The Proposal of Step Stress Testing Automation

P. Tanuska, *Member, IACSIT, Member, IAENG,* O. Vlkovic and L. Spendla

***Abstract*—The main goal of article is the automation possibilities of accelerated stress testing. The proposal is focused into control systems as a part of Information systems. The contribution demonstrates the usage of Step Stress Test model in the fully automated test system, using minimum of human intervention. Our proposal captures the particular states which occur in the test system and has been visualized in UML diagrams.**

***Index Terms*— stress testing, automation, UML**

I. Introduction

The testing is one of the most important phase in the development cycle. It can be done by either manual testing or automation testing. The testing scope and activity is designed in feature test suite basis. Testing scope usually falls under one of the main aspects of testing including Utility, Reliability, Robustness, Performance and Correctness. Individual aspects may seem isolated, but there is a close link between them and they mostly cannot be met without at least a partial fulfillment of the other aspects.

A. Accelerated Testing

Accelerated testing is an approach for obtaining more information from a given test time than would normally be possible. It does this by using a test environment that is more severe than that experienced during normal equipment use. Since higher stresses are used, accelerated testing must be approached with caution to avoid introducing failure modes that will not be encountered in normal use. Accelerated testing falls into two main categories, each with a specific purpose: Accelerated Life Testing and Accelerated Stress Testing [5].

B. Accelerated Stress Testing

Uses accelerated environmental stresses to precipitate latent defects or design weaknesses into actual failures to identify design, part or manufacturing process problems which could cause subsequent failures in the field. Requires a thorough understanding, or at least a workable knowledge, of the basic failure mechanisms. Estimation of item life may, or may not, be a concern [5].

Manuscript received Jun 22, 2010.
Pavol Tanuska is with the Institute of Applied Informatics, Automation and Mathematic at the Slovak University of Technology, Trnava, Slovakia, (phone: +421 918 646 061; fax: +421 33 5511758; e-mail: pavol.tanuska@stuba.sk).
Ondrej Vlkovic is with the Institute of Applied Informatics, Automation and Mathematic at the Slovak University of Technology, Trnava, Slovakia, (e-mail: ondrej.vlkovic@stuba.sk).
Lukas Spendla is with the Institute of Applied Informatics, Automation and Mathematic at the Slovak University of Technology, Trnava, Slovakia, (e-mail:Lukas.spendla@stuba.sk).

Accelerated test models relate the failure rate or the life of a component to a given stress such that measurements taken during accelerated testing can then be extrapolated back to the expected performance under normal operating conditions. The implicit working assumption here is that the stress will not change the shape of the failure distribution [5].

In choosing a model, the key criterion is that it accurately models the reliability or life under the accelerated conditions to the reliability or life under normal operating conditions. Great care is essential in choosing the most appropriate model, and in selecting the appropriate range of validity for the chosen model in a specific application. Documenting the rationale for these choices is important [5]. There are several Accelerated Test Models, to the most common we can include:

Inverse Power Law

The inverse power law relationship (or IPL) is commonly used for analyzing data for which the accelerated stress is nonthermal in nature [6]. The inverse power law (IPL) model is given by,

$$L(V) = \frac{1}{K.V^n} \quad (1)$$

where:

- L represents a quantifiable life measure, such as mean life, characteristic life, median life, B(x) life, etc.
- V represents the stress level.
- K is a model parameter to be determined, (K > 0).
- n is another model parameter to be determined.

Arrhenius Acceleration Model

The Arrhenius relationship is commonly used for analyzing data for which temperature is the accelerated stress [6]. The Arrhenius model is given by,

$$(V) = C.e^{\frac{B}{V}} \quad (2)$$

where:

- L represents a quantifiable life measure, such as mean life, characteristic life, median life, or B(x) life, etc.
- V represents the stress level (in absolute units if it is temperature).
- C is a model parameter to be determined, (C > 0).
- B is another model parameter to be determined.

Miner's Rule (Fatigue Damage)

The fatigue damage model is given by,

$$CD = \sum_{i=1}^{k} \frac{C_{Si}}{N_i} \leq 1 \quad (3)$$

where:

- CD = cumulative range

- C_{Si} = number of cycles applied at a given mean stress Si
- N_i = the number of cycles to failure under stress Si, (as determined from an S-N diagram for that specific material)
- k = the number of loads applied

This model assumes every part has a finite useful fatigue life and every cycle uses up a small portion of that life. Failure is likely to occur when the summation of incremental damage from each load equals unity. Miner's rule does not extend to infinity, however. It is valid only up to the yield strength of the material; beyond that point it is no longer valid [5].

C. *Fully Censored Test*

Stress testing is based on pass or fail against desired reliability goal. The test is designed around hypothesis: a product has a given reliability with a given confidence. The hypothesis is tested by determining how many parts should last for a given period of time at a given severity with no failures. This type of test is called fully censored because (if it passes) the test is ended before any parts fail. In other words, the time-to-failure is censored. The precision of the test depends on two key factors: number of parts tested and number of lives tested [3].

D. *Step Stress Testing*

As an extension of stress testing we can consider step stress testing, which seeks to improve some of the stress testing features. A step stress test starts the same way a fully censored testing starts. A fixed number of parts run through one live, what is for example represented by 500 hours on Fig. 1. After one live, the stresses applied to the product are elevated in steps in order to precipitate failures. The precision of the test depends on three key factors:

1) Number of parts tested
2) Number of lives tested
3) How the stresses are increased

The grater the number of parts, the more precise (and more accurate) the test is. Increasing the number of lives decreases the number of parts needed, but introduces an assumed distribution in the population. Increasing the stress reduces increases the graduation on the time-to-failure. At low stress, the time to accumulate damage is very long and differences between time-to-failure will also be long.

As stress is increased, the rate of damage increases, the time to failure decreases and the spread between times to failure decrease [3].

The accuracy of step stress testing depends on accuracy of both testing steps. The first part of step stress test is fully censored, pass/fail test, as you can see on Figure 1. Therefore the accuracy is same like with fully censored stress testing. The accuracy of the second part of the test depends on how stresses are increased. If all of the stresses are stepped in a way to ensure that the rate of damage for each failure mechanism is kept in proper proportion, then the accuracy will be driven by the same issues as for the one life portion of the test. If the stresses are increased in a way that biases the test toward one failure mode, then the time-to-failure results will be inherently inaccurate.

The resources for a step stress test are the same as for a fully censored test. That means that the resources depend on the service stress applied. Most test use a single axis vibration and relatively low thermal ramp rates [3].

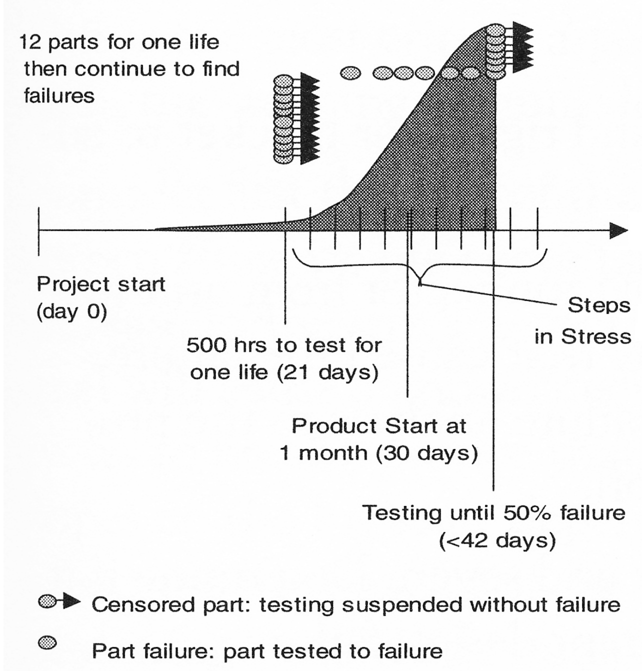

Figure 1 - Example of step stress test [3]

E. *Automated Testing*

Today, there are many testing tools on the market. However, they are considered to be fairly primitive with regard to be quality required [4]. They can test software with some degrees of automation so that the test engineers can devote more time to solving problems in high-risk areas, but their automation is limited for example in these points:

The test scripts often need debugging.

None of them can complete an entire test process independently. They implement processes that may not be consistent with the organization's software design. [8], [9]

The reverse engineering process is separated from the test script generation. One cannot test a product before understanding it. Testing a new product under development is always a learning experience for the test engineers. The time and effort involved depends on the complexity of the product and the experience of the test engineers. To a great extent, one of the benefits of using an automated tool is to prevent the test engineers from spending too much time learning a new product [10].

With the automated software testing tool, testers do not need to write test script by hand or by recording test scenarios. The software testing process will work with the least amount of human interaction [1], [7].

F. *Standards For Software Testing*

Over the years a number of types of document have been invented to allow for the control of testing. They apply to software testing of all kinds from component testing through to release testing. Every organization develops these documents themselves and gives them different names, and in some cases confuses their purpose. To provide a common set of standardized documents the IEEE developed the 829

Standard for Software Test Documentation for any type of software testing, including User Acceptance Testing.

This White Paper outlines each of the types of document in this standard and describes how they work together. There are eight document types in the IEEE 829 standard, which can be used in three distinct phases of software testing:

1) Preparation Of Tests
   - Test Plan: Plan how the testing will proceed.
   - Test Design Specification: Decide what needs to be tested.
   - Test Case Specification: Create the tests to be run.
   - Test Procedure: Describe how the tests are run.
   - Test Item Transmittal Report: Specify the items released for testing.
2) Running The Tests
   - Test Log: Record the details of tests in time order.
   - Test Incident Report: Record details of events that need to be investigated.
3) Completion of Testing
   - Test Summary Report: Summarize and evaluate tests.

II. Proposal of Automated Accelerated Step Stress Testing

Design of our automated step stress test model is based on six steps, which are captured in Fig. 2. It should be noted, that this model captures the automated software testing in the phase of regression testing, whereas our model focuses on stress testing. This model served as the basis for our stress testing sequence.

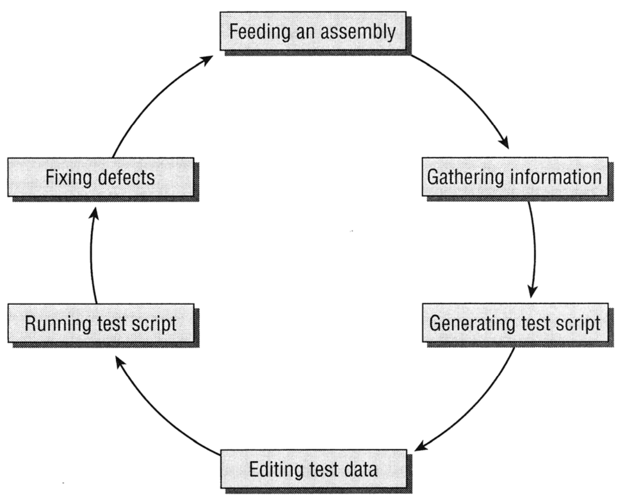

Figure 2 – The base six steps of the automated software testing [1]

Acceleration of stress testing has been reinforced by using of step stress testing. As captured in Fig 1, it builds upon the original stress testing and adds additional life cycle where stress values are increased by 10% every 10% of the life cycle. This procedure is repeated until it reaches 50% of failed parts.

A. *Proposal of modified phases of step stress test*

The first step of our proposal is captured in Fig. 3 - the model of basic step stress test. This proposal is modeled as an UML Activity diagram with a certain degree of abstraction. It captures the initialization and iterations of step stress test. It should be noted that this diagram includes simple, but also complex activities. The test is brought to the end, if in the iteration are more than 50% of failed parts, or even if there is at least one failed part in a fully censored test.

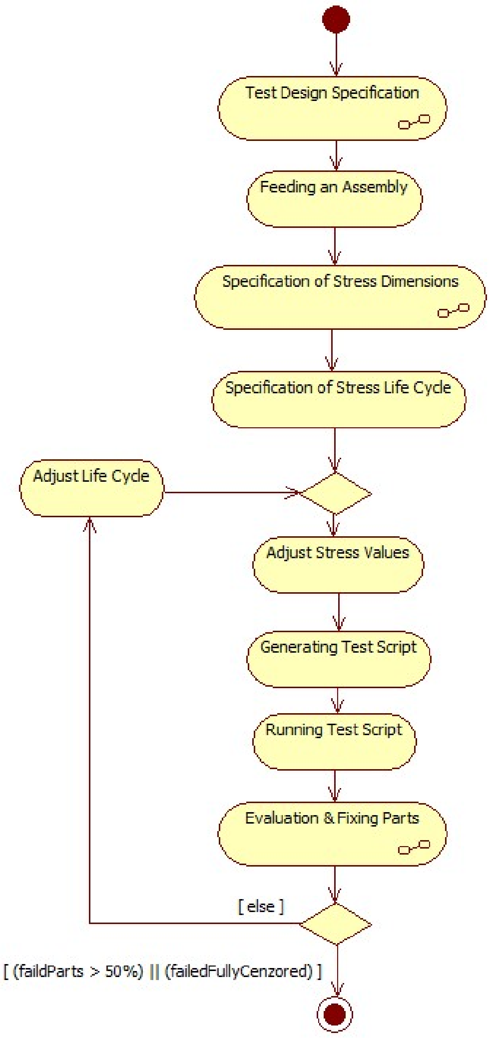

Figure 3 - The proposal of modified phases of automated testing

B. *Proposal of automated stress testing*

The next step is proposal (Fig. 4), which captures the time sequence of the automated step stress testing mapped for each phase of testing methodology IEEE 829. This was slightly modified for greater clearness and simplicity. The phase Test Plan includes a procedure, which describes how the testing will proceed and it is captured by modeled sequence diagram as a whole. In the diagram we use the name Test System that better reflects this phase in terms of process automation. We also merged phases Test Log and Test Incident Report to the Test Execution phase, mainly because of phases modeling difficulty.

After obtaining the necessary information, the test system passes to the phase Test Design Specification, where are defined stress variables, as well as the length of the life cycle and test sample preparation.

Subsequently, the test system begins performing cycle. The first iteration cycle is fully censored test, in which are the Stress Values initialized to the initial value. Another step is to generate test scripts as well as its subsequent execution. After the execution, the individual results are evaluated. It should be noted, that in the first iteration of the test should not come to the mistakes, in consideration of the fully censored tests character.

Further iterations of step stress testing occur almost identical. They are different in length of the life cycle, which is reduced to 10% of the original fully censored test length. In all following iterations stress values are also increases by 10% of the original value. If there is a failure of individual test parts, they will be corrected and recorded in the Test Report Summary.

The test system will interrupt test proceeding, if more than 50% of the parts fail. In the end, results are evaluated and summarized in the Test Summary Report.

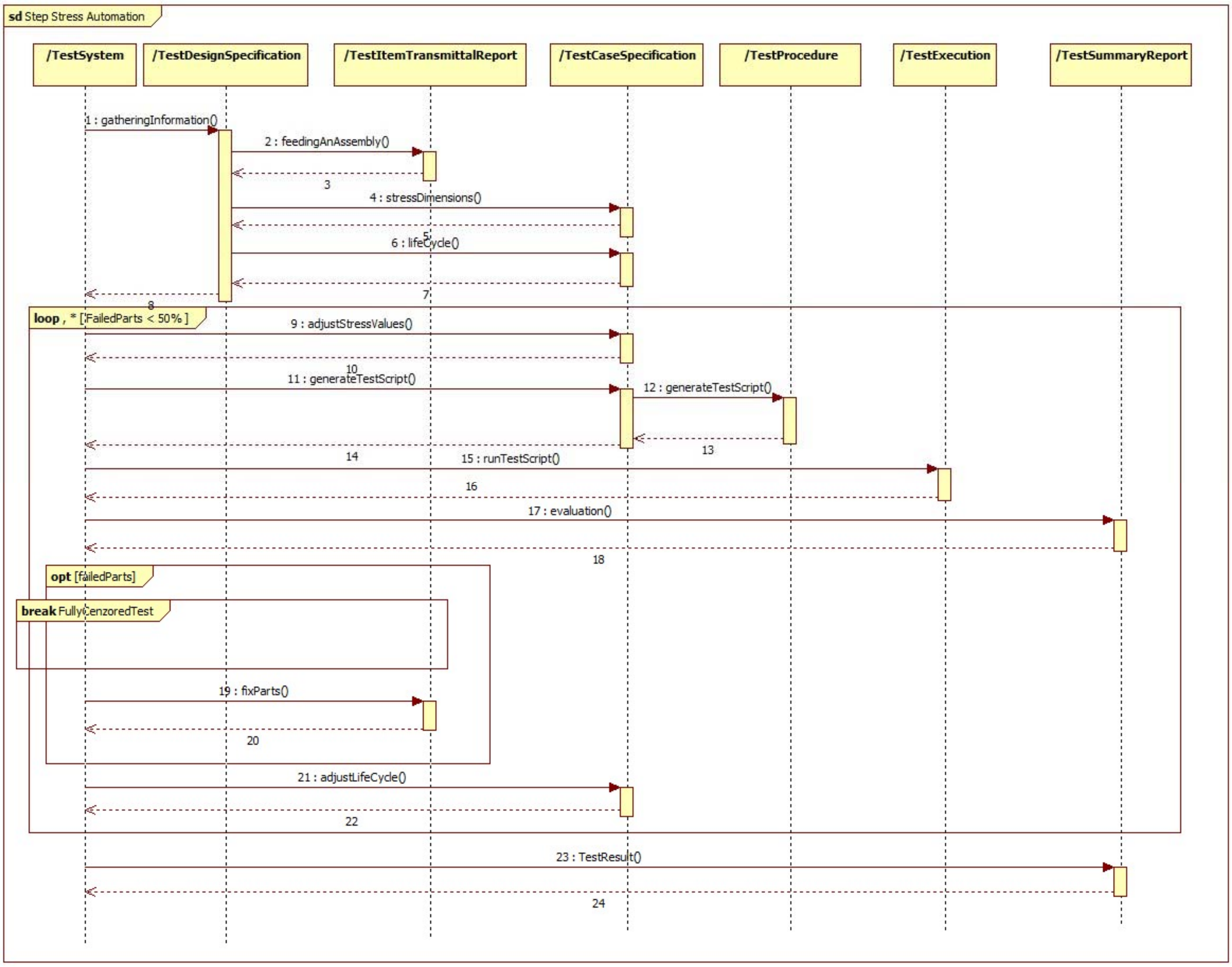

Figure 4 - Proposal of automated stress testing review.

C. *The state model*

The third step of our proposal is modeling of states together with identification of events that activate transitions (captured in Fig. 5). A set of states is captured as a synchronous sequence. This is mainly due to the complexity and clarity over asynchronous model. Because of the degree of abstraction of our model, we have not captured and described internal actions and activities in individual states.

A set of previous standard states, according to standard IEEE 829 is captured as initial state, named PreviousStandardStates. This complex state occurs as the first state in our proposed testing process. When the event gainedSpecificationOfRelevantData occurs, testing process passes to the state Sampling, where the input action is invoked. The role of this action is to describe the parts to be tested. The state has also the task of filling the test sample, repairing the broken parts, etc. At signalizing the output event, the output activity occurs moving the sample of the tested parts to actual iteration of the test.

By the event TestSampleReady the process passes into the state TestParameterSpecification. The role of the input action is to identify the iteration in which the testing process is present. This is mainly because of identification of the initial fully censored test. Some of the internal actions of this state are the life cycle adjustment, stress values adjustment and initial stress dimension specification. The task of the output action is the verification of the changed values in this test iteration. This event occurs after the innovation of adjustedValues action that moves the process to TestProcedureDefinition state.

After reaching this state, the loadPreviousTestScript action is executed. Previous test script is loaded for its further modification in this iteration. This state is specific, inter alia, by the generating of the test script to be loaded and carried out in the state TestExecution after sending a message on the generated test script. It should be noted that as an output action the verification of the generated test script must be carried out.

It should also be noted that this state TestExecution is a combination of the two document types of the IEEE 829 testing methodology. The main task of the output action is to store the data obtained from the test script.

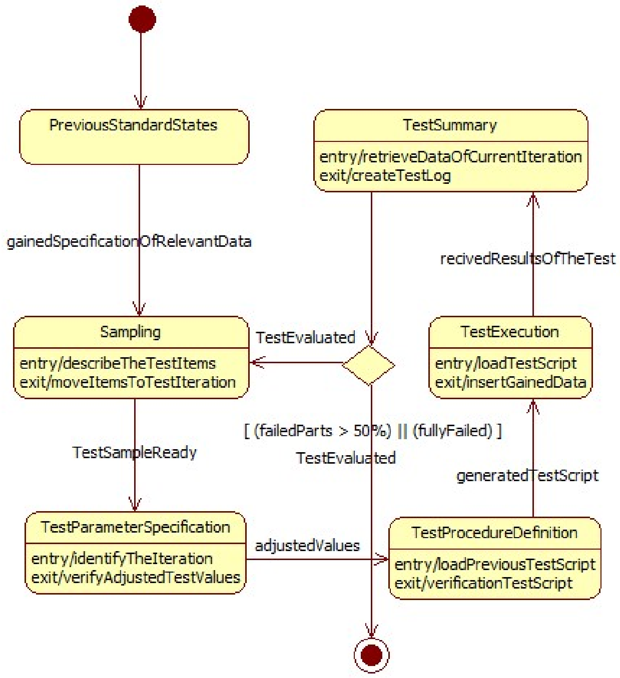

Figure 5 - Global State machine diagram

After receiving the test results, which represent the output event, test proceeds to TestSummary state. This state with the input action gains data from the current iteration. This state also brings together all important information about current iteration of the test and records it for the later whole test summary. The result is a test log, which is also a part of the output action.

The event that initiates the next iteration of the test is TestEvaluated. The whole process repeats until more than 50% of the parts from the test sample fail or if in the fully censored test iteration a failure occurred. These conditions are based on standard methods of the step stress testing.

D. *Interaction between different phases of our model*

The following brief interaction overview diagram in Fig. 6 captures the occurrence of interactions between different phases of our proposed step stress testing model. In contrast with the sequence diagram captured in Fig. 4, which realize the top testing use case as a whole, the proposed interaction overview diagram contains references to partial sequence diagrams and interaction occurrence, which implement the use cases. It should be noted that the partial use cases were not modeled because it is not so cardinal for our model proposal. From the problem domain aspect, the model is better described with the already proposed UML diagrams.

III. Conclusion

The aim of this article is to design the base model of automated step stress testing for control systems. Our proposal is based on the modified basic steps of automated software testing. As a type of accelerated stress testing, we chose the step stress testing to use. Our design was captured by various diagrams in UML 2.0.

References

[1] L. Kanglin, W. Mengqi, "Effective Software Test Automation". Alameda, USA : SYBEX, 2004. 408 pp. ISBN 0-7821-4320-2.

[2] D. Kundu, N. Balakridhnan, N. Kannan, T. NG, Step Stress Model. Kanpur, India : IIT Kanpur, 2006. [cit. 29.1.2010]. [Online]. Available: http://home.iitk.ac.in/~kundu/seminar25.pdf.

[3] A. Porter Accelerated Testing and Validation. Burlington, USA : Newnes, 2004. 242 pp. ISBN 0-7506-7653-1.

[4] G. Tassey The Economic Impacts of Inadequate Infrastructure for Software Testing . Collingdale, PA : DIANE Publishing Co, 2003. 300 pp. ISBN 0-7567-2618-2.

[5] N. H. Criscimagna. Accelerated Testing, [Online]. Available: http://www.theriac.org/DeskReference/viewDocument .php ?id=61 Accessed: 2010-05-20, 2010

[6] P. Vassiliou, A. Mettas Understanding Accelerated Life-Testing Analysis, [Online]. Available: ftp://ftp.estec.esa.nl/ pub3/ tos-qq/qq/RAMS2003ConferenceTutorial/Tutorials/2Dnotes.pdf Accessed: 2010-05-03, 2003

[7] O. Moravcik, P. Schreiber, P. Vazan "The proposal of more precise method for lot size determination in batch production". In: Comec 2008 : V.Conferencia Científica International de Ingeniería Mecánica. Del 4 al 6 de Noviembre de 2008, Cuba. - Santa Clara : Facultad de Ingeniería Mecánica Universidad Central "Marta Abreu" de Las Villas, 2008. - ISBN 978-959-250-404-2

[8] J. Kunstar, I. Adamuscinova, Z. Havlice " The use of development models for improvement of software maintenance", Acta Universitatis Sapientiae, Informatica, 1, 1, 2009, pp. 45-52, ISSN 1844-6086.

[9] A. Trnka " Výber modelu testovania aplikácií dátových skladov. Choice of datawarehouse applications testing model." In: Journal of Information Technologies. 2008 - ISSN 1337-7469.

[10] M. Stremy, A. Elias " Virtual Laboratory Communication". In: Annals of DAAAM and Proceedings of DAAAM Symposium. - ISSN 1726-9679. - Vol. 20, No. 1. Annals of DAAAM for 2009 & Proceedings of the 20th international DAAAM symposium "Intelligent manufacturing & automation: Focus on theory, practice and education" 25 - 28th November 2009, Vienna, Austria. - Vienna : DAAAM International Vienna, 2009. - ISBN 978-3-901509-70-4, s. 0139-0140

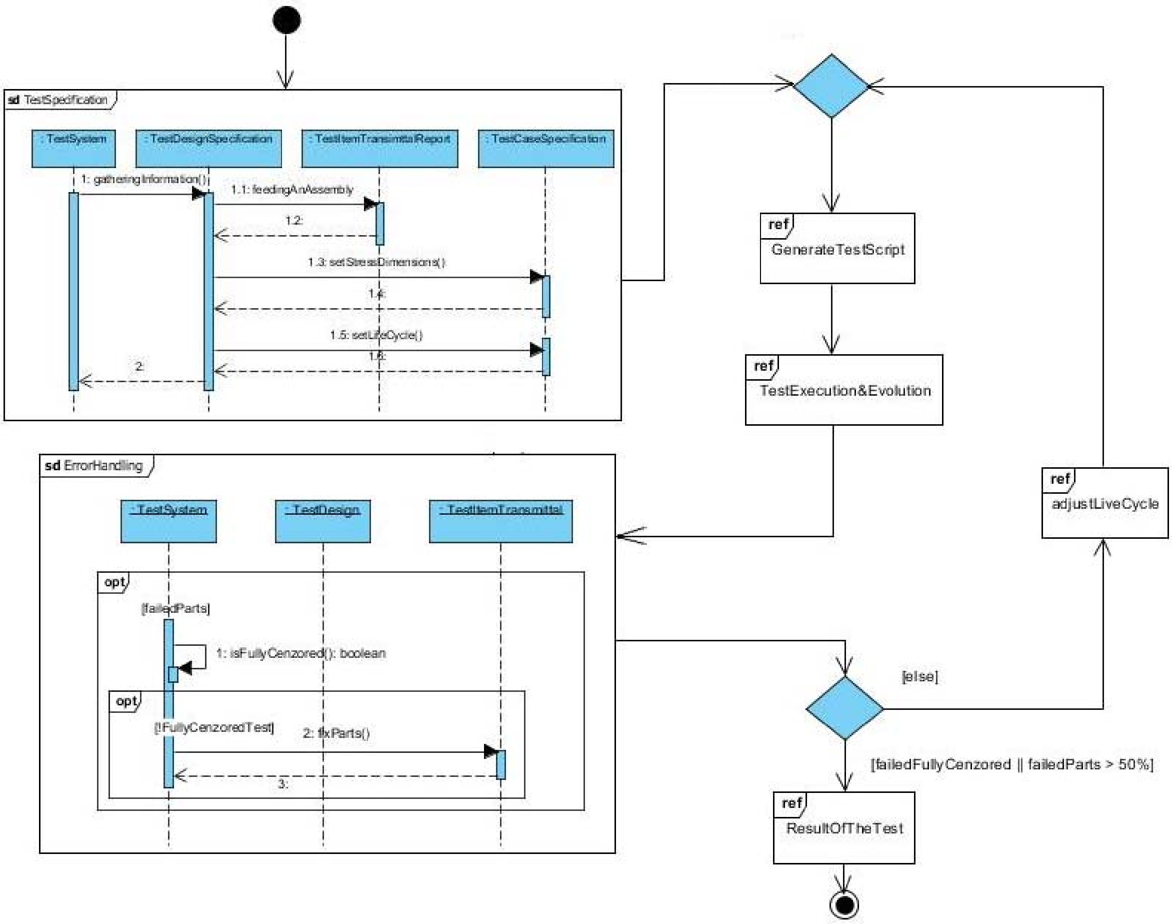

Figure 6 - Interaction overview diagram of automated stress testing