therefore its accuracy cannot be defined in a simple way. However the main factors with an influence on it are identified according to the (12):

1. Importance of tests regarding operation conditions
2. Tolerance for repetition of testing conditions

Step stress Testing

Today more methods of accelerated testing are on the first position. Stress testing, as a type of accelerated testing of reliability, belongs to the accelerated life testing. The main idea of accelerated testing is that materials are exposed to higher than usual conditions to avoid early failure. Data gained from these experiments must be extrapolated to estimate a lifetime distribution under normal conditions (13).

Step stress testing can be considered as accelerated extension of stress testing with an aim to improve some of its aspects. Step stress testing starts in same way as a completely fully test. Firm number of elements goes through life cycle which is for example 500 h. Load applied on a product

was continuously increased after the first life cycle. The purpose of load is to gain a failure of tested elements (12).

Precision of the step stress test is connected with three key factors (12):

1. Number of tested elements
2. Number of tested life cycles
3. Method of load increase

The first two factors are identical as by the fully censored test. A high value of time to failure is declined by a high increase of load. Time needed to accumulate failures is very long and differences of particular times to failure are very big by a low increase of load. A rate of damage is increased by a growing load, time to failure is lowered and a range between times to failure is shorter (12).

Accuracy of the step stress testing depends on an accuracy of the first and the second part of test. Accuracy of the first part is connected with an accuracy of fully censored test. Accuracy of the second part depends on a method as increased loads (12).

Repeatability of the step stress test is relatively high while there are some limiting factors regarding to a test character (12):

- Real load distribution of tested elements is not determined
- Test can last relatively long time
- Test can be very misleading if an importance of test is not accurate regarding to operation conditions

Time of test duration depends on a length of one life cycle which is defined on base of its technical analysis. Test can last few hours or some months. Fully censored test usually lasts a shorter time for simple load

conditions and a longer time for more complex load conditions. Step stress testing takes in general two times more than a fully censored test which is a base for the test (12).

1.3.2 Performance testing

Performance testing is a type of system testing which determine times of different critical company processes and transactions which can be compared. Load of system is low during the performance testing but the system operates with a database of production size. This method of performance testing determines "the best possible" performance in a concrete configuration or infrastructure. This test helps to identify necessary modifications in testing process which must be made (14).

In practice there are mostly used performance tests with automation tools. They help to minimize influences from perspective of users so they can secure a high accuracy of repeated measurements. Advantage is that the same testing scripts can be used later by the load testing. Gained results of load testing can be compared with original results of performance tests (14).

A key quality indicator of performance testing can be repeatability. Repeating of the same performance test should provide a same set of results for identical tests. If results are not always identical then differences between concrete performed tests do not relate to changes of application, configuration or environment (14).

Specific Aspects of Performance Testing

Performance tests contain some specific steps which must be considered by testing. An important step of planning is for example a defining of aims before the next process, specific aspects of testing data and determination of result evaluation (15).

It is necessary to define the aim and used metrics before concrete testing. These reference values are the base for optimization and selection of appropriate testing techniques. Applied criteria should correspond with requirements for example a complex intranet application should not be given to 10 000 parallel users. Performance requirements should be defined in a phase of specification while it is necessary to verify their consistency and validity (15).

The next important theme which is often forgotten is testing data. Ideal performance testing of performance should be made on a system operating with real data. These data are comparable with a volume of production data. In the most cases it is not possible to gain testing data from a production system. In these cases a lot of data must be artificially generated (15).

Real volumes of data are a base for meaningful performance testing. Obstacles in performance connected with database can be often recovered only with a help of data load (15).

To correctly evaluate and present test results it is important to know data which must be documented. There is no general method for analysis and correction of data (15).

1.3.3 Load testing

Load testing is a variant of performance testing. The difference is that testing is processed only by a full load. The aim of load tests is the time determination of various critical operations and company processes which are used to verify expected parameters. They enable also evaluation of application possibility by a load and they help to operate correctly with documentation of correct, incorrect and failure operations. Regarding the test character of a wide input from real operation is needed for an accurate simulation of load (it is possible to use also generated data) (16). Operations which must be included into testing are an important factor for results of load testing. Basic classes of functions or processes which should be included into a test are in the Table 1.

OPERATION GROUPS FROM PERSPECTIVE OF LOAD TESTING Table 1

Operation group	Description
Often repeated operations	If they are not sufficiently efficient, they have a potential to influence a performance of all operations.
Critical operations	They have an influence on basic aims of system and their failure by load has the strongest influence on a system performance
Operation of reading	Test should include at least one operation processed only for data reading
Operation of modification	Test should include at least one operations processing data modification

The purpose of every load test should be clearly described and documented. Mostly we can divide it into one of two categories (16):

- Determination of risk
- Determination of minimal configuration

We estimate a probability by a risk determination that a system performance will follow formal performance requirements (for example required times of feedback by a certain load). Load testing does not reduce a risk directly but with a help of identification and evaluation of risks, a proposal of possible modifications and impulses for steps to minimize a risk (16).

We look for a minimal configuration by a determination of it. This will enable fulfilment of required formal performance requirements. It is especially a minimizing of hardware, software and other costs (16).

2. ANALYSIS OF PROBLEMATICS OF SAFETY-CRITICAL SYSTEMS

It is needed to define basic terms before a determination of differences and specific aspects between traditional and safety systems. Firstly it is needed to define a relation between safety and safety-critical systems. These terms are connected together and they are not complete without each other. Except of them basic spheres of safety aspects are designed in dependence on type of danger.

Safety-critical systems in understanding of context of our testing belong to safety-critical computer systems. While the scheme mentioned in this part is kept also by standards for safety-critical systems. Basic advantages and disadvantages of application of these computer systems are also formulated.

One of the most specific properties of safety-critical systems is a real time which is included in requirements for nearly all safety-critical systems. We do not need only to reach correct results but we must also get them in a correct time. Except of that there are more types of system resources which are used, categories where a system belongs to, types of a real time and planning algorithm for deciding. These properties can be included and applied in a designed model of testing.

Because our designed model of testing is orientated on a communication subsystem, it is important to focus on industrial networks and their safety. There are more types of industrial networks and their protocols used for safety of industrial networks.

Results of this chapter helped us to see better a testing of safety-critical systems. These contain some identical properties as standard

systems and therefore processes for their testing do not need to differ. They must implement requirements for testing of these systems with standards and guidelines. Except of that we can better understand aspects which must be considered by testing of communication system.

2.1 Safety and safety- critical systems

A term of safety is a very important term of safety-critical systems. From the perspective of safety-critical systems it is possible to define safety as a property of system which will not threaten human lives or environment. The more general definitions define a safety as a situation when risk is reduced to an acceptable level. Following this definition there is no absolute safety or a zero risk. It is not valid only for safety-critical systems but also for all spheres of life (17).

Based on these definitions of safety we can formulate a term of safety-critical system. It is a system when its incorrect functionality (failure) can have huge consequences as loss of lives, heavy injury, damages on environment and others. This definition says that every system can be named as safety-critical system when human or environment can be hurt by unspecified way (17).

In dependence on type of danger particular aspects of safety can be divided into three spheres (17):

- **Primary safety**

  This includes all aspects corresponding with danger from system´s hardware. For example a protection of user from electrical shocks, a reduction of fire risk (which can be in hardware) (17).

- **Functional safety**

  It is important by safety-critical systems. It includes an evaluation of possible danger sources which can be formed in components of controlled systems. While EUC must be in a state to cause damages. For example a safety arm of robot can not endanger a worker in any case (17).

- **Indirect safety**

  Some systems do not represent a danger but they are critically relevant. In this case it is needed to consider an indirect safety. For example systems of emergency alert which can threat humans by a failure (17).

Although this classification is focused on hardware of safety-critical systems (in examples), it can be a safety-relevant (critical) system component. Also a simple access to database system can be safety relevant with data influencing on safety-critical functions (17).

2.2 Safety Critical Computer Systems

Safety critical computer systems present a connection between a controlling computer system and a complete system (17). These parts must be integrated together, which is shown in the scheme in ***Fig. 5***.

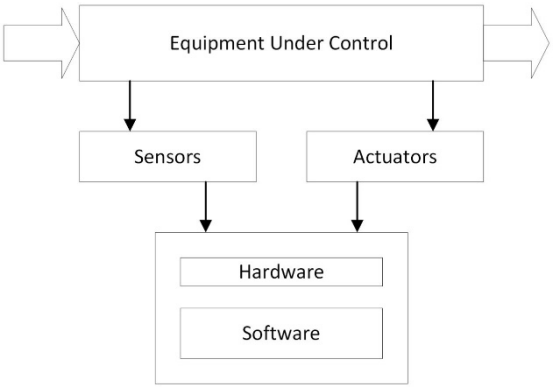

Fig. 5 *Structure of control and safety system*

Information on a controlled process or its environment (called as EUC) is collected with various sensors. Gained data are analysed with concrete software and new orders for EUC are created on their base. These are transferred with actuators (17).

Safety-critical computer systems can be divided into two groups according to their function. The first are control systems which serve especially to control other system elements. They can be classified as safety-critical in a case when a controlled process presents a danger in certain conditions. The second group are safety systems which are implemented on processes (or systems) which are safety-critical in their base. They serve to control other elements of system (process) or its environment. In case of data change gained from sensors the system signalises this fact on a control place or some measurement is processed (17).

Reasons for implementing of computer systems in safety-critical applications are similar as reasons by traditional applications. There are for example a high performance (for example by data scanning and calculations), compactability and flexibility (simple adaptability of properties) (17).

Their disadvantages in this sphere are less obvious. The main problem is an enormous complexity of hardware and software while the main aim by a design of safety-critical hardware and software is to decline the complexity to the lowest possible rate. Software is complex in its base and it is not possible to test all cases (17).

2.3 Systems of Real Time

A system of real time is a system with specific aspects including logical and time requirements on accuracy. System operates logically correctly when we gain correct results. System operates correctly from time perspective if we reach results in a correct time. Not only is the specification of time requirements important by systems of real time but also verifying of time correctness. Systems of real time can be characterised as (18):

- Controlled by events
- Their failure is expensive and dangerous
- Parallel/multi-wire programming
- Permanent operation without operator activity
- Requirements for reliability and resistance against failures

Systems of real time should have guaranteed time accuracy (feedback) also in the case when some events require access to the same system resources. System resources can be classified into the following classes 18:

- **Adequate**

  Nearly every methodology for system design can be used to fulfil time requirements of application (18).

- **Inadequate**

  Application has such requirements which are out of limits of possible technologies. No methodology can be used to fulfil time requirements of application (18).

- **Adequate but rare**

  It is possible to fulfil time requirements of application but only with using of precise methods of source allocation. From perspective of real time the most interesting applications are located in this area (18).

It is important to say that nearly every control system is a system of real time. The complexity of the system (number of sensors, action members) tells if it has a sense to consider logical as well as time requirements (18).

Application of systems of real time can be classified as (18):

- **Clearly cyclic**

  Every task is started periodically with stable requirements for system tools. They are for example a digital regulator, control of flight, monitoring in a real time (18).

- **Prevailing cyclic**

  The most tasks are started periodically but a system reacts also on external asynchronous events. These are for example modern

avionic and control systems (18).

- **Asynchronous and predictable**

  Most tasks are not periodic. Requirements for system tools can differ during the sequence of initiations but they have known limits or static distribution. They are for example a processing of signals and observing of objects (18).

- **Asynchronous and non-predictable**

  They react prevailing to asynchronous events. Solved tasks have a high complexity. Typical examples are intelligent control in a real time, simulation in real time and virtual reality (18).

2.3.1 Time limits

To define time limits it is needed to specify two elementary terms: task and job. A task is a sequence of code formed by an amount of jobs. These are started to support a certain system function. A job represents an instance of task. Jobs need resources to be operated. Every job is characterised with a set of time parameters, calculations from physical laws describing a controlled process (18):

- Release time

  Time instant when a job is prepared to running

- Deadline

  Time instant till a job must be accomplished. This time is also called as the critical time limit of process.

- Relative term to deadline

  Interval length between release time and deadline.

- Response time

  It is defined as a difference of a real time of accomplishment and a release time.

Regarding to deadline the systems of real time can be divided into (18):

- **Systems with hard real time**

  System of real time with strict deadlines. Hard deadlines are such deadlines which must be unconditionally followed. If not, the system works not correctly and it has often catastrophic consequences. They are important tools to verify that terms will be kept. Typical examples are a nuclear power plant and a plane control (18).

- **System with soft real time**

  These are systems of real time including required deadlines. Required soft deadline can be sometimes changed. To define the word "sometimes," probabilistic description of requirements can be used (for example 99% of terms must be accomplished) or with help of function defining describing the usability of batch according to the deadline, for example phone centrals and multimedia applications (18).

Except of previous deadlines there is also the first firm deadline. A required deadline is such that functionality is close to zero in a moment when deadline is reached (delayed batches are not needed) (18).

2.3.2 Scheduling algorithm

Scheduling algorithm for systems of real time can be classified into (18):

- Static scheduling
- Dynamic scheduling
- Scheduling with a static priority
- Scheduling with dynamic priority

Decisions are processed only in certain selected time instants by a static scheduling (based on hardware timer). Schedule is calculated off-line and it is stored for using during a programme running. This method is possible only when a system is deterministic (18).

Dynamic scheduling algorithms based on priorities are applied by a dynamic scheduling. Every batch is associated with a priority and the batch with the highest priority is running. All batches of one task have the same priority by the scheduling with a static priority. Various batches of the same task can have different priority by the scheduling with a dynamic priority. The term of the dynamic priority is based on the method used by an implementation of this scheduling (18).

2.4 Industrial Networks and their Safety

Designed model of testing is focused on testing of communication subsystem and therefore it is needed to identify elementary problems of industrial networks and their safety. These networks are today a part of wide measurement and control systems. Communication ways in a control

system are one of important and sensitive spots. Communication safety is in standards defined with keeping of (19):

- Confidentiality (only authorised objects/subjects have access to data)
- integrity (data can be modified only with authorised subjects and information origin can be approved)
- accessibility (data are accessible for authorised subjects into certain time, service will not be refused)

To reach safety aim in communication it is recommended to apply safety functions which are processed with suitable selected safety mechanism. These mechanisms can have software, hardware, physical or administrative form (19).

In many cases an industrial network is a part of system participating on a control of safety-critical processes. Undetectable failure of transferred data can cause significant material damage on the equipment environment and therefore the system must be designed so that it fulfils a required SIL (19).

2.4.1 Types of Attacks in Communication

Attack by communication is a creation of safety incident which uses a weak spot of communication system and it causes material damage or damage on health. Attack can be in general classified as intentional, unintentional or accidental. It is possible to attack hardware components of system in general in communication system (natural catastrophes, attacks caused by fire, stealing and others), system software (deleting of software

because of a wrong configuration, failure of archive system, operator failure and others) or data. From perspective of data these attacks can be divided into communication interruption (active attack on availability), monitoring (passive attack on confidence), change (active attack on integrity) and value adding (active attack on integrity and authenticity) (19).

Safety- critical communication subsystem used by a control of safety-critical processes has specific aspects in comparison with standard communication system. Basic differences are (19):

- Processes controlled in a control system of communication subsystem are time-dependent
- Character of data transfer is mostly cyclic
- Regime of transfer with confirming of data is often used
- Transfer system must guarantee the determined SIL of the system

While a time factor (and therefore also time dependence) of these systems depends on a control level and process type. In ***Fig. 6*** there are included time factors characteristic for concrete layers (19).

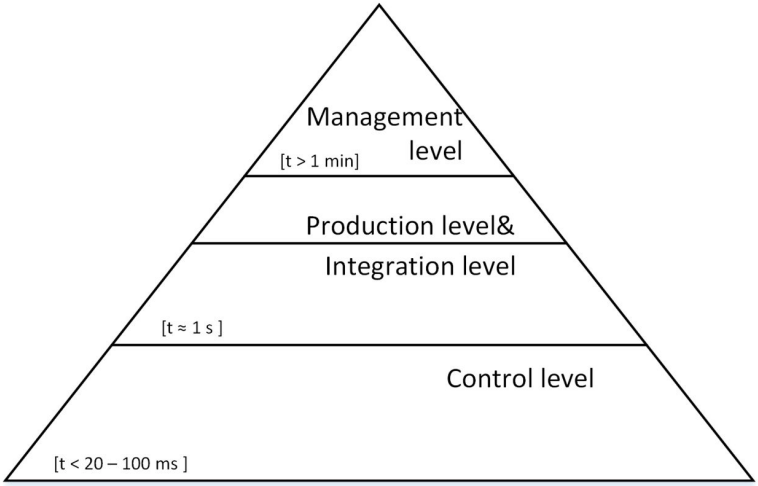

Fig. 6 *Pyramid architecture of production IS and time factors*

From perspective of safety-critical systems it is not important only keeping of accuracy and integrity of transferred data but also their time accuracy (from perspective of a real time).

2.4.2 Requirements for Safety Protection

To limit a risk connected with danger it is needed that safety-critical communication system was in compliance with the standard IEC 61508. It has to contain safety protections. During communication the following requirements for security must be fulfilled (19):

- Communication authenticity
- Communication integrity
- Delivery of users data
- Accuracy of data control

Requirements for safety protections must be included into a specification of requirements for system and its safety. Library of used protections can be little bit modified in a concrete application respecting a risk level determined for every type of attack. Determined SIL of concrete data and process must be considered, too. Every selected safety protection must include a documented proof of safety consisting of its analysis and description of safety reaction in case of failure detection (19).

2.4.3 Application protocols

Industrial networks as well as networks in information systems use a layer model of communication according to the ISO/OSI. To increase a performance and to eliminate delays it was needed to omit extra layers of the model ISO/OSI from protocol set. Omitted layers were network, transport and relation ones which are not directly required on level of industrial bus networks. A link layer has a control of connection in this case. Omitted layers decrease the connection control, transfer of extra data and packets and a delay is lowered to a minimum (19).

CLASSIFICATION OF PROTOCOLS OF INDUSTRIAL NETWORKS

Table 2

Layer ISO/OSI	Profiles
Application	Application protocols (MMS, CAL, CIP, FMS/DP, ...)
Presentation	
Relation	Higher protocols (ISO, TCP/IP, ...)
Transport	
Network	
Link	Protocols of physical and link layer (Fieldbus, industrial Ethernet, TokenBus, ...)
Physical	

Especially specific company buses and industrial networks labelled as Fieldbus networks are applied on the level of physical and link layers (19).

2.4.4 Fieldbus networks

Fieldbus is a term for a group of industrial network protocols used by control in a real time. From 1999 it is normalized in the standard IEC 61158. Term Fieldbus names a digital communication bus between control systems and their attached equipment. Fieldbus operates with more technology types and their existing protocols and it belongs to one of the most used standards in area of industrial networks (19).

TYPES OF ATTACKS AND PROTECTIONS IN NETS OF FIELDBUS TYPE Table 3

Type of attack	Safety protections
Damage of message	Sequence number
Accidental repeating of message	Time stamp
Modification of sequence	Time acceptance
Loss of message	Authentication of connection
Non-acceptable delay of message	Response
Inserting of message	Safety code
Masking of message	Redundancy with cross control
Wrong addressing of message	

The standard IEC 61784-3 defines more types of attacks on transferred reports in industrial networks Fieldbus by a transfer through black communication canal (according to the IEC 61508-2). Because of these types of attacks and to secure elementary communication requirements for safety-critical systems there are exactly defined types of safety protections

in industrial networks of Fieldbus type (based on IEC 61784-3) (19). Overview of attack types and particular protections is in the Table 3.

It is necessary to say that the table does not express correlations between types of attacks and their protections (19).

Communication Profile of CIP Safety

Protocol CIP belongs to the most applied protocols of industrial networks especially thanks to the progress of architecture and independence from specific products of concrete producers. CIP presents an open object protocol supporting a wide scale of communication services not only for traditional Fieldbus networks but also for Ethernet networks in industrial networks. It can be integrated into various products of concrete producers in particular environment. Protocol CIP can be characterised as (19):

1. It is conceived as an object, every communication node is modelled with connected objects.
2. It provides a wide scale of communication services and it supports specific requirements of industrial equipment and control applications.
3. It uses a single form to transfer all data.
4. Complete profile set for typical nodes of industrial networks.

Standard of protocol service CIP can be extended into area of safety-critical communication with communication profile CIP Safety. This extension enables to transfer safety-critical as well as non-critical data in one medium and it helps to form safety-critical connection between two or more applications (19).

CIP Safety is the protocol CIP with a wider model of application layer and user profiles of equipment. Adding of the CIP Safety services to these layers increases safety to standard applications from SIL 0 to SIL 3 (19).

General communication failures for Fieldbus technology were used by a design of safety measurements of communication protocol CIP Safety. In Table 4 there are definitions of safety measurements which can eliminate an effect of expected communication failures (19).

COMMUNICATION FAILURES AND CONCRETE MEASUREMENTS IN THE CIP SAFETY Table 4

Communication errors	Safety measurements				
	Time stamp	Identification of sender and receiver	Safety CRC code	Redundancy with a cross control	Others
Repeating	X				
Lost	X				
Input	X	X			
Change of sequence	X				
Damage			X	X	
Delay	X				
Connecting of SC/SC data		X			
Connecting of SC/NSC data	X	X	X	X	X
Errors of connection	X				

Although the safety CRC code provides an additional protection for communication errors in data, it was not added to errors in the table (19).

Communication Profile ProfiSafe

Safety profile ProfiSafe for a bus network Profibus (one of the Fieldbus protocols) comes from general requirements for safety-critical communication according to the standard IEC 61508. It was designed for safety-critical communication between safety-critical peripherals and controllers. Most equipment is constructed on a base of computer so the profile ProfiSafe was implemented in software (19).

Safety measurements are one of important parts of the ProfiSafe profile. They eliminate errors in black communication channel (according to the IEC 61508-2) and other types of general communication errors based on the Fieldbus technology (19). Particular safety measurements classified as communication errors are in the Table 5.

COMMUNICATION ERRORS AND PARTICULAR MEASUREMENTS OF PROFISAFE Table 5

Communication errors	**Safety Measurements**			
	Sequence number	Time lapse with confirmation	Authentication of transmitter and receiver	Control of data integrity
Repeating	X			
Deleting	X	X		
Inserting	X	X	X	
Change of sequence	X			
Damage				X
Delay		X		
Masking		X	X	X
Errors of switchers	X			

Except for these safety measurements, the producer of Profinet equipment defines so-called F parameters which are not transferred and they are formed during parameterization and configuration of equipment before a concrete data transfer. These are for example the address of source and target, Watchdog timer monitoring and parameter definition for message (19).

3. DEFINITION OF DIFFERENCES AND SPECIFIC ASPECTS OF TRADITIONAL SOFTWARE SYSTEMS AND SAFETY-CRITICAL SYSTEMS

Regarding the fact that the most testing programmes are formulated for testing of standard systems, it is necessary to define differences from the perspective of testing between traditional and safety-critical systems. There is an amount of standards for an area of safety-critical systems as well as the standards for standard systems which cannot be described in detail. Therefore specific standards for the following analysis were identified on the basis of the previous steps.

We have focused on the standard IEEE 829 which specifies types of documents for concrete phases of software testing. The results of analysis can be used by our proposal for a testing model regarding the fact that the difference between a concrete testing process of standard and safety-critical systems is not so big.

The next norm of our attention is IEC 61508 which presents a base for all safety-critical systems. It is standard which must be followed by every safety-critical system and requirements on testing must be identical for every safety-critical system.

Requirements from the previous standard for safety-critical systems are further described in the next standards and guidelines where specification of concrete areas is added. We have focused on the standards IEC 50128, IEC 61511, CAP 670, DO-178B, IEC 62061 and the guideline MISRA, which present the main standards of concrete areas. These standards and guidelines enable a simpler and more efficient implementation of safety requirements for concrete areas.

The next group of standards were the standard for nuclear power plants. These require the highest level of security and therefore their specification of requirements should be on higher and more specified level. We have chosen the standards IEC 60880, IEC 61513, IEC 62138, IEC 60987 and IEC 61226 which represent the standards for area of development proposal of safety-critical systems of nuclear power stations.

Although all analysed standards for safety-critical systems contain requirements for testing they are defined in a certain abstract level. Therefore we have analysed the guideline GAMP dealing with testing of automated systems. This guideline is focused on testing in specific sphere (food and pharmaceutical industry) so its requirements for testing are appropriate also for safety-critical systems in general.

Knowledge gained from analysis must be generalized for an application in our proposal of model. It is especially because standards and guidelines for safety-critical systems deal especially with an integration testing. System testing is mentioned only as one test type which must be made. Therefore requirements of testing must be generalized so they can be applied by any testing type.

Generalized knowledge from analysis of this standard and guideline will present a base for a proposed testing model. They represent requirements for testing of safety-critical systems which must be implemented with our model.

3.1 Analysis of the Standard IEEE 829

Some documents types were created for a testing control during years. The documents are for testing of all types of software, from a testing of

components to a final testing before delivery to customer. Every organisation develops these documents individually and with different titles which do not fulfil main purpose in some cases. The standard IEEE 829 for a documentation of all types of software testing including the user acceptance testing has been created to provide a complex set of standardized documents IEEE (20).

The standard IEEE 829 contains eight types of documents which can be used in all phases of software testing. This standard covers in general all types of testing. It enables to use documents for any situation.

From perspective of software documentation the complete content of this standard is important. Therefore we focus on concrete document types by our analysis.

3.1.1 Summary of Results

The studied standard IEEE 829 (20, 21), specifies various types of documents which can be used by testing of software. The task of concrete documents is clearly specified, requirements and a connection between documents are included. Specified documents present various process phases of software testing.

Because they focus on the software testing in general it is possible to use it also for documentation of software testing process of safety-critical systems. Or in a context of a system testing which we study. Particular phases and their documents will present a base for our proposal of testing model.

3.2 Analysis of the Standard IEC 61508

The standard IEC 61508 also named as IEC/EN 61508 is focused on a functional safety of electrical, electronic and programmable electro-technical safety-critical systems. It includes also requirements from the American standard ISA-S84.01 from 1996 and the German standard DIN 19250 from 1994.

It is named as a basic standard for all safety-critical systems. The standard completely covers a safety life cycle, however some requirements have more advantages and they can be interpreted better in particular sphere standards.

The standard has seven parts, considering a testing as well as safety-critical systems the parts 1, 2 and 3 containing requirements for these systems are the most important. Requirements are defined for particular phases of concrete life cycle. The structure is not modified neither by our analysis.

Other parts deal with instructions and examples of these systems and they have an information character from perspective of testing.

3.2.1 Summary of Results

Analysed parts of the standards (22, 23, 24) are focused on a complete process of design and development of hardware and software of safety-critical systems. Safety requirements are defined which must be followed in particular phases of safety life cycle. In the case of any safety-critical

system it is needed to respect requirements and recommendations of this standard.

Requirements for testing are focused prevailing on an integration testing. Other test types are also mentioned, mostly only as test types which must be processed. However, the analysed parts of the standard contain also requirements which must be implemented into a process of testing. There can be included for example requirements for a development cycle, data, data communication, verification, validation and integration tests.

Because this standard is a base for all standards and guidelines of safety-critical systems, it is required to implement gained requirements into our proposal of testing model. Identified requirements must be generalized; they are not focused directly on system testing. These generalized requirements will present elementary requirements which must be followed in our model.

3.3 Analysis of Regional and Product Standards of Safety-critical Systems

The analysis of regional and product standards is formed on base of specific standards which will cover a significant part of specified safety-critical systems. We have focused on the main standards which are enlarged and completed by requirements of the standard IEC 61508.

These so called product or regional standards enables a better specification of problematic areas of concrete sector. The following pages include brief characteristics of standards together with their commentaries. We have focused on the main standards of concrete sectors.

BS EN 50128: Railway Applications – Communication, signalling and processing systems. Software for Railway Control and Protection Systems

BS EN 50128 is the European standard which defines technical and processing requirements for software development of the PE systems for using in a control of railway transport and in protection applications. It helps to secure a safety and a correct proposal of control of railway and protection software (25).

They are orientated on a complete process of specification, design, implementation and approval of software for railway control and protection systems. Except of these they contain requirements for assessment, maintenance and security of software quality. They do not forget the integration of proposed software with concrete hardware.

IEC 61511: Functional safety – Safety control systems of continuous technological processes

IEC 61511 is the specific standard focused on final users in manufacturing industry. This standard provides the best safety process which should be applied by all users by implementation of modern SIS (26).

It focuses on the system, hardware and software requirements for area of technological processes. Except of these it contains also an instruction for application of these requirements and defining of required level of a complex safety.

CAP 670: Air Traffic Services Safety Requirements

CAP670 is the standard which defines a regulation frame and requirements connected with providing of air traffic services (27).

This standard describes a method and process of approval by a provider of air services. These requirements are focused on fulfilment of international agreements and standards to reach a safety of air operation. It includes also instructions and methods which can be used to gain an agreement with this standard.

DO-178B: Software Considerations in Airborne Systems and Equipment Certification

The guideline DO-178B deals with a software development of board systems in air industry. Its creation is a result of cooperation of more organisations dealing with this sphere. The task of this guideline is to provide instructions for software production of board equipment. It simplifies a certification of this software in area of air industry (28).

This document reflects on the amount of certification requirements, as well as on a recommended process of software approval of board systems. Its tasks depends on a method how a software failure can influence a system safety.

IEC 62061: Safety of Machinery – Functional Safety of Safety-related Electrical, Electronic and Programmable Electronic Control Systems

The standard IEC 62061 defines requirements and it introduces recommendation for design, integration and validation of the E/E/PE control systems with connection to safety for machinery (29). Content of this standard is determined by a structure of the standard IEC 61508 and therefore it deals with parallel requirements. These requirements are related to levels of safety integrity 1 to 3.

MISRA: Development Guidelines for Vehicle Based Software

The association MISRA was created as a cooperation between producers, suppliers and designers of vehicles for development support of safety electronic systems in vehicles and other closed systems. This association has issued more guidelines specialised on car industry (30).

This guideline can be labelled as the main one. Its task is to provide a support by a design and a formation of safety software in vehicles. This guideline is supported by more reports containing detailed information and recommendations for this process.

3.3.1 Summary of Results

Analysed standards and guidelines contain information and requirements which simplify their application into concrete spheres of safety-critical systems. Guidelines enable a better and more efficient implementation of general processes into concrete specific areas in a case of proposal of safety-critical systems of a concrete area.

Analysis of these standards and guidelines was followed to gain an overview on specific implementation areas of safety-critical systems.

Each of these standards and guidelines is specified in different and strong specific area. While each has an amount of standards dealing with the concrete area into details. To design a testing model on this level it would be needed to know specific aspects of a concrete area. Some specific knowledge cannot be reached without a detailed specification of concrete system.

Therefore the results of analysis will be used by considering of specific aspects which must be processed in various phases for example by specification of particular safety requirements, identification of testing steps and definition of differences. The standards provide also the real areas of testing implementation of safety-critical systems.

3.4 Analysis of the Standards for Nuclear Power Equipment

Specific group of industrial and regional standards are the standards for nuclear power devices. It means that it is also for computer systems which should be implemented in area of nuclear power devices. They present one of the highest safety levels with requirements which must be followed by particular safety-critical systems.

It is required to consider the amount of standards by a design, project, installing and implementation into a process of safety-critical systems of nuclear power devices on a software base (31).

We have studied the standards which are the most used in this area by analysis. We have selected especially standards which are focused on a complete system, hardware, software and requirements for particular system categories.

The following pages describe the brief characteristics of the most used standards in area of nuclear power devices.

IEC 60880: Software for Computers in the Safety Systems of Nuclear Power Stations

The standard IEC 60880 contains requirements for software of safety-critical systems based on computer used in nuclear power stations with

safety functions of the A category. It focuses on the strictest safety functions. It is connected with every phase of software development and documentation including a specification of requirements, design, implementation, V&V and operation (31).

IEC 61513: Nuclear Power Plants – Instrumentation and Control for Systems important to Safety – General requirements for systems

This standard includes general requirements for measurement and regulation systems which are important for safety. It includes requirements for a total safety cycle of safety-critical systems, safety life cycle, integration and operation of these systems (31).

IEC 62138: Nuclear Power Plants – Instrumentation and Control important for Safety – Software aspects for Computer-based Systems performing Category B or C functions

The standard provides requirements for the software of safety-critical systems which are based on computers and they process safety functions of the B and C category. It focuses on less strict requirements for safety functions. It deals with all aspects of requirements for software of safety-critical systems (31).

IEC 60987: Programmed digital computers important for nuclear power stations

This standard deals with computer systems important for safety of nuclear power plants. It focuses on hardware of computer systems of the 1st and 2nd classes (these are defined in the IEC 61513) in nuclear power plants (31).

IEC 61226: Nuclear Power Plants – Instrumentation and Control Systems important for Safety - Classification

This standard defines methods of information classification and control functions for nuclear power plants, concrete safety-critical systems and equipment. These functions are classified into categories which determine an importance of safety functions. It includes criteria for classification of safety-critical systems for the A, B and C categories (31).

3.4.1 Summary of Results

Nuclear power facility represents one of safety-critical systems types with the highest safety criteria. Regarding to possible risks these requirements are more than authorized.

Specific standards and guidelines (32, 33, 34, 35, 36) for nuclear power facilities cover completely all phases needed for specification, design and implementation. They include also requirements for particular supporting activities as control of modification, categorization, verification and others. They define exactly measurements which can prevent some risks.

Analysis of standards and guidelines of nuclear power stations provided us requirements for software and data communication, an overview of specific processes to secure safety, a relation between requirements and concrete steps for their securing and specified verification methods and validation of these specific systems. All these requirements represent a huge amount of information which cannot be processed in the range of the monograph and neither to transfer it into testing model.

Particular phases need an amount of knowledge from specification of a concrete system.

Information gained with this analysis help to achieve closer requirement specifications for testing of safety-critical systems in general.

3.5 Analysis of Guideline GAMP

The guideline GAMP presents a recognized instruction for validation of automated systems including a testing of software products prevailing in area of food and pharmaceutical industry. It does not talk about concrete techniques and activities of standard testing but it is focused on requirements which must be followed by a testing of safety-critical systems (37).

Analysed guideline GAMP4 consists of four parts: the main part, management, development and operation. From perspective of testing the guideline D6 has the highest importance in a part of development. It deals with the testing process in general and it does not consider specific aspects for selected test types.

3.5.1 Summary of Results

Guideline GAMP (37, 38) does not discuss testing in general in difference with standards and guidelines for safety-critical systems, but it concentrates on concrete steps which must be processed by testing. Testing process and concrete testing activities of this standard are very close to the testing of standard systems.

It deals with the testing in area of food and pharmaceutical industry and so requirements for testing can be applied also in general. It can serve as a basis for any type of testing of safety-critical systems and also for system testing.

3.6 Generalization of requirements for testing of safety-critical systems on the basis of analysis of standards and guidelines

Considering an extension of requirements for safety-critical systems with analysed standards and guidelines, it is necessary to simplify and generalize them for the next classification into a model of system testing. Because the system testing is not a known term from perspective of standards we have focused especially on requirements relating to a general testing or similar test types.

Because the testing is not proceeded independently it is needed to formulate briefly activities which are by all phases and requirements groups and which are specified for a system. Every phase in any safety life cycle has also supporting activities except of the main activities. These are for example modification processing, where all requirements for modifications must be controlled and explained, also in case of refusal. The next important activity is a documentation which must be processed during the whole life cycle. This must contain information important for activities in concrete phases, as well as their outputs needed for the next activities and control of completing of tasks and requirements. Concrete phases of the safety life cycle cannot be considered as independent activities but as a complex set of more activity groups which must be processed.

The main groups of requirements are made in the first phases of a complete safety life cycle. The first ones are the safety requirements which are based on safety functions defined on a base of the previous analysis ECU and defined concept. The other requirements are defined after classification of safety functions to their E/E/PE systems (also computer systems). Every function must have certain requirements of safety integrity after classification. The requirements are a base to define SIL. We cannot forget the safety requirements relating to the E/E/PE systems. They contain requirements for particular subsystems and elements of safety systems as well as relations between hardware and software. Therefore also safety requirements of software relate to these requirements. They must be defined for every safety E/E/PE system because they are necessary for their implementation. The important factor for testing is also the architecture of software which describes properties and connection of all elements and subsystems.

The ASIC development life cycle is recommended for a design and development of safety critical systems. This life cycle is identical with the standard development V-model which is recommended with development life cycle also for software. This development cycle is recommended also by the guideline GAMP.

A high importance is put on the independence of particular elements by the development of safety critical systems. Integration testing is using the biggest space in standards and guidelines from perspective of testing. Because these tests are processed before system tests, their generalization can provide an output for our designed model of system testing. Created integration test should contain:

- Testing cases (including of their expected results) and data
- Testing environment (tools, support software, description of configuration)
- Testing criteria which will be considered

The integration testing does not use the testing of all input combinations (black-box testing) but their reduction with an appropriate analysis is recommended. Logical result documentation, fulfilment of aims and criteria, documented failures and the following corrections are the most important aspects. Attention must be paid to a detailed process description of corrections by unsuccessful tests. It is necessary to verify again elements or subsystems which were influences by a modification during any other corrections.

Specific corrections are given to configuration data determining a system running. Their parameters must be in an allowed range and combinations. Therefore it is needed to secure a data consistency, a validity of values and to avoid of an overflow of data source. An important requirement from a perspective of system testing is a required determination of a maximal load and a time of running.

To secure data correctness is necessary for a following of requirements. Standards and guidelines are focused on both perspectives while particular methods cannot be logically divided (for example a division of concrete levels of data communication is important because of transferred data but also from perspective of transfer in a real time).

Testing can be considered as a tool used by a design and implementation of safety systems, therefore they should follow requirements for their support tools. The basic requirement is cooperation

(integration) with tools from the previous and following phases. They can be classified as software support tools of the T2 category because of their general classification. Requirements are in the level of need of their specification and documentation.

Requirements for the testing of safety-critical systems can be completed with the requirements for concrete testing steps with an application of the GAMP guideline. This guideline deals with a testing in food and pharmaceutical industry therefore it can be generalized and used by testing of all safety-critical systems.

Explicitness and documentation of the testing process is emphasized. Process of testing must be documented, they must be evaluated and non-tested components must be explained. Except of this the tests must be classified according to the concrete groups and the participating persons must be determined.

Every created test must have concrete input conditions for the testing initialization. These are for example the requirements for hardware, software configuration, calibration, testing software and testing data.

Clearly defined requirements are for a structure of testing scenario. The testing scenario must be labelled, clearly named and it should contain a connection to the tested requirements. Then it has to include the orders for test as well as prerequisites and activities after test accomplishment. Also acceptance criteria for the test evaluation must be determined.

Not only test results but also a real testing process must be saved for every test. The test must pass or fail. In a case that it failed the test is classified as a negative test and the next process must be found. The test can

be repeated, but it can be also excluded from the testing (one or whole test group) or it can help to modify a system.

4. DESIGN OF TESTING MODEL OF COMMUNICATION SUBSYSTEM BY SAFETY-CRITICAL CONTROL SYSTEMS

Design of the system testing model presents the main aim of the monograph. It is necessary to define an area of possible implementation of designed model before the concrete proposal. If we consider a typical safety-critical system, such as the SKR of a nuclear power plant, we can use our designed testing model in a highlighted area in ***Fig. 7***.

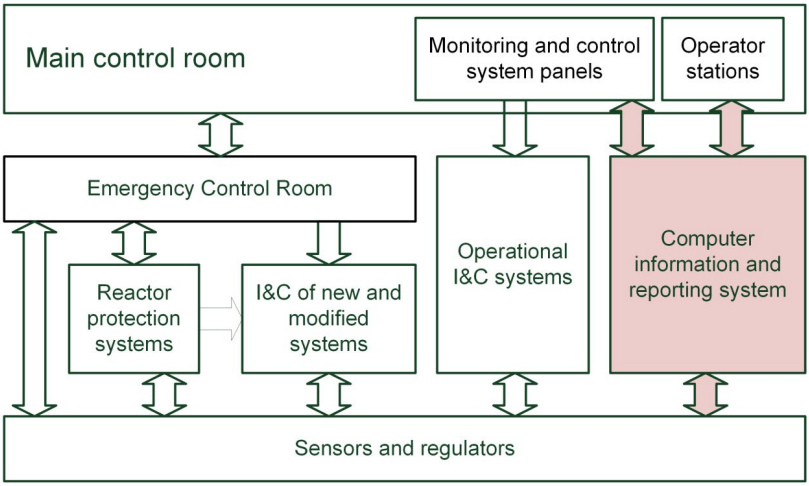

Fig. 7 *Part from a system of control and operation of nuclear power plant*

A concrete model design does not represent an independent activity and therefore it is important to focus on some steps before a concrete model design.

Firstly the meaning of modelling by design and specifications of systems or processes had to be defined. Although the modelling is normally used by software design of computer systems, its application in other areas

is not very common. Modelling can help to show properties and behaviour of nearly every element of real life. The language UML has been applied by a design of our model, this language expresses unambiguously a structure and running of a modelled system or process.

System testing is defined as one of the necessary test types in the standards and guidelines for safety-critical systems. Its integration into a process of design and development of safety-critical systems is still not defined. There is no exact description when a system test must be processed in which the phase of design of safety-critical system. Therefore it was necessary to integrate a system testing into a design and development phase before the concrete proposal of system testing model.

Therefore we have suggested a basic continuity of steps for design and development of safety-critical computer systems. These were completed with concrete system testing and other specific steps for a better sequencing of the proposed process. Based on them we have identified requirements and actions in particular steps which are used in the system testing. The process of design and development shown in this part integrates not only a system testing but it presents also a connection between standard and safety-critical systems.

Because the area of the system testing is very wide it was required to specify closer and to select appropriate types of system testing. The selected step stress and performance testing was chosen because of advantageous properties in connection with analysed requirements of safety-critical systems.

After specification of the previous steps it was possible to design a testing model of communication subsystem of safety-critical control

systems. This implements the requirements defined with standards and guidelines for testing of safety-critical systems, while emphasis is given to requirements of data communication, as well as problematics of real time by data communication. The designed model can be considered as a connection of the testing process and standards for safety-critical systems. Considering a specific area of testing the proposed model operates with a certain rate of abstraction and it is focused especially on important steps and states from the perspective of standard or safety-critical systems.

4.1 Importance of modelling

Modelling can be defined as the creation of a compact (or smaller) system with some identical features with bigger systems. The model plays a similar role as the plans by a building construction. The model secures before a particular processing that the system follows requirements of final users and a design follows required properties. It minimizes complicated and financially demanding modifications of the already created system.

Modelling helps us to work with a certain rate of abstraction. The model can hide details which are not important from its perspective but from the global perspective it focuses on various aspects (39). The decision about what a model reflects has an influence on its understanding or solution planning. We talk mostly about the reflecting of a system from more perspectives and each can reflect different aspects of the modelled system.

The most used language is UML (Unified Modelling Language) which is processed by the group OMG. Although this language is used for

reflecting of a software design, it is possible to use it for reflecting of practically any system or process. The language UML shows a static system structure as well as various aspects of its dynamic behaviour. Every group is defined by some types of diagrams; each of them is focused on representing certain properties or behaviour. Regarding its close connection the statically reflected properties can be completed with a dynamic behaviour and show cooperation of particular system elements. An important aspect is also exactness of the UML language. Each diagram and element has exactly defined properties. It is possible to understand a method of system modelling without analysis or a process reflecting the model.

We have applied some diagrams of the UML language by our designed testing model of communication subsystem of safety-critical systems. We tried to use the diagrams which are appropriate for gaining of a certain perspective while it is still important to describe the model. It provides more exact specification of concrete elements as well as of a context of complete diagram.

4.2 Design Process of Safety-critical Systems

Although our designed model is focused on the testing of safety-critical systems its integration into a process of design and development of safety-critical systems is important. A continuity of steps from (40) illustrated as the UML diagram of activities in the ***Fig. 8*** is a basis for a brief model of process of design and development. This represents an approximate time continuity of particular basic steps by a real design of safety-critical computer system.

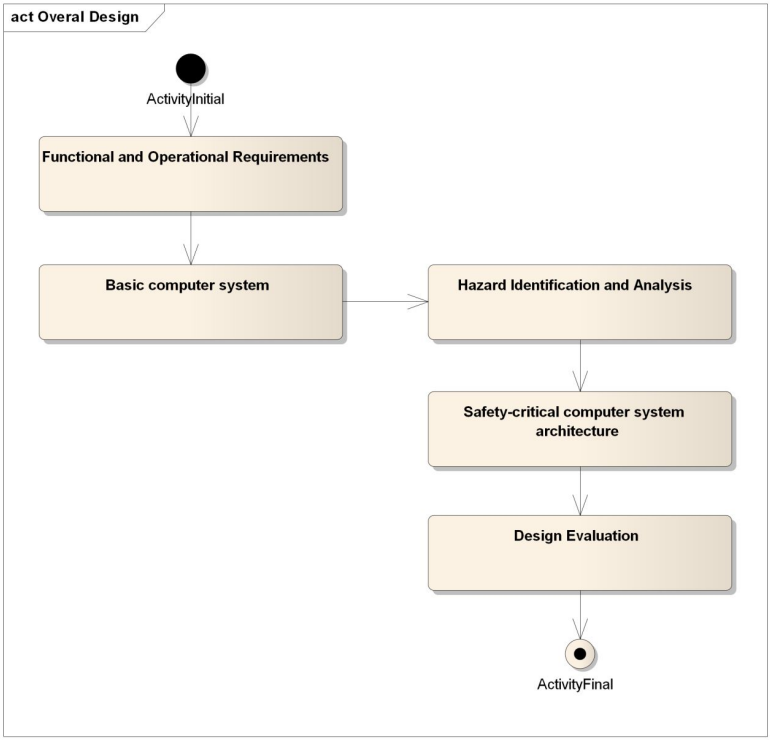

Fig. 8 *Basic steps of design and development of safety-critical systems*

Particular activities of this diagram can be divided into two parts. Steps illustrated on the left show a design of standard computer system for control of required process (or in general EUC) without safety functions. The following steps (shown on the right) are needed to secure a system safety which was proposed in the previous step. This activity continuity presents a connection between a design of standard and safety-critical systems.

4.2.1 System Testing in the Process of Design and Development of Safety-critical Systems

Based on the previous steps of design and development the proposed UML activity diagram was modified and it is shown in ***Fig. 9***. This is based on the previous diagram and it is completed with a concrete system testing and two steps which come from the standards and guidelines of safety-critical systems.

The design is shown as a direct continuity of steps; it is necessary to verify and validate every step. Considering requirements of safety-critical systems it was needed to think also about a correct processing of changes, documentation, planning and other in concrete steps. Therefore it is a very interactive process demonstrated with a certain degree of abstraction. The main reason was complicated illustrations of required support activities for each activity which are not significant from the perspective of integration of system testing into the process of design and development.

Design of a safety-critical system starts with functional and operating requirements on a computer system. These requirements define what the system should do and what method is used for their operation. Only normal system operation is considered and it means that any requirements are not included into safety.

The basic computer system is designed and created in the next steps according to the previous requirements. This is also focused on a normal system operation without implementation of any safety requirements or functions.

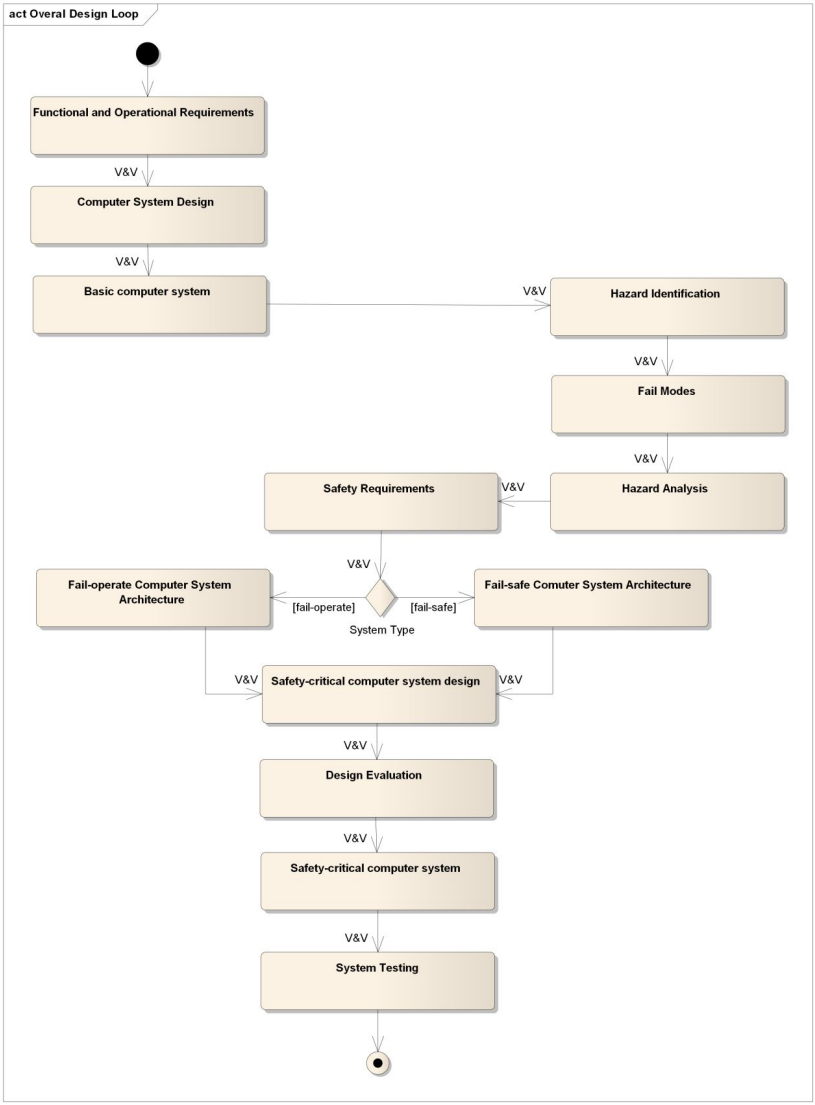

Fig. 9 *Integration of system testing into a process of design and development*

Design of the standard system must be completed with safety properties and function in the next steps. We have to define a basic computer safety system. It is focused on a possible danger (mostly concentrated around a control process) which is a consequence of one or more risks. In the next step possible risks and their consequences are identified. Failure modes are determined on the basis of this identification, where the system or concrete components can occur. This identification is concentrated on hardware, software and systematic failures.

Based on the analysis of identified risks and failure modes the safety requirements for proposed safety-critical computer system are determined. We have to choose between one of two basic architectures from the perspective of a control system. The system is transferred into a non-operating state by an application of the fail-safe architecture therefore there is no risk in case of failure occurring. This system must be able to identify errors and failures or to change the configuration into a safe (non-operating) state. By the application of fail-operate architecture the system must also identify errors and failures; however, it must to continue in operation in case they occur. Concrete component must be isolated by failure and the configuration must be back into an operating state.

The main priority is elimination of all dangers with the design modification by a proposal of any safety-critical system. This modified design can be considered as a design of safety-critical system. In the case that it is not possible to eliminate all dangerous aspects with a design modification the next variant is a modification of architecture of basic system. We focus on a reduction of danger connected with a rest risk.

It is important to evaluate a designed architecture after specification of the design and architecture of safety-critical system. This step must be processed for each concretely designed architecture (not only from perspective of process) if it is considered as suitable after analysis and testing.

After a finding of an appropriate architecture the system testing of the designed system can be processed and it must be done with a completely integrated system. Requirements for testing of a standard computer systems as well as safety-critical computer systems must be considered by the testing.

4.2.2 Requirements of System Testing from Perspective of Safety--critical Systems

It is important to connect particular specific requirements and properties of these systems to integrate system testing into the process of design and development of safety-critical systems. Requirements and specific steps needed for processing of concrete activity were identified for concrete steps of process of design and development (shown in ***Fig. 8***).

Specific steps were displayed as the UML diagrams of application however they were reflected as a unit with the UML interaction diagrams for better illustration and similarity with the previous process. We have focused on a static representation of process of design and development. Therefore this diagram is divided into two parts. The first part (in ***Fig. 10***) reflects a specification of functional and operation requirements, a design of basic computer system and identification and risk analysis.

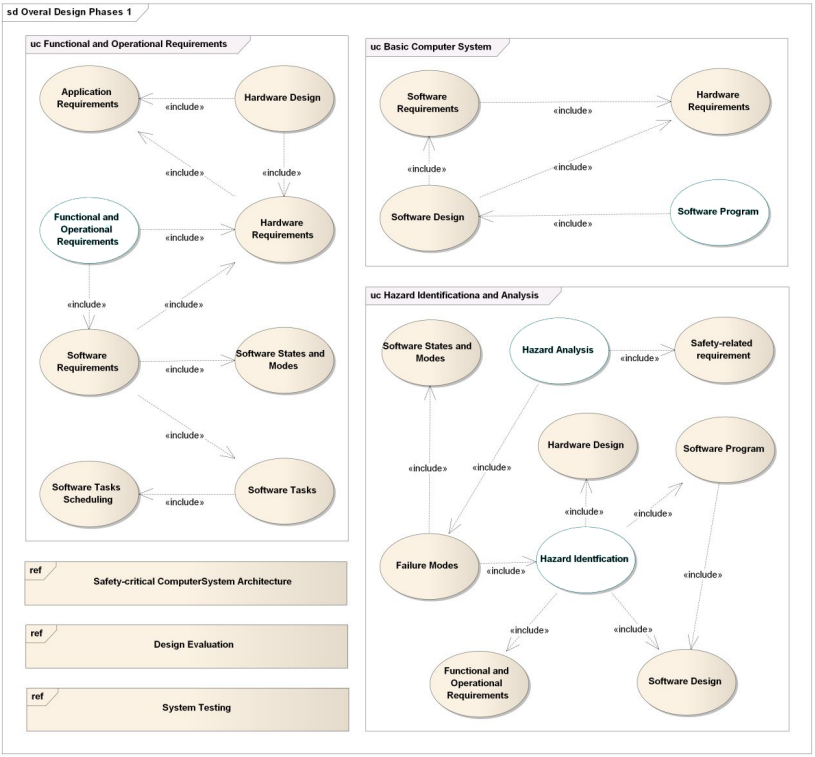

Fig. 10 *Requirements of particular steps of design of safety-critical computer systems – Part 1*

The first step in a design of safety-critical system is specification of functional and operation requirements containing requirements for software as well as hardware. Requirements for a concrete controlled process (or EUC) must be considered, they can present outputs for containing requirements. The software requirements must include also concrete tasks processed by software and which are responsible for various parts of

software function. Distribution from the perspective of running in a real time must be considered, too. Except of definition of particular system tasks the identification of concrete states or system modes is a very important step. These are important from perspective of the next design and testing of safety-critical systems.

Because our design of testing is focused on software testing the existing hardware is considered also in the next step. The basic computer system will therefore represent a software program implementing previous requirements. Particular software programme must come from its design which follows hardware and software requirements. Concrete requirements and specifications must be considered.

All possible risks and their consequences must be identified after the phases which are equivalent with a design of standard systems. We go from functional and operation requirements as well as specification of software and hardware design by the identification. It is important to consider also specific aspects of the designed software programme. Based on these requirements and specific aspects the concrete failure modes of system and its hardware and software components are identified.

Based on the analysis of these failure modes we can specify safety requirements for a proposed safety-critical computer system. These requirements will not be specified into details considering the testing of safety-critical systems. In general we can include here all basic requirements for safety-critical systems by standards and guidelines.

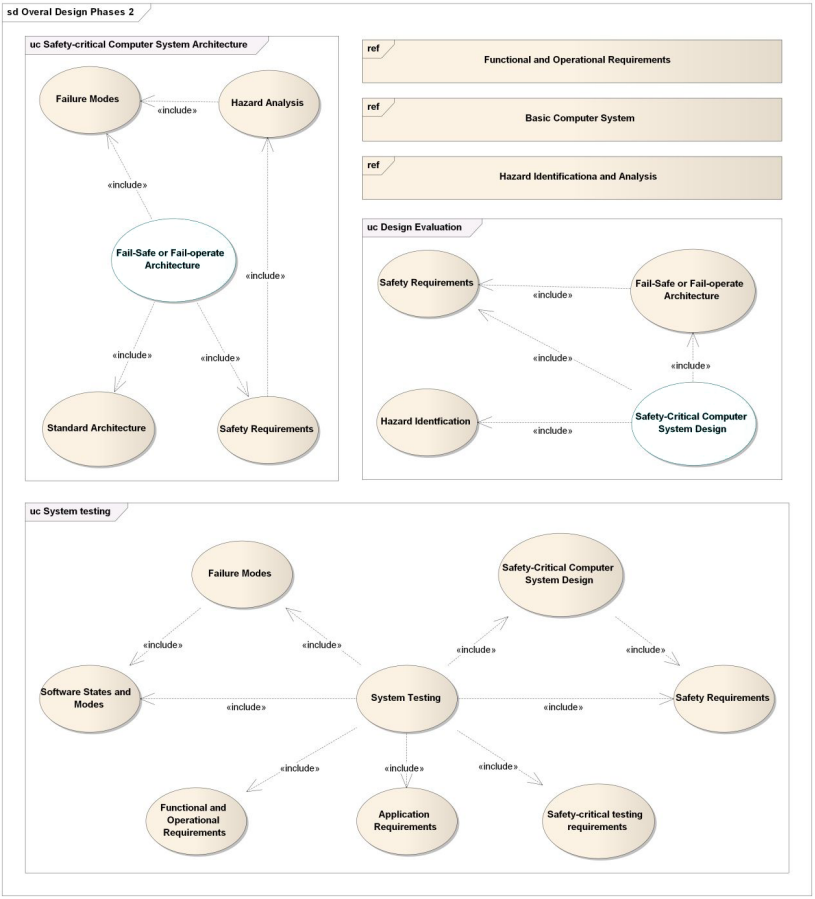

Fig. 11 *Requirements of concrete steps of design of safety-critical computer systems - Part 2*

The second part of the diagram (in ***Fig. 11***) reflects the requirements and specific aspects of an architecture design of safety-critical computer system, an approval of its design and a concrete system testing.

The design of architecture must be based on specifications of safety requirements and identified failure modes. From the perspective of a controlled process the architecture of fail-safe or fail-operate types must be used. Except for these, some standard architectures of concrete computer system can be applied in a practice (for example simplex architecture, simplex architecture with backup, dual architecture with two active channels, triplex architecture ...).

Selected architecture influences the design of a concrete safety-critical computer system. This design must be evaluated and its convenience for planned application must be considered. In this step we have focused more on a concrete design of this system because an area of design evaluation is very specific. There are more methods which can be applied. Particular design of safety-critical computer system must be based on a proposal of basic computer system and it must implement safety requirements. The selected architecture and identified failure modes must be considered.

The processing of a system testing must be done after an evaluation of safety-critical computer system. An integration test must be done before it, which will secure that particular parts work correctly. The system testing proves if an integrated system works correctly as a whole. A concrete testing process must implement requirements for a testing of particular safety-critical systems. A specification of tests is based on:

- **Functional and operational requirements**

  They are important from the perspective of activities and functionalities, a concrete system can operate or provide them and decide which can and had to be tested. They include the requirements for hardware as well as software.

- **Requirements for tested process**

  It does not need to be directly a process, but also one or more "controlled" computer systems in a context of standards and guidelines of safety-critical systems.

- **Safety requirements**

  They define the limits of functional and operational requirements as well as specific aspects for gaining of required safety EUC. Not all system functions must be safety-critical therefore it is necessary to consider both types of requirements (standard and safety ones).

The design specification of concrete safety-critical system must be studied, too. The system provides certain requirements or restrictions considering selected system testing which is not included in the previous requirements. Particular failure modes and states of the tested system must be known by a system testing.

A concrete system testing covers a wide test spectrum. Each of these tests has certain specific aspects which are used by the testing. The previous identification presents a summary of the main requirements which must be applied by the system testing of safety-critical systems.

4.3 Specification of the System Testing Type

The system testing covers a wide area so it is not possible to generalize the process of system testing. We would have to focus on the main activities with a loss of specific aspects of the safety-critical system testing. Therefore it was needed to select concrete test types which are appropriate for the testing of communication subsystem as well as the testing of safety-critical systems in general.

Some types of the system testing are not suitable directly for an area of safety-critical systems or the specific requirements and data are needed to know for an implementation. Although these tests must be also processed, their specification and analysis of particular problematic areas cannot be described in one monograph. We have decided for two test types from all tests offered with the system testing. Each of them has a different perspective on the testing of safety-critical systems.

The first type is the performance testing which is focused on testing performance by a low load. More variants of the performance testing are directly specialised on the testing of data communication and a whole system. Our designed model was focused on the performance testing in general and we did not limit on a concrete variant (for example the testing of network sensitivity). It is because they work on the same principle as our designed model of the performance testing, however they require knowing specific data on a tested system. This condition cannot be fulfilled considering our perspective level on the process testing of safety-critical systems.

The second type is the stress testing which focused on determination of substantiality and reliability of the system behind the limits of normal operation. The stress testing could be considered for a testing concept with steps which can be done with other test types. Particular loads and their range by a test process can present the input parameters of other tests which are a concrete load.

The main problem of classic stress testing is that we do not need to gain any relevant results after a successful test running. Therefore we have decided to apply the stress testing for our designed model which is an

alternative variant of the classic stress test. Its main advantage is guaranteed achievement of test results.

4.4 Outputs for a Model Design of System Testing

The six basic steps of the automation testing present a base of suggested models for system testing. These are illustrated in the ***Fig. 12***, while they reflect the automated software testing in the phase of regressive testing.

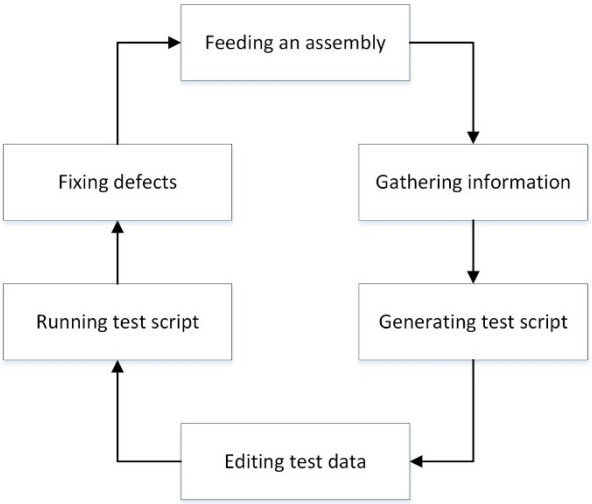

Fig. 12 *Six steps of automated software testing*

They do not represent an iterative process of automated testing but also an iterative continuity of particular steps in general. The regressive testing is focused on identification of failures by modifications of the programme code in the system; therefore these steps are similar to the system testing.

The step of failure repairs can evoke repair of the created failures (for example by the stress testing), but it can be a modification of the tested system or its setting. The testing process of our designed models is proposed on the base of these iterative steps.

Other output for our designed models was the standard IEEE 829 for a software testing. This standard defines eight types of documents which include all phases of software testing from the test planning via its processing to its evaluation. Each type of document/script identifies concretely requirements and steps which must be followed in a concrete phase. It is possible to use them for any test type and therefore it is suitable for our proposed step stress and performance testing. It enables also completion and modification of concrete scripts used in some steps of design.

The total design of testing model is focused especially on steps running with a concrete test. The previous identification and analysis the same as the following modification activities are not displayed in our models. These phases are specific for a concrete testing system while some parameters must be defined for a concrete testing system based on the previous experience. If this area is completed with requirements of safety-critical systems, these phases become more demanding. Therefore these activities are only described.

Because the testing is specialised on safety-critical systems, it was necessary to add specific requirements into the testing process. A concrete testing process displayed in the designed models was completed with specified requirements on testing of safety-critical systems. These were gained with a previous analysis of standards and guidelines. We concentrate

on a communication subsystem of safety-critical control systems; it was necessary to consider the specific aspects and requirements of data communication in these systems.

4.5 A Model Design of the Performance Testing of a Communication Subsystem

The first model design of system testing is specified into the performance testing. This test type is (together with the load testing) one of the most used tests for system testing from perspective of data communication. There are more concrete variants of the performance testing (for example the testing of network sensitivity) which are focused on certain attributes of data communication.

Our designed model deals with the performance testing in general and we have used the appropriate UML diagrams for its display. These show the performance testing of communication subsystem of safety-critical control system from various perspectives.

4.5.1 Process of the Performance Testing of Communication Subsystem of Safety-critical Control Systems

The first part of the designed model of performance testing is identification of particular steps and a proposal of their continuity by the testing. Particular steps of the performance testing for traditional systems were associated to concrete phases of the standard IEE 829. So we have reached an overview about the performance testing sequence for standard systems.

Testing of safety-critical systems does differ from the traditional systems and it has specific requirements for testing. These were identified by the standard and guidelines analysis and then they were generalized. Every requirement was associated to a concrete phase according to the standard IEEE 829 and the actions which must be processed are identified. These actions were implemented into steps of the performance testing for standard systems.

The final model of performance testing for safety-critical systems is displayed as the sequence UML diagram in ***Fig. 13***. This diagram shows steps of the performance testing from perspective of time.

Regarding the number of activities the concrete specific steps were shown in concrete partial diagrams. For greater transparency TestSummaryReport and TestIncidentReport from the standard IEEE 829 were merged into the phase TestExecution in this diagram.

Various activities not shown in this model must be made before the proposed performance testing. These are for example the steps required in a concrete test plan.

It is for example identification and explanation of non-tested system parts or person list contributing on testing from perspective of the testing of safety-critical systems.

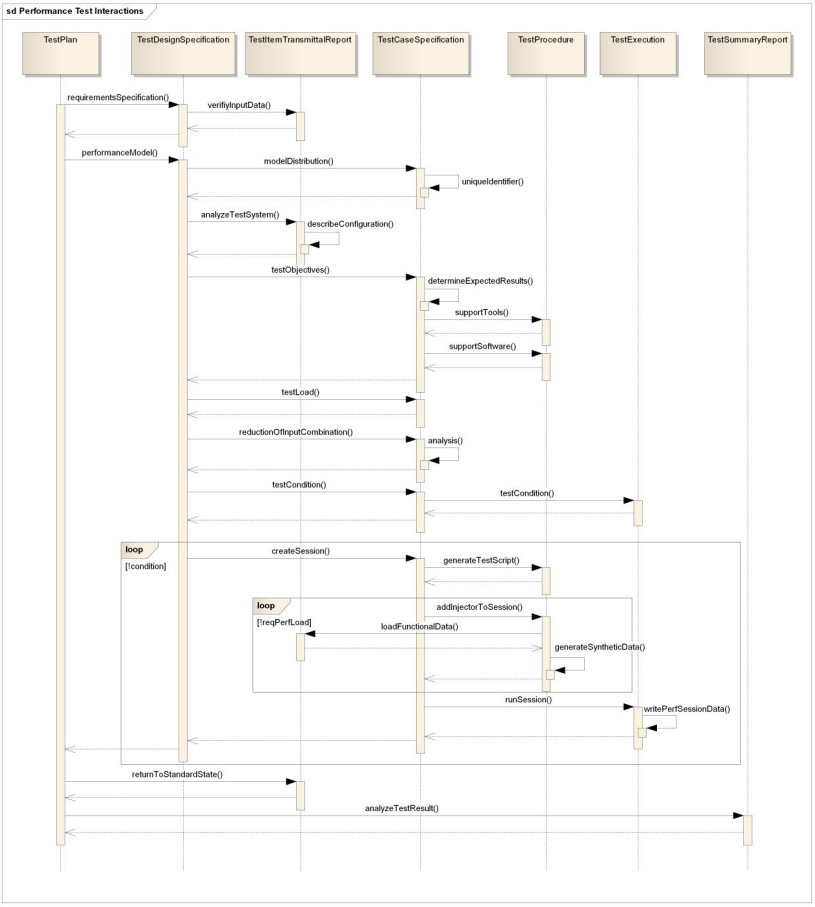

Fig. 13 *Design of the performance testing steps for safety-critical systems*

The first step of design is a system specification and forming of the performance model. The performance model represents various functionalities provided by tested system and a using rate. This model is a base for functionality that can be applied by the system testing. Degree

and details of reached system specification is therefore very important and it has to include the standard as well as the safety system aspects, while, for example, allowed ranges and complex inputs have to be considered because these processes are very wide and complex, these steps were reflected with a certain abstraction.

The next step is analysis of the system which will be tested. The analysis results must be documented for the next using. The gained data are important for a testing process which could be labelled as testing data in a testing context of the safety-critical systems. Important data for a test process were documented as separate steps. A defined identifier of the testing case must be generated before a concrete specification of data.

The testing aims are the first important information which tells what focus on by testing and what is the evaluation of gained results. We can use the estimations on the basis of specification, previous experience or the past iterations of test. We should know also how to specify requires support tools or support software needed by testing.

Then the load must be determined which will be used by testing, the performance testing is concentrated on process times and transactions by a low load. The testing must proceed by common loading of the system in a normal operation. We can make more measurements at once by the load with parallel load test (for example simulation of failure state). It is important not to overstep a maximal system load which is specified in its safety requirements.

The last important steps are a reduction of input combinations for test (based on analysis and safety requirements) and a determination of conditions of test accomplishment.

Then a testing cycle starts. In this part concrete sessions are formed which are made according to the rate of functionality use. In concrete sessions there are formed the testing scripts which represent various user activities including possible alternatives.

Injectors are added to the concrete session in the next step, their number depends on a required load. It is not a total system load but a load from the perspective of performance test. Every created injector must read the real (production) data which are used by simulation operation with a system. In the case that these data are not available or sufficient, it is possible to use also generated data (which must follow the specific aspects).

Later all injectors of an actual session are started and all relevant information needed to a test creation must be documented. This step is documented with a certain rate of abstraction and it will be described into details in a separate diagram.

If one relation is accomplished and the conditions for test finishing are not fulfilled, the next relation is formed and the whole previous process is repeated. If the condition is fulfilled, the testing is finished and the system must get into an original state. This state is renewed on base of system analysis before a concrete test. It can be a restart of the system, change of its mode, a change of configuration parameters or an accomplishment of parallel load test.

The results of concrete relations are analysed and evaluated to provide relevant results considering the defined aims. Provided results should be in a form which is not suitable for the next processing or in a format appropriate for the next support tool. Considering specific requirements of

safety-critical systems for this phase, the concrete steps were closer specified in an independent diagram.

Evaluation Phase of the Performance Test

A big emphasis is given to the test evaluation from perspective of safety-critical systems. The important steps for an evaluation of concrete relations were illustrated in sequence UML diagram shown in the ***Fig. 14***.

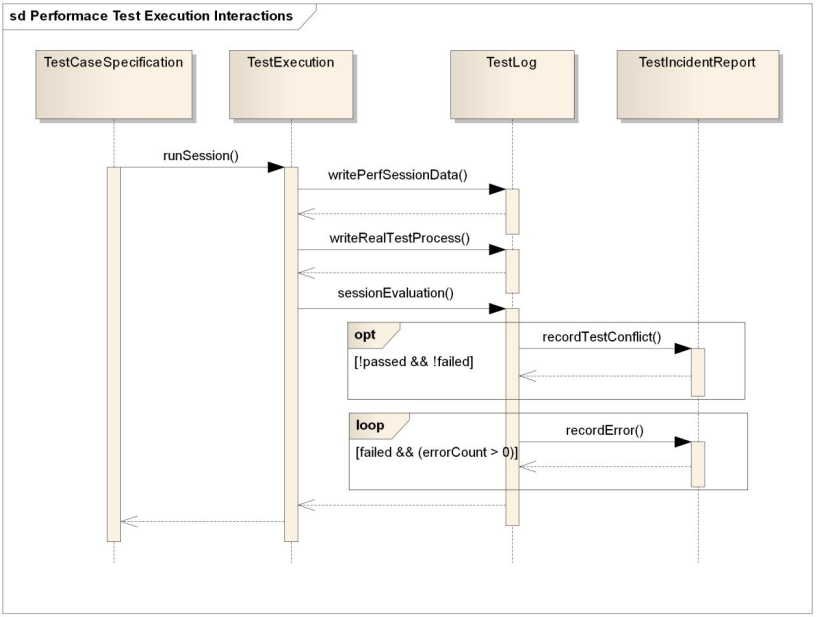

Fig. 14 *Evaluation of Relations of the Performance Testing*

Concrete evaluation of tests and the next analysis must be documented not only from data of an actual relation but also a real testing process. This requirement is emphasized not only in standards of safety-critical systems but also in the GAMP guideline.

Because our designed testing model is focused on a communication subsystem it is needed to document data necessary for testing of data communication.

In the case that the test fails all identifies errors must be documented into one or more reports on an incident. Considering the complexity of this process the designed model operates with only one report about an incident. Reports on incidents should be formed and modified on base of test requirements. It can be a classification based on a subsystem base, failure types, categorisation from perspective of the GAMP and others.

The evaluation of relation results is problematic in specific cases during its process. If we cannot determine if the test passed or failed, the test is labelled as a conflict one and this situation is documented in the report on incident. Such test must be considered by the final evaluation.

We have to say that this model displays only the main requirements for test evaluation from perspective of safety-critical systems. It focuses therefore more on specific support activities than a concrete evaluation process. Considering the specification of tested systems and a big range of possible functionalities, it is not possible to document this process into details.

Phase of Results Analysis of the Performance Test

The next important step is the results analysis of concrete tests. This process was completed with specific steps for testing of the safety-critical systems and illustrated as a sequence UML diagram in ***Fig. 15***.

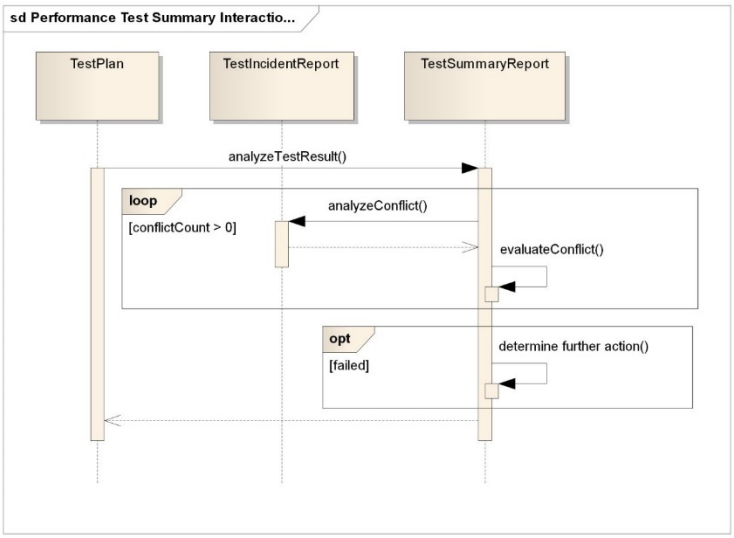

Fig. 15 *Result Analysis of the Performance Testing*

In compliance with specified requirements of the safety-critical systems, each test must pass or fail. Therefore all conflicts must be considered which were created during the process of test evaluation. Because particular data specific for the testing of data communication cannot be evaluated in all cases with a simple method, they do not need to be the conflicts in a real meaning. Analysis and conflicts evaluation can present also the evaluation of information gained by a running of concrete relations of the performance test.

In the case that some test failed by the consideration, it is important to define the next process. This test can be again repeated, a system can be modified (with tools for a control of changes), or the test (or test group) is excluded.

The same as a previous partial model, this one is also focused on the specific support activities (from perspective of the safety-critical systems) as well as a concrete process of evaluation.

Test States from Perspective of the Safety-critical Systems

The concept of test evaluation from the perspective of standards and guidelines for these systems is briefly illustrated as the state UML diagram in ***Fig. 16***.

The diagrams demonstrate concrete states where the test can occur from perspective of evaluation. The tests must be considered immediately after processing and their state is changed on the base of results of this evaluation. Although the evaluation criteria should be clearly defined, the standards and guidelines for safety-critical systems consider the alternative that the test cannot be evidently approved. This alternative can occur if the parameters for a verification of information and time correctness of data communication cannot be defined but clearly considered. In this case we have to analyse gained information after a processing of all relations. At the end of some time period (for example after testing accomplishment, at the end of work shift, etc.) all conflicts must be considered and logically classified (ambiguous tests).

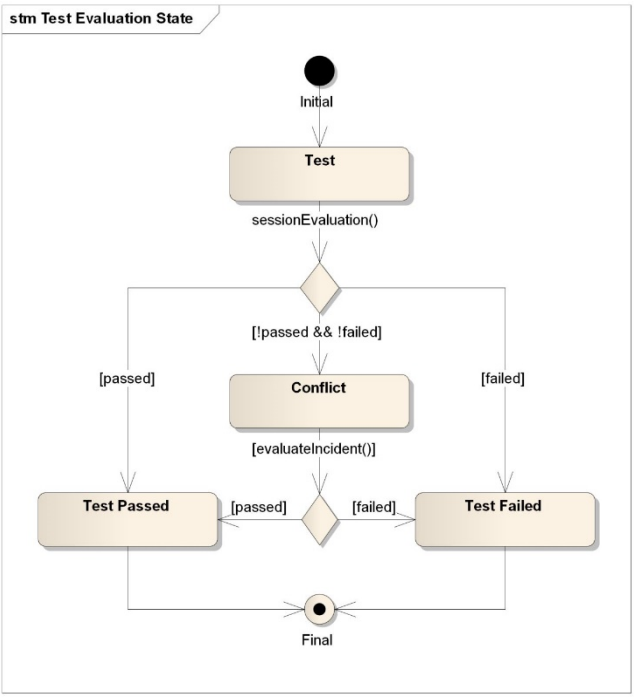

Fig. 16 *Test states from a perspective of standards and guidelines*

From perspective of our designed model of testing we could identify many states where the test occurs during the process testing. This diagram reflects only the test states which are important considering the standards and guidelines for safety-critical systems.

4.5.2 Overview of Performance Testing States of Communication Subsystems of Safety-critical Control Systems

The next view on our designed model of performance testing is an overview of concrete states of performance testing. This overview is

displayed as the state UML diagram in ***Fig. 17*** defining states and events which activate transfers between particular states.

The set of states in this diagram is illustrated as a synchronous sequence considering the complexity and a weak illustration of asynchronous model. The designed model reflects the testing with a certain rate of abstraction, while it illustrates and describes only specific actions and activities of particular states. It means those which are important for the performance testing process or the testing of safety-critical systems.

The group of previous standard states (according to the standard IEEE 829) is displayed as an original state called PreviousStandardStates. This complex state is the first one in our designed testing process. After the gaining of performance model the event GainedPerformanceModel will follow which moves the testing process to a state ProblemDomainSpecification. Definition of the use rate for every possible functionality is the task of input action. In this state it is also necessary to specify all data which are important for a testing process. The specification presents an output action of this state.

The testing process moves to the state SessionCreation by the event TestCondition which is a condition for a test accomplishment. Its input condition has the task to generate unique testing scripts for an actual iteration which will be operated by one or more injectors. Verification of this testing script is an output action. After verification the process testing will be moved to the state InjectorsCreation.

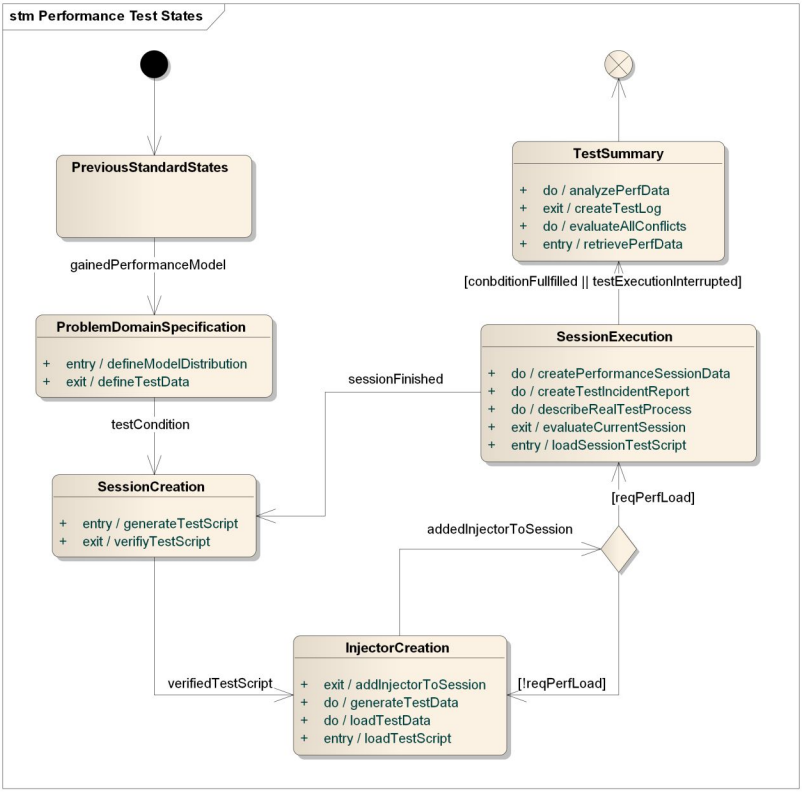

Fig. 17 *Overview of performance testing states for safety-critical systems*

The test scripts which were formed in a previous state are loaded after gaining of this state. Then real (production) testing data are loaded and they represent various actions of injectors. Except of these the synthetic testing data can be generated and they can add randomness into a testing process. Assigning of injector to an actual relation is the output action by a release of this state.

The process is moved to the state SessionExecution by the event AddedInjectorToSession if the required load is gained (from perspective of performance test). This load differs from an actual load of a whole system. If an actual load of performance testing is smaller than it is required by the test, the new injector is formed. Our designed model is shown as a synchronous continuity; particular injectors in this diagram are formed one after another.

After gaining of the state SessionExecution the testing scripts of actual relation are loaded. The main task of this state is a starting of injectors created in an actual session. Saving of data and a real process of particular session is also needed. If there are errors by a running of concrete injectors, we have to create one or more reports on incidents. This process has a certain rate of abstraction as a previous look. In this state we have to record responses on concrete actions and their times which are needed from a perspective of testing of communication subsystem. The task of output action after leaving of this state is an evaluation of results of an actual relation. Therefore we have to process concrete internal activities. These present requirements of the standards and guidelines for safety-critical systems.

If a condition for test accomplishment is not fulfilled by a leaving of this state and the test is not interrupted then a new relation is created. In another case the process comes to the state TestSummary. Reports on performance from all previous iterations are loaded by a process of input action. An important task of this state is also an analysis of all data and evaluation of all conflicts which were formed during processing of concrete sessions. In this case it must be defined if the test passed or failed in

difference to a previous evaluation. This action presents one of the basic requirements for testing of safety-critical systems. The report on test is formed by a leaving of this state considering the test aims. This test report should be in a format which is suitable for the next processing or in a format of required following support tool.

4.6 Design of Stress Testing Model by Testing of the Communication Subsystem

The second model design of system testing is focused on the stress testing. Considering the certain disadvantages of the standard stress testing we have focused on a using of one from accelerated methods. Applied step stress testing is based on the original stress test, but it improves some of its properties (for example guaranteed result gaining).

Its process is identical as by an original stress test but it is completed with an extra life cycle. The stress values are increased in 10% of every 10% of life cycle. This process is repeated till 50% of tested parts failed. These conditions are applied into our designed testing model and they present conditions of a test accomplishment. Therefore this condition does not need to be defined when the complete testing is specified.

4.6.1 Modified Phases of Step Stress Testing

The first step by our design of stress testing model was a proposal of basic steps of the step stress test. This design was displayed as the UML diagram of activities with a certain rate of abstraction in ***Fig. 18***.

This diagram reflects an initialization and concrete iterations of the step stress test while it shows simple as well as complex activities. The

testing process is accomplished when more than 50% of all elements failed during all iterations (except of an introduction fully censored test). The second possibility of accomplishment is in the case when one of the tested elements failed in a fully censored test.

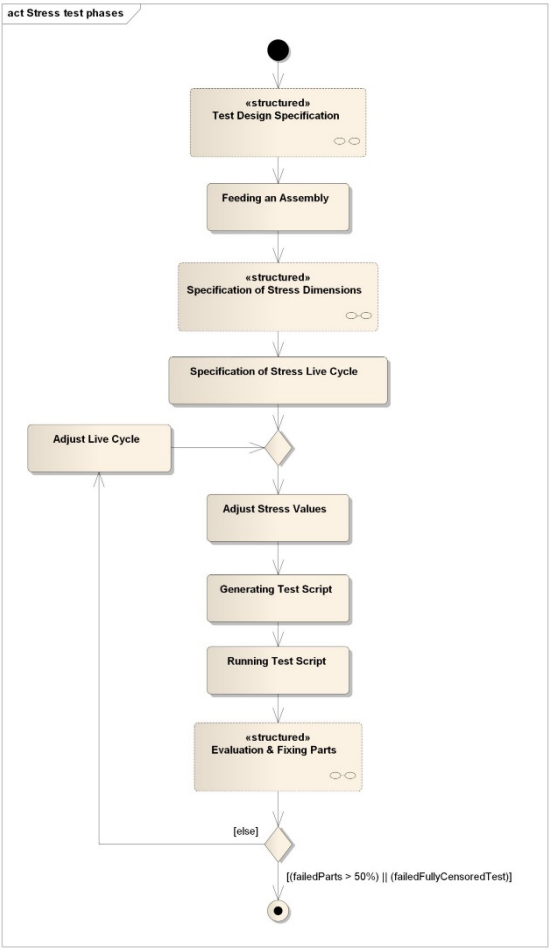

Fig. 18 *Design of modified phases of the step stress testing*

In the proposed diagram we have shown the complex steps which identify important and specific steps for a proposed process of the step stress testing. Particular basic steps of the step stress testing present outputs for the next perspectives on a total process of testing. By a closer look at the next models it is possible to see a concept based on this diagram.

It must be said that this diagram reflects only steps of the step stress testing for standard systems. It does not include our specified requirements for testing of safety-critical systems. These were implemented to other perspectives on the proposed model of the step stress testing.

4.6.2 Process of the Step Stress Testing of a Communication Subsystem of Safety-critical Control Systems

The next step was an illustration of time continuity by the step stress testing. Concrete steps were attributed to the phases of the standard IEEE 829. So we have gained the process of steps of the step stress testing in concrete phases for standard systems.

Specific aspects of the testing of these systems were implemented for a use in area of testing of safety-critical systems. These were reached by analysis of the standards and guidelines for safety-critical systems. Requirements for testing of these specified systems were identified.

Their implementation was processed by the step stress testing for area of safety-critical control systems. It was modelled as the sequence UML diagram, displayed in ***Fig. 19***.

Considering a number of activities some steps were more specifically documented in an independent partial diagram. This method can identify in more simple way the differences between particular types of designed

system tests. Except of that we have joined the phases TestIncidentReport and TestRecord into the phase TestExecution.

The first phase of the TestPlan describes the process of testing and it operates all activities connected with testing. Many steps must be made before the concrete designed testing process and these steps define and specify various data for testing process. From the perspective of safety-critical systems it is needed to identify and explain all non-tested components and save a list of persons contributing by the testing.

The first step which must be processed in a designed model is a gaining of all needed information for testing process. In this phase the basic requirement is a preparation of tested system which must be sufficiently described for using in the next steps. The steps are described in details after their accomplishment; it tells how a concrete testing will be performed. Test script which is created on its base must be perfectly identified. On the base of this information we can specify required support tools and support software tools which will be required by a concrete testing.

Then concrete load parameters must be defined, which must be approved because of tested system and their number must be reduced based on an analysis. The length of life cycle must be also specified and the expected results must be determined. We have to focus not only on the whole test but also on its concrete iterations.

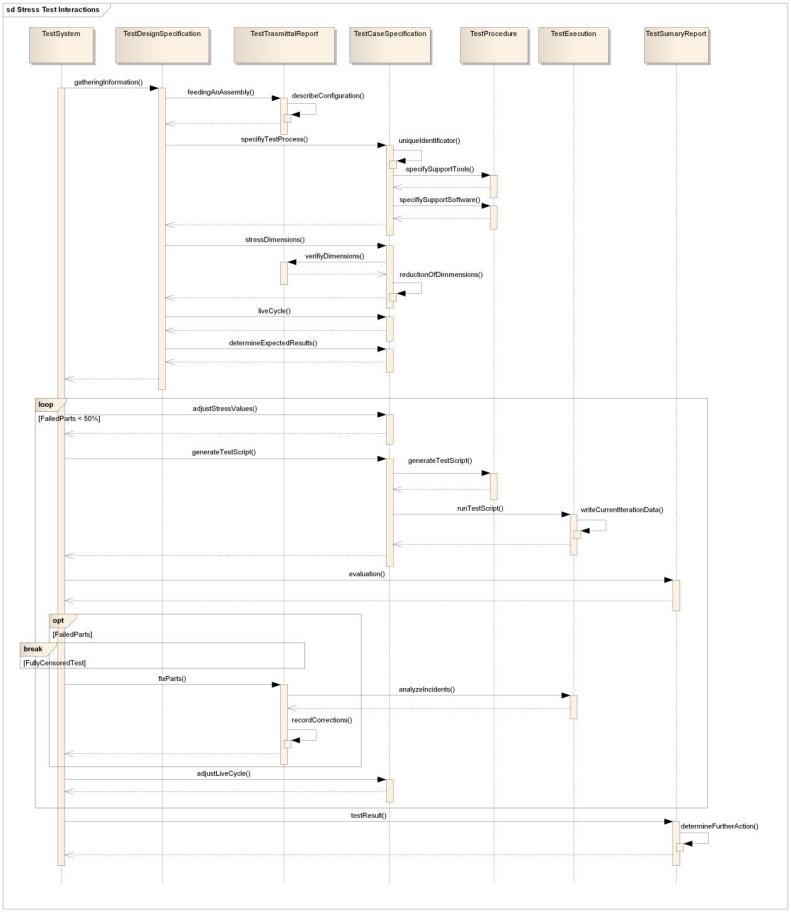

Fig. 19 *Design of steps of the step stress testing for safety-critical systems*

The following testing process start to make concrete testing cycles. The first iteration is fully censored test (it means the standard stress test), when values of the stress parameters adjusted to output values. The next step is a generating of test script and its process. It is important to record

data from an actual iteration. These steps are specific and they were displayed in a diagram. Particular results of an actual iteration were evaluated after finishing. In case of failure of system part it is needed to analyze all reports on incidents and the actual iteration can be evaluated. We have to consider the aspects of data communication.

Considering a character of completely fully censored test there should be no failures in its first iteration. If there is failure the whole process is interrupted and the testing process continues with a complete test evaluation.

Other iterations of the step stress testing are running in identical way. They differ in the length of life cycle which is reduced to 10% of the length of original completely fully test. If there are errors in testing system the parts with failure must be repaired. We follow the record of concrete incidents. Process of repair as well as corrections must be documented for a total test assignment.

The testing system interrupts the testing process when more than 50% of tested parts fail in all iterations. At the end all results must be evaluated and summarized. The next process has to be determined according to the test results. The whole test can be again repeated, there can be a modification of concrete system elements (with tools for control of changes), or an actual test can be excluded (or group of tests). Results from this phase should be in a form which is standardized or suitable for the next support tools

Process Phase of the Step Stress Test

Standards and guidelines for safety-critical systems have specific requirements for processing and documentation of data from the testing process. Therefore the steps which must be displayed in a partial sequence UML diagram in ***Fig. 20***.

During the test the data must be documented as well as a real process of an actual iteration. In case there are the errors during the test, they have to be described in one or more reports on incident. Considering various methods of data creation on incident, this process is generalized. It operates only with one report on incident.

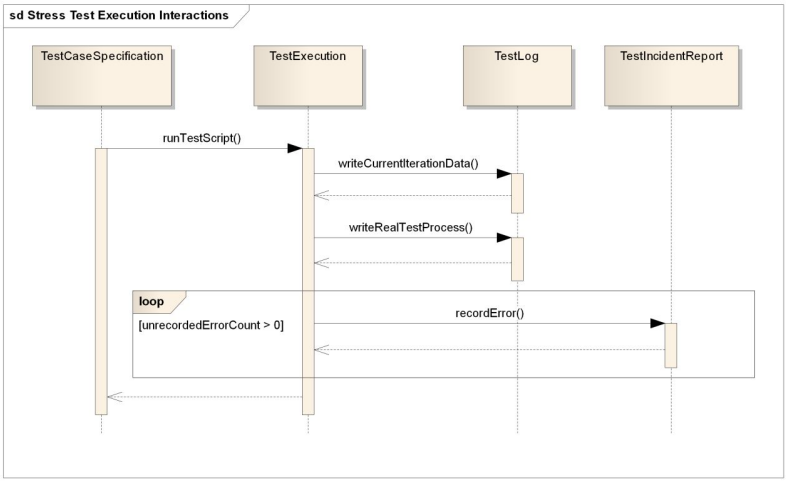

Fig. 20 *Detailed phase of the step stress test*

Step stress testing (as well as stress testing in general) can use other types of system tests for concrete specific loads of system (stress). These must be selected properly to gain required values. Concrete test process can

present a starting of system test group which initiate particular specific system loads. The evaluation is made from a perspective of step stress test. The tests processing a specific system load provide and collect data for this evaluation.

Our suggested model of step stress testing focuses on a testing of communication subsystem of safety-critical systems. Recorded data should contain also data needed for an evaluation of testing from perspective of communication subsystem. This information must be evaluated by tests processing particular specific load considering the concept of step stress test defined in the previous paragraph.

A conflict can occur when we cannot tell if the test passed or failed from perspective of safety-critical systems. Considering the character of step stress testing we should logically know if the system element failed or passed.

We have focused on concrete actions by the specific steps in these phases following requirements for testing of these systems as well as a concrete evaluation. Specification of evaluation method of step stress test of communication subsystem is much less concrete than by the performance testing. Therefore we would have to follow important steps when we had to study a concrete tested system.

4.6.3 State Overview of the Stress Testing of the Communication Subsystem of Safety-critical Control Systems

The next step was to identify states where the step stress test can occur in the designed model of step stress testing. While we identified also events

which activate concrete transfers. This overview of state and particular actions was displayed as the state UML diagram in ***Fig. 21***.

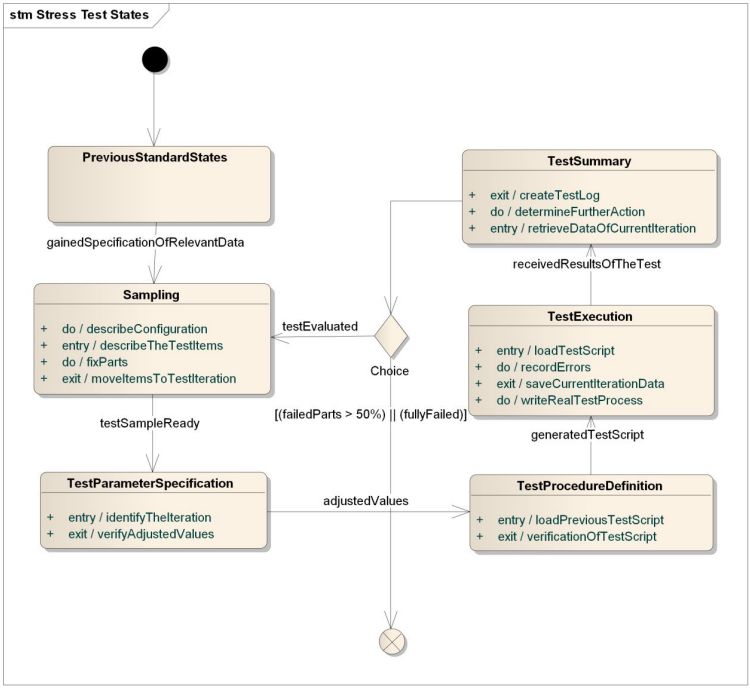

Fig. 21 *Overview of states of the step stress testing for safety-critical systems*

Set of states was illustrated as a synchronous sequence from a reason of better explicitness and illustration comparing to an asynchronous model. We have shown and described only important activities considering the abstraction rate of designed model. We focused on the important actions from view point of the step stress testing and testing process of safety-critical systems.

The set of previous standard states is illustrated as an initial state PreviousStandardStates. This complex state starts as the first in our designed process of step stress testing. After specification of needed data of testing process it moves to the state Sampling with an input action. Its task is to describe the elements of tested system. The configuration of concrete elements as well as their repair in other test iterations is identified in this state. The created sample of tested system elements (and a whole testing process) is transferred into actual test iteration by an alerting of output event.

The process moves to the state TestParameterSpecification by a preparation of testing sample. The input action identified an actual iteration where the testing process is. Its determination is important for an identification of an original fully censored test. Internal actions of this state process a modification of life cycle, setting of load values and specification of initial load rates. The task of an output action is to verify changed load values. After processing, there is an event AdjustedValues, which transfers a whole process to the state TestProcedureDefinition.

Testing script from the previous iteration is loaded for its next modification after the gaining of this state. This state is specific with a generating of testing script which moves the testing process into the state TestExecution where the testing script is loaded. Created testing script must be verified first.

The state TestExecution is a concrete test processing. The real testing process must be kept and all errors occurring during the testing process were documented. It means a fulfilment of basic requirements for test processing from perspective of safety-critical systems. It is important to

collect and save all information needed from a perspective of testing of application communication. The main task of its output action is a saving of all gained data from an actual iteration.

The test enters the state TestSummary after the reaching them where data is loaded in an actual iteration. This state has a task to summarize important information from an actual iteration and document them for a total test process. In case of the last iteration it is needed to determine the next process of testing. The result of this state is the protocol on test which is created with an output action. This protocol should be in an appropriate format for the next support tool.

The event EvaluatedTest serves to initiate the next test iteration. The whole testing process is repeated when more than 50% of elements from testing sample in all iterations failed, or if an error occurs in the first iteration which presents the fully censored test. These conditions are based on standard processes of the step stress testing and they present conditions for test accomplishment.

5. VERIFICATION OF THE DESIGNED TESTING MODELS

Verification or validation must be processed to verify a correctness of the designed models of performance and step stress testing of communication subsystem of safety-critical control systems. The verification would approve if the designed models were created correctly considering their specification. The validation would focus on an approval of designed models in connection with real expectations and requirements of customers (41).

The area of testing of safety-critical systems what our designed models of performance and step stress testing are focused on, present a concrete specified area. Specification of designed models can be gained with the analysis of concrete standards and guidelines. The problem is an identification of importance of particular requirements by testing of safety-critical systems. The real customer requirements for testing of safety-critical systems are also difficult to reach. These requirements are mostly a part of internal company guidelines and they focus on a concrete and specified area. They are the part of a company know-how and no firm allows an application outside of firm.

It is very difficult to process the validation when our designed testing models present a certain generalization or a unified process for a specified area. Therefore we have decided to verify the designed models against our specified requirements for testing. These requirements are based on the generalized requirements for testing of safety-critical systems which should be gained with an analysis of standards and guidelines for these systems.

It is recommended to use an appropriate metrics for a definite verification process. Therefore we have defined metrics for our designed models which determines a completeness of implemented model requirements. Based on its value we will evaluate if the concrete model can be considered as verified.

5.1 Metrics Proposal

Our proposal of metrics was based on the standard metrics for software. We have analysed particular quality attributes for software products and we have identified an appropriate group. The standard metrics was selected from this group which will be a base for our proposal.

The selected standard metrics was modified for a planned process of validation and the main parameters were defined. The specified requirements for testing of safety-critical systems were classified into three groups to simplify the validation process.

5.1.1 Selection of Appropriate Metrics

There are many standards defining a software quality and concrete metrics to secure them in the area of informatics including also safety-critical computer systems. Therefore we have used the standard ISO 9126 by a selection of suitable metrics. This one provides internal and external metrics for the quality of software products.

Analysis of particular groups of quality attributes helped to identify the most suitable group of functional characteristics. Functionality can be defined as an ability of information system to include functions which

secure the supposed or defined needs of users by a system application in defined conditions. They can be used for a measurement of a quality of product requirements fulfillment according to definitions in its assignment (42). This group of metrics is therefore suitable as a base for our verification of designed models.

Considering the definition of validation process we have focused especially on internal metrics. These are used for measurement of product characteristics which cannot be deduced from its behaviour (42)

This criterion limits particular quality attributes which do not have any internal metrics (for example functionality compliance).

The identified suitability is the most ideal concrete functionality characteristic. This can be defined as an ability to provide functions for an ensuring of specified tasks and needs of user. It considers if the functions are available or not. Its basic internal metrics is a completeness of implemented functions which is defined as (42):

$$X = 1 - \frac{A}{B}, \qquad [1]$$

where X is a value of metrics,

A is a number of functions found by the control according to the specification,

B is a number of functions in a specification.

This metrics is completed with an operating with scales and it will present a base for our suggested metrics.

5.1.2 Definition of Metrics

We have defined the following metrics to verify our designed models. Its value will be considered as a criterion which will determine if the designed model follows or not the specified requirements.

Name and identification

Metrics determines the Integrity of Implemented Model Requirements. This gives a question what is the completeness of implementation of concrete requirements. It can be added to the internal metrics for an evaluation of adequacy which is one of the functionality characteristics.

Definition

Metrics for a determination of integrity of implemented functions is defined with a function:

$$X(\frac{A_1}{B_1}, \frac{A_2}{B_2}, \frac{A_3}{B_3}) = 1 - \frac{A_1 W_1}{B_1} + \frac{A_2 W_2}{B_2} + \frac{A_3 W_3}{B_3}, \qquad [2]$$

where X is a function of metrics,

A_1 is a number of our implemented specified general requirements for testing,

A_2 is a number of our implemented specified general requirements for test specification,

A_2 is a number of our implemented specified general requirements for test processing,

B_1, B_2 and B_3 is a total number of our specified requirements for a concrete attribute,

and W_1, W_2 and W_3 are their relative weights.

Determination of weight of concrete attributes

We have divided the designed models into three categories to simplify the verification of designed models. We determined a value of its importance for every category while we decided for the constant weight considering the testing area. It is especially because we cannot define the importance of requirements from perspective of testing of safety-critical systems. The constant weight brings problems in the calculation exactness and therefore we strengthened the general requirements for testing of safety-critical systems. From this reason our specified requirements for testing have a concrete normalized value 4 (from the interval <1, 5>) and requirements for specialization and test processing has the normalized value 3. The weight for very attribute was defined on the base of (43):

$$W_i = \frac{E_i}{(\sum_{i=1}^{n} E_i)}, \qquad [3]$$

where W_i is a defined weight,

E_i is a normalized attribute value from the interval <1, 5>.

Values of weight of concrete attributes determined according to the [3] as well as their normalized values are in the Table 6.

OVERVIEW OF ATTRIBUTES OF A PROPOSED METRICS AND THEIR WEIGHT Table 6

Index	Attributes	Normalized value of attribute	Weight
1	Implemented our specified general requirements for testing	4	0.4
2	Implemented our specified general requirements for test specification	3	0.3
3	Implemented our specified general requirements for test processing	3	0.3

Dimension

The defined metrics reaches values in the interval <0, 1>, the lower values of metrics mean a higher quality of designed model from perspective of our specified requirements for testing of safety-critical systems. This range presents a standard between metrics in the standard ISO 9126.

Initial and Final Value

The designed models are very specified therefore a determination of the final value is not easy. The intended application of designed metrics is a verification of our designed model and there are no previous values (or initial values) to simplify an estimation of required final value of metrics.

Our designed models deal with the testing of safety-critical systems, so the achieved values of metrics should be as low as possible. It means that designed models should implement the highest number of our specified requirements for testing of safety-critical systems.

Source of Data for Measurement

Data needed for a determination of metrics value come from two basic sources. The first are our specified requirements which are a base for our designed model. These have been reached after analysis of standards and guidelines for safety-critical systems. Our specified requirements present particular attribute groups and we have to determine the implementation completeness.

The second source is a concrete testing model of safety-critical system which we want to verify. The particular model implements requirements for testing of safety-critical systems and its partial actions and events must reflect and include these requirements. It does not need to be directly the specific actions or events but requirements can be included into concrete elements of the designed model.

Processes of verification

To determine the values of metrics it is needed to define requirements which are implemented by the concrete model. In our specification of requirements for testing of safety-critical systems we have identified concrete requirements which must be implemented in the specific testing models. It is important to classify them into one of the attribute groups which they represent. This step determines a total number of our specified requirements of concrete group.

Then it is needed to analyse all steps, actions and states of a concrete testing model and to assign them into our requirements. Some requirements are implemented only in a verbal way therefore it is needed to add them to a concrete specified requirement. Based on this classification we can

determine a number of implemented requirements in a concrete designed model.

The value of proposed metrics is calculated after these parameters according to the [2], which determines the completeness of implemented requirements of a concrete testing model.

5.2 Verification of Designed Model of the Performance Testing

Verification of the designed model of performance testing was processed with our defined metrics. The requirements which could not be fulfilled have been excluded. We have identified a number of implemented requirements of concrete attributes according to the determined process of metric verification and based on them we calculated the values of metrics. The Table 7 illustrates the specified general requirements for testing, the concrete actions and steps which implement them. The total number of our specified requirements was 16, while 15 from them implement the designed model of performance testing.

IMPLEMENTATION METHOD OF OUR SPECIFIED GENERAL REQUIREMENTS IN THE DESIGNED MODEL OF PERFORMANCE TESTING Table 7

Specified requirement	Method of Requirement Implementation
Testing cases	
Data specification for testing	Steps specifying required data in the phase TestCaseSpecification, Action defineTestData in a state ProblemDomainSpecification
Determination of expected test results	Step determineExpectedResults in the phase TestCaseSpecification
Testing environment	
Definition of support tools for testing	Step specifySupportTools in the phase TestProcedureSpecification
Definition of support software for testing	Step specifySupportingSoftvare in the phase TestProcedureSpecification
Description of configuration of tested system	Step describeConfiguration in the phase TestTransmittalReport
Documentation	
Documentation of test results	Step writePerfSessionData in the phase TestLog and other associated steps, Action of creation of createPerformananceSessionData in the state SessionExecution
Documentation of aims and criteria fulfillment	Analysis step analyzeTestResult in the phase TestSummaryReport, Action of createTestLog in the state TestSummary
Documentation of test errors	Step of the recordError in the phase TestIncidentReport, Action of creation of createTestIncidentReport in the state SessionExecution

Specified requirement	Method of Requirement Implementation
Configuration Data	
Input configuration data must be in an allowed range	Step of verification of verifyInputData in the phase TestTransmittalReport
Input configuration data mutst be approved combinations	Step of verification of verifyInputData in the phase of TestTransmittalReport
Data Communication	
Ensure a data correctness from perspective of transfered data	In the text phases TestExecution and TestSummaryReport and in the state SessionExecution
Secure a data transfer in a real time	In the text phases TestExecution and TestSummaryReport in the state SessionExecution
Support Tools	
Standardized process inputs of testing	
Standardized process outputs of testing	Defined in the text phase TestSummaryReport, The action of createTestLog in the state TestSummary
General	
Determined testing criteria for the test finishing	Step of testCondition in the phases TestCaseSpecification and TestExecution, Event of testCondition moves the test from the state of ProblemDomainSpecification into the state SessionCreation
Reduction of number of input combination	Steps of InputCombinationReduction and analysis in the phase TestCaseSpecification

Then we have determined all our specified requirements for the test specification. The appropriate actions and steps of the performance testing model can be added. They are all illustrated in the Table 8. The total

number of our specified requirements was 7. The designed model of performance testing implements them all.

IMPLEMENTATION METHODS OF OUR SPECIFIED REQUIREMENTS FOR TEST SPECIFICATION IN THE DESIGNED MODEL OF PERFORMANCE TESTING Table 8

Specified requirement	**Method o requirement implementation**
Testing and Test Plan	
To document all non-tested components	In the text phase of TestPlan
To store a list of persons by the test	In the text phase of TestPlan
For every created test	
To determine input testing requirements for every test	Included in the step of testObjectives in the phase TestCaseSpecification, The action defineTestData in the state ProblemDomainSpecification
Scenario of Testing	
To label and name the test	Step UniqueIdentifier in the phase TestCaseSpecification
To determine prerequisite for testing	Contained in the step of testObjectives in the phase TestCaseSpecification, Action of defineTestData in the state of ProblemDomainSpecification
To define orders for test process	Contained in the step of testObjectives in the phase TestCaseSpecification, Action of defineTestData in the state ProblemDomainSpecification
To define activities for test accomplishment	Step of returnToStandardState in the phase TestTrasmittalReport

At the end we have selected specified requirements for testing process and we added concrete actions and steps implementing them to the requirements. This assignment is displayed in the Table 9. The total number

of our specified requirements was 6, while the designed model of performance testing implemented them.

IMPLEMENTATION METHOD OF OUR SPECIFIED REQUIREMENTS FOR TEST PROCESSING IN THE DESIGNED MODEL OF PERFORMANCE TESTING Table 9

Specified requirement	**Implementation Method of Requirement**
For every test	
To save a real testing procedure	Step of writeRealTestProcess in the phase TestLog, Action description of describeRealTestProcess in the state SessionExecution
To save the test results	Step of RealTestProcessReport in the phase TestLog, Action of createTestLog creation in the state TestSummary
To save a support project documentation	Processed with concrete phases IEEE 829
Test Evaluation	
Final evaluation of test results (Pass / fail)	Steps of analyzeIncident and evaluateIncident in the phase TestSummaryReport, Action of evaluateAllConflicts in the state TestSummary
If test failed to classify it as a negative test	In the text in the phase TestIncidentReport
To determine the next process for failed tests	Step of determineFurtherAction in the phase TestSummaryReport

We calculated metrics value for the designed model of performance testing based on a number of our specified and implemented requirements for testing of safety-critical systems according to the [2]:

$$X\left(\frac{15}{16},\frac{7}{7},\frac{6}{6}\right)=1-\frac{15\times 0,4}{16}+\frac{7\times 0,3}{7}+\frac{6\times 0,3}{6} \qquad [4]$$

$$X(\frac{15}{16},\frac{7}{7},\frac{6}{6}) = 0{,}025 \cdot \quad [5]$$

The lower the metrics value is, the more quality the designed model of performance testing is from a perspective of our specified requirements for testing of safety-critical systems. Considering the calculated metrics value (which is close to the ideal value) we can say that the designed model of performance testing follows the specified requirements.

The designed model of performance testing can be considered as verified in relation to our specified requirements for testing of safety-critical systems.

5.3 Verification of Designed Model of the Step Stress Testing

Based on our defined metrics we made also a verification of the designed model of step stress testing. We did not include the requirements which cannot be fulfilled considering the character of the step stress test. Our specified general requirements for testing of safety-critical systems, we summarized them together with concrete actions and steps which implement them in the designed model of the step stress testing in the Table 10. The total number of our specified requirements was 17, while 16 of them implement the designed model of step stress testing.

IMPLEMENTATION METHOD OF OUR SPECIFIED REQUIREMENTS IN THE DESIGNED MODEL OF STEP STRESS TESTING

Table 10

Specified requirement	Method of requirement implementation
Testing cases	
Specification of data for testing	More steps in the phase TestCaseSpecification, Event of gaining of gainedSpecificationOfRelevantData between the states PreviousStandardStates and Sampling
Determination of expected test results	Step for determineExpectedResults in the phase TestCaseSpecification
Testing environment	
Definition of support tools for testing	The step specifiySupportTools in the phase TestProcedureSpecification
Definition of support software for testing	The step specifiySupportSoftware in the phase in the phase TestProcedureSpecification
Description of configuration of tested system	The step describeConfiguration in the phase TestTransmittalReport, Action describeConfigruation in the state Sampling
Documentation	
Documentation of test results	The step writeCurrentIterationData in the phase of TestLog, The action of saving of saveCurrentIterationData in the state TestExecution
Documentation of aims and criteria fulfillment	Steps of evaluation and testResults in the phase TestIncidentReport
Documentation of test failures	The recordError in the phase TestLog, Action of recordErrors in the state TestExecution
Documentation of corrections and process of repair actions by failure	The step recordCorrection in the phase TestTrasmittalReport

Specified requirement	Method of requirement implementation
Configuration data	
Input configuration data must be in an approved range	The step verifiyDimensions in the phase TestTrasmittalReport
Input configuration data must have approved combinations	The step verifiyDimensions in the phase TestTrasmittalReport
Data communication	
To secure a data correctness from perspective of transferred data	In text phases TestLog and TestSummaryReport and on the state TestExecution
To secure a data transfer in a real time	In text phases TestLog and TestSummaryReport and in the state TestExecution
Support Tools	
Standardized inputs of testing process	
Standardized outputs of testing process	In text phases TestSummaryReport and In the state TestSummary
General	
Determined testing criteria after accomplishment of test	Defined in test process characteristics.
Reduction of number of input combinations	The reductionOfDimensions in the phase TestCaseSpecifications.

Then we added the next specified requirements for specification of tests to concrete steps which implement them in the designed model of stress testing. All are in the Table 11. The total number of our specified requirements was 7, while the designed model of step stress testing implemented them all.

IMPLEMENTATION METHOD OF OUR SPECIFIED REQUIREMENTS FOR TEST SPECIFICATION IN THE DESIGNED MODEL OF STEP STRESS TESTING Table 11

Specified Requirement	**Method of requirement implementation**
Testing and Testing Plan	
Documentation of all non-tested components	In text phase TestPlan
Keep a list of persons by the test	In text phase TestPlan
For every created test	
Every test has to have input requirements for testing	Many steps in phase TestCaseSpecification, Event gaindedSpecificationOfRelevantData between the states PreviousStandardStates and Sampling
Testing Scenario	
Definite labeling and name of test	The Step uniqueIdentification in the phase TestCaseSpecification
Determination of prerequisites for testing	The step specifyTestProcess in the phase TestCaseSpecification
Defining of orders for test processing	The step specifyTestProcess in the phase TestCaseSpecification
Defining of activity for test accomplishment	The step fixParts at the end of iteration in the phase TestTransmittalReport or the action fixParts in the state Sampling

At the end we determined an implementation method of our specified requirements for testing. These requirements with implementing steps are summarized in the Table 12. The total number of our specified requirements was 5. The designed model of the step stress testing implements all these requirements.

IMPLEMENTATION METHOD OF OUR SPECIFIED REQUIREMENTS FOR TESTING IN THE DESIGNED MODEL OF STEP STRESS TESTING

Table 12

Specified Requirement	**Implementation Method of Requirement**
For every Test	
To save a real testing procedure	The step writeRealTestProcess in the phase TestLog, the action writeRealTestProcess in the state TestExecution
To save the test results	The writeCurrentIterationData in the phase TestLog, the event receivedResultsOfTheTest among the states TestExecution and TestSummary
To save a support project documentation	Proceeded with particular phases IEEE 829
Test Evaluation	
Definte evaluation of test results (Pass / fail)	In text phase TestSummaryReport
To determine the process for test which failed	The step determineFurtherAction in the phase TestSummaryReport, the action determineFurtherAction in the state TestSummary

We have calculated a value of metrics for designed model of step stress testing based on a number of our specified requirements for testing of safety-critical systems and their implementation according to the [2].

$$X(\frac{16}{17},\frac{7}{7},\frac{5}{5}) = 1-\frac{16\times 0,4}{17}+\frac{7\times 0,3}{7}+\frac{5\times 0,3}{5} \qquad [6]$$

$$X(\frac{16}{17},\frac{7}{7},\frac{5}{5}) = 0,024. \qquad [7]$$

The lower the value of metrics is, the more quality the designed model of step stress testing has. It is from perspective of our specified

requirements for testing of safety-critical systems. Considering the calculated value of metrics (closer to zero, closer to the ideal value) we can say that the designed model of step stress testing fulfils requirements.

The designed model of step stress testing can be considered as verified to specified requirements for testing of safety-critical systems.

6. ACHIEVED RESULTS

The main aim of this monograph is the model design of a step stress and performance testing of safety-critical systems. It was necessary to specify requirements for testing of safety-critical systems and to process various support activity during testing.

At the beginning we have analysed relevant standards and guidelines for safety-critical systems. The standard IEEE 829 was analysed first which focuses on a documentation and concrete process phases of testing software. The next was the standard IEC 61508, which is the basic standard for safety-critical systems. The next standards were IEC 50128, IEC 61511, CAP 670, DO-178B, IEC 62061 and guideline MISRA representing the main regional and product standard and guideline for specific areas of safety-critical systems. The standards IEC 60880, IEC 61513, IEC 62138, IEC 60987 and IEC 61226 were excluded from the previous group of standards and guidelines for area of nuclear power stations. At the end we analysed the guideline GAMP which focuses on validation of automated systems.

The analysed standards are specialised prevailing on integration testing. System testing is mentioned only as one of the tests which must be made. Requirements gained in analysis must be generalized. Except of testing we have focused on the requirements from perspective of data communication which must be considered by testing of communication subsystem.

Concrete areas of safety-critical systems are enough specific that they cannot be included into model design. We would have to know an exact

specification and details of concrete tested system by a model design on this level.

Therefore we have decided to focus on the testing of safety-critical systems in general. We have applied the general requirements for safety-critical systems which were specified into details with using of knowledge from regional and product standards and guidelines including the standards and guidelines for nuclear power plants. According to these requirements we have identified differences and specified requirements for testing of safety-critical systems.

Based on the analysed standards we have identified and suggested basic steps of design and development of safety-critical systems. The analysed standards and guidelines do not include a design and development of system testing so it was needed to implement them there. We have defined actions and requirements for every step of our designed process. Or we defined those which must be followed by the process. We identified and specified requirements and data for system testing of safety-critical systems. Our suggested steps as well as specified requirements and data were captured with appropriate UML diagrams.

Based on our specified requirements for testing and previous design we started with a design of models system testing of communication subsystem by safety-critical control systems. We focused on two types of test systems, specifically the step stress and performance testing. These models are based on the same outputs while each is specialised on testing of other system aspects. Both models implement our specified requirements for testing of safety-critical systems as well as our specified requirements for testing of communication subsystem.

The designed model of performance testing is focused on testing of performance of safety-critical system. It is orientated on a concrete testing process and concrete states which occur by the testing. To reflect particular perspectives we applied the diagrams of the model language UML (concrete sequential and state diagrams).

On the other hand the designed model of step stress testing is focused on reliability and robustness of safety-critical systems. We identified a step continuity of the step stress testing for standard systems before a concrete design. These steps were used as a base for propose of testing process and identification of states which can occur. To display them we used the appropriate diagrams of language UML (concrete diagrams of activities, sequential and state diagrams).

Suggested testing models had to be verified after their creation. Due to the fact that the safety-critical systems are in a specific area where the real documents are difficult to reach, there were suggested testing models to be verified in connection to specified requirements for testing. We have suggested metrics for their verification and we defined its parameters. The suggested metrics helps to reach values which we use when we want to determine the range of implementation of specified requirements for testing of safety-critical systems.

Concrete aims of the monograph can be considered as fulfilled based on the previous evaluation of gained results.

CONCLUSION

Today, there is a higher emphasis on safety, not only in specific areas such as nuclear power engineering and the air industry, but also in common industry in a wider sense. One of the priorities of the European Union is certification in various spheres and it is also focused on safety in different spheres.

The systems with tasks to keep safety of controlled process or equipment are called safety-critical systems. Failure of these systems can have catastrophic consequences and it can endanger health, human lives or environment. The word “critical” has the task to emphasize possible causes in their failure. These systems are designed with a certain level of security. There are more factors which can harm them. Safety of these systems must be approved and therefore also tested.

Designed testing models focus on the testing of safety-critical systems as a whole. Their base is the system testing which can start after integration testing. We used the standard system testing by their design. These were completed with requirements for testing of safety-critical systems.

We have proposed two models of system testing made by the step stress and performance testing. Each of these system tests uses a different principle and it evaluates a system from another perspective. Both models are measured for testing of communication subsystems. The area of safety-critical systems is very specific so the designed model operates with a certain rate of abstraction. The model which would reflect concrete actions had to be proposed for a concrete system and to know its specification.

Proposed models can be enlarged from many points of view. It is possible to complete them with more detailed steps for a concrete safety-critical system or to focus on a specific area. Or the steps can be added which are required with other standards or to specialise on other system area. Or other perspectives on proposed testing processes can be created with use of other UML diagrams.

REFERENCES

1. DIJEV, S. Elements of Industrial Automation - Part 2: Industrial Networks for Communication and Control [online]. Sofia: Technical University of Sofia, 2007 [quot. 2010-01-24]. Available on the Internet <http://anp.tu-sofia.bg/djiev/PDF%20files/Industrial%20Networks.pdf>
2. BERTOLINO, A. Software Testing Research and Practice In The 10th International Workshop on Abstract State Machines, Italy, 2003, p. 1-21
3. JACOBSON, I., CHRISTERSON, M., JONSSON, P., ÖVERGAARD, G. Object-Oriented Software Engineering, Addison-Wesley, 1994.
4. BINDER, R.V. Testing Object-Oriented Systems, Addison-Wesley, 1999
5. VAŽAN, P., TANUŠKA, P., KEBÍSEK, M. The data mining usage in production system management. In: World Academy of Science, Engineering and Technology. Year 7, Issue 77 (2011). ISSN 2010-376X
6. JAKUŠ, M. Testovanie – metóda zabezpečenia kvality softvérového produktu. (Testing-Method of Quality Assurance of Software Product.) Bratislava: STU
7. PAREKH, N. Software Testing: Acceptance testing [online]. [cit. 2010-01-12]. Available on the Internet <http://www.buzzle.com/articles/software-testing-acceptance-testing.html>
8. HEDBERG, J. Factory Acceptance Testing Guideline [online]. 2006. [cit. 2010-01-20]. Available on the Internet <http://www.sp.se/sv/index/services/functionalsafety/Documents/Factory%20acceptance%20testing%20guideline%20Process.pdf>
9. BORYSOWICH, C. Creating Acceptance Test Case Approaches & Scenarios [online]. 2008. [cit. 2010-01-23]. Available on the Internet <http://it.toolbox .com/blogs/enterprise-solutions/creating-acceptance-test-case-approaches-scenarios-24823>

10. DESIKAN, D., RAMESH, G. Software Testing: Principles and Practice. India: Pearson Education, 2006. 486p. ISBN 978-81-7758-121-8
11. PAREKH, N. Software Testing: Stress Testing [online]. [cit. 2010-01-12]. Available on the Internet <http://www.buzzle.com/articles/software-testing-stress-testing.html>
12. PORTER, A. 2004. Accelerated Testing and Validation. Burlington, USA : Newnes, 2004. 242 pp. ISBN 0-7506-7653-1.
13. KUNDU, D.; BALAKRISHNAN, N.; KANNAN, N.; NG, T. 2006. Step Stress Model [on line]. Kanpur, India : IIT Kanpur, 2006. [cit. 29.1.2010]. Available on the Internet <http://home.iitk.ac.in/~kundu/seminar25.pdf>
14. RPM SOLUTIONS. Performance Tests [online]. [cit. 2012-5-8]. Available on the Internet <http://loadtest.com.au/types_of_tests/ performance_tests.htm>
15. SENN H. Load and performance testing for software applications [ONLINE]. Testing Experience, Germany, ISSN 1866-5705. [cit. 2010-03-08], Available on the Internet <http://www.testingexperience.com>
16. RPM SOLUTIONS. Load Tests [online]. [cit. 2012-5-8]. Available on the Internet <http://loadtest.com.au/types_of_tests/load_tests.htm>
17. SCHWARZ, M. A. Einführung in die Softwaretechnik für sichere und verlässliche Software.(Theory of Software Technology for Secure and Reliable Software.) Germany: Institut für Informatik im Paderbom, 2004. 25s.
18. SOJKA, M. Programování systémů reálného času (Programming of System of Real Time.), Praha: ČVUT, 2010
19. FRANEKOVÁ, M., KÁLLAY, F., PENIAK, P., VESTENICKÝ, P. Komunikačná bezpečnosť priemyselných sieti (Communication Security of Industrial Networks.), Žilina: Žilinská Univerzita, 2007. ISBN 978-80-8070-715-6

20. COLEY CONSULTING. IEEE 829 Documentation [online]. [cit. 2012-04-10]. Available on the Internet <http://www.coleyconsulting.co.uk/IEEE829.htm>
21. COLEY CONSULTING. IEEE 829 Test Plans [online]. [cit. 2012-04-10]. Available on the Internet <http://www.coleyconsulting.co.uk/testplan.htm>
22. IEC 61508-1: Funkčná bezpečnosť elektrických/ elektronických/ programovateľných elektronických bezpečnostných systémov. (Functional safety of electrical/ electronic/ programmable electronic safety-related systems.) Part 1: General Requirements. 2010
23. IEC 61508-2: Funkčná bezpečnosť elektrických/ elektronických/ programovateľných elektronických bezpečnostných systémov. (Functional safety of electrical/ electronic/ programmable electronic safety-related systems.) Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems. 2010
24. IEC 61508-3: Funkčná bezpečnosť elektrických/ elektronických/ programovateľných elektronických bezpečnostných systémov. (Functional safety of electrical/ electronic/ programmable electronic safety-related systems.) Part 3: Requirements for programme equipment. 2010
25. BSI. British Standards Institution [online]. 2012 [cit. 2012-05-20] Available on the Internet <http://shop.bsigroup.com/>
26. ARC ADVISORY GROUP. Reduce Risk with a State-of-the-Art Safety Instrumented System [online]. Boston, 2004 [cit. 2012-04-10] Available on the Internet <http://www2.emersonprocess.com/siteadmincenter/PM DeltaV Documents/ Articles/ARCWhitePaper/Reduce-Risk-with-a-State-of-the-Art-Safety-Instrumented-System.pdf >
27. CAP670: Air Traffic Services Safety Requirements [online]. 2012 [cit. 2012-03-24] Available on the Internet <http://www.caa.co.uk/docs/33/cap670.pdf>

28. CRITICAL SYSTEMS LABS. DO-178B [ONLINE]. [cit. 2012-05-12] Available on the Internet <http://www.criticalsystemslabs.com/pgs/DO_178B.html#RTCA_DO-178B_overview>
29. IEC. International Electronic Commission Webstore [ONLINE]. [cit. 2012-05-13] Available on the Internet <http://webstore.iec.ch >
30. MISRA CONSORTIUM. Development Guidelines for Vehicle Based Software. 1994, ISBN 0-9524156-0-7
31. TANUŠKA, P., GESE, A., KEBÍSEK, M., VLKOVIČ, O., ŠPENDLA, L., SCHREIBER, P., VAŽAN, P. Porovnanie požiadaviek na zabezpečenie kvality bezpečnostných systémov a prevádzkových riadiacich systémov jadrových zariadení. (Comparsion of the requirements for quality assurance of safety systems and operational control systems of nuclear power plants.) Bratislava: STU, 2010. 58s.
32. BEL V, BfS, Consejo de Seguridad Nuclear, ISTec, NII, SSM, STUK 2010. Licensing of safety critical software for nuclear reactors: Common position of seven European nuclear regulators and authorized technical support organisations [Online]. 2010. 157s. [cit. 2011-04-11] Available on the Internet <http://www.belv.be/images/pdf/part_1_and2_version_2010.pdf>
33. MAAE NS-R-1 – Safety of Nuclear Power Plants: Design.
34. IEC 60880: Nuclear Power Plants – Instrumentation and control systems important to safety – Software Aspects for computer – based systems performing category A functions.
35. IEC 61513: Nuclear Power Plants – Instrumentation and control for systems important to safety – General requirements for systems.
36. IEC 62138: Nuclear Power Plants – Instrumentation and control important to safety – Software aspects for computer – based systems performing category B or C functions.
37. GAMP4 - D6: Guideline for the Testing of an Automated Systems

38. PHARMPRO. GAMP Standards For Validation Of Automated Systems [Online]. 2008. [cit. 2012-04-8] Available on the Internet <http://www.pharmpro.com/Articles/2008/03/GAMP-Standards-For-Validation-Of-Automated-Systems/>
39. ŠKULAVÍK, T., SCHREIBER, P., MORAVČÍK, O. Steady state response simulation of a 2-D micromirror in Simulink. In: Advanced Materials Research. Vol. 488-489 : 2012 2nd International Conference on Key Engineering Materials, ICKEM 2012, Singapore, 26-28 February 2012 (2012). ISBN 978-303785382-5
40. DUNN, W. R. Practical Design of Safety-Critical Computer Systems. USA: Reliability Press, 2002. 358p. ISBN 0-9717527-0-2
41. KOPČEK, M., ŠKULAVÍK, T. Experimental Verification of the Computational System for the Optimal Pilot Bus Selection. In: Lecture Notes in Engineering and Computer Science. WCECS 2012. Vol. II. : World Congress on Engineering and Computer Science 2012. Proceedings IAENG & IET. San Francisco, USA, 24-26 October, 2012. - Hong Kong : International Association of Engineers, 2012. - ISBN 978-988-19252-4-4
42. UČEŇ, V., NOVOTNÝ, O., SLOVÁK, J., VODIČKOVÁ, K. Metriky v informatice: Jak objektivne zjistit přínosy informačního systému. (Metrics in Informatics: How to find out the contributions of information system from objective perspective.) Havlíčkúv Brod: Grada, 2001. ISBN 80-247-0080-8
43. BUSKEY, C. D. A Software Metrics Based Approach to Enterprise Software Beta Testing Design. Dissertation thesis, Pace University, 2005. 268p.

CONTENTS

Author:	Ing. Lukáš Špendla, PhD.
Title:	DESIGN FOR A TESTING MODEL OF A COMMUNICATION SUBSYSTEM FOR A SAFETY-CRITICAL CONTROL SYSTEM
Published in:	Ilmenau (Germany)
Edited by:	Universitätsverlag Ilmenau
1st edition:	2013
Editions:	Scientific monographs in Automation and Computer Science
Pages:	140 (figures 21; tables 12; proofreading – Brian Green)

Quires (20 standard pages): 5,59