2010 Seventh International Conference on Information Technology

Organizational Testing Management Maturity Model for a Software Product Line

Etiene Lamas[1], Érica Ferreira[1], Marcos Ribeiro do Nascimento[1],
Luiz Alberto Vieira Dias[1] and Fabio Fagundes Silveira[2]
[1]*Brazilian Aeronautics Institute of Technology, ITA*
São José dos Campos, São Paulo, Brazil
[2]*Federal University of São Paulo, UNIFESP*
São José dos Campos, São Paulo, Brazil

Abstract

This paper presents a framework entitled Organizational Testing Management Maturity Model (OTM3). The proposed framework is a set of structures to support the development and testing of Software Product Lines. This set follows the Experimental Software Engineering concepts. OTM3 is a framework for interactive, incremental, and continuous models. It shall provide the information for the organization, and a method to identity, establish and keep the capabilities demanded by test maturity models. This is achieved through: patterns, measures, controls, and software engineering best practices.

Keywords: Software Testing, Maturity Models for Software Testing, Software Product Line, Experimental Software Engineering, Process Management with Testing.

1. Introduction

The Organizational Testing Management Maturity Model (OTM3) is under development and it uses Software Testing (ST) techniques for processes measurement improvements in Software Engineering (SE), focusing on: i) direct experience on issues relating to the measurement of ST, ii) continuous improvement of software generated product maturity, iii) use of technical specification, and iv) design using an adaptation of the method of Goal Question Metric (GQM) [1]. The OTM3 was designed to allow the development of qualitative and quantitative metrics in an iterative and incremental approach.

Software Product Lines (SPL) [2] were used in this paper as an approach to a set of structures, in order to adapt the Experimental Software Engineering (ESE) concepts, focusing in drawing, verification and validation of ST for each SPL [3].

In order to apply OTM3 in a practical environment, its concepts were used in the Amazonian Integration and Cooperation Project for Modernization of Hydrological Monitoring (ICA-MMH) [4]. This project under development at ITA, provides a management organization capable of: i) deal with the SE application architecture in 3-tier web; ii) use SE integrated management systems, and iii) design and construction Model Engineering Platform (MEP) for data collection (hardware and embedded software).

2. Related Works

There are four relevant methods for conducting experiments in the field of SE: scientific, engineering, experimental, and analytical [5]. The most appropriate one for the framework OTM3 is the experimental.

The Quality Improvement Paradigm (QIP) is linked to the concept of the Experience Factory [6] which is the set of tools for storage, modification, and withdrawal of project information packed. Another instrument connected to QIP approach is GQM [7]. This approach provides the process of improving the measurement model based on layers.

According to Brykczynski, Meeson, and Wheeler [8], many defects found in testing are directly traceable to requirements and design flaws, which could have been detected earlier. Defects that are found out later in the project are not only costly to fix, but also the effort put on the software design and development are lost. Thus, the inspection on documents is important, right from the start of the project.

Related inspection methods and reading techniques, the requirements inspection Test-Case Driven (TCD), is an efficient and effective inspection technology, and a "variant" to ideas behind Test-Driven Development (TDD), Checklist Based Reading (CBR), Two-Person Inspections (TPI), and Perspective Based Reading (PBR) [9, 10].

3. Model OTM3

The framework OTM3 is composed of categorization, depicted in Figure 1, with: i) Columns for processes domains of ST: Operational, Collaborative, Decision Support; and ii) Map Processes,

978-0-7695-3984-3/10 $26.00 © 2010 IEEE
DOI 10.1109/ITNG.2010.134

that contains: standards, measure, control, and continuous improvement sub-plans, and iii) in each sub-plan, a PDCA cycle [11] is used: (P)lan, (D)evelopment, (C)ontrol and corrective (A)ctions.

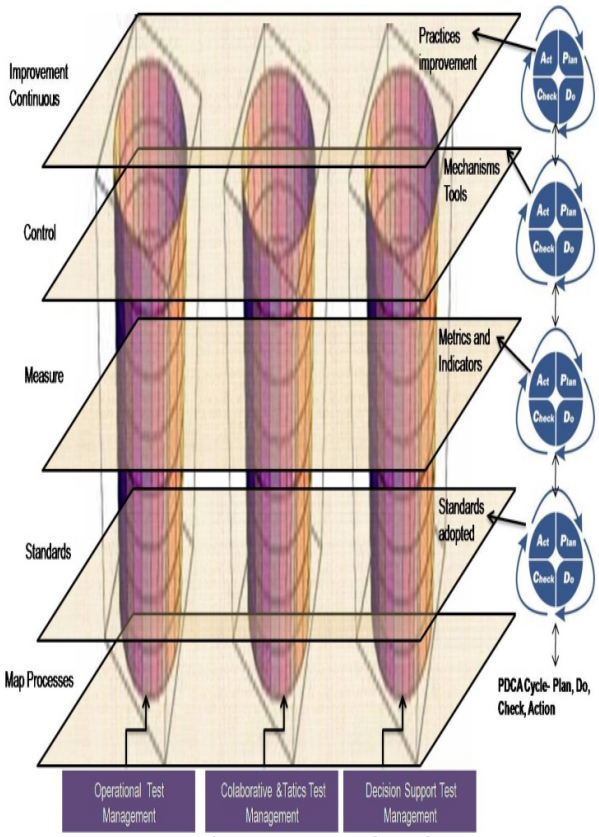

Figure 1. Framework OTM3

The framework OTM3 considered domains are: i) Operational Testing Management – define a set of operational base ST processes that allow to follow the ST workflow; ii) Collaborative & Tactics Test Management - define a set of ST processes for integration and tactic inter-operability allowing the task consolidation and multi-programming activities for ST; and iii) Decision Support Test Management - define a set of processes for information treatment and analysis, and extraction of strategic patterns for decision making over of software testing management

Key Performance Indicators (KPIs) represent the outcome through measure by a metric Goals Questions Indicators Measures (GQIM) [12].

The framework allows the diagnostics for capability maturity of the organization through the application of a questionnaire. This questionnaire is interactive and incremental as well as qualitative and quantitative (quali-quant), for each management domain of process testing.

The OTM3 was based on the standard SPEM 2.0 [13] and it is composed by a set of processes in maps that constitute the main phases of ST, namely: Planning, Construction, Test Execution, Test Finalization and Approval.

4. Construction

The Construction Phase is aimed at test developments. The basic objectives of this phase are: Develop and Review Strategies from Scenarios. Test Case Scenarios aim to prepare the environment for the Test Execution Phase. The Construction Phase also realizes Measurement and Analysis (Figure 2).

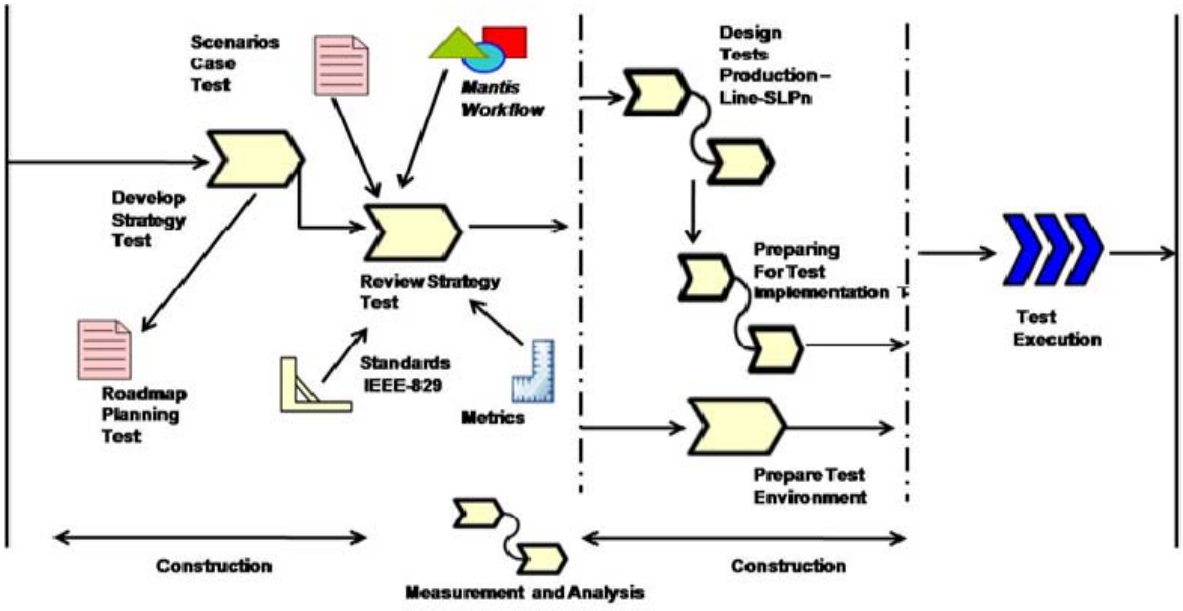

Figure 2. Construction

5. Test Execution

As shown in Figure 3, Test Execution includes the following steps: i) testing techniques activities (i.e. black and white-box testing and Performance testing) are applied; ii) Architectural Integration Testing level; iii) System Testing; iv) the Acceptance Testing; and v) Execution Phase is considered complete, then it will occurs the next phase of model OTM3, which is the Test Finalization Phase.

It is very important to emphasize that the activities in the Test Execution Phase always apply the Regression Test Selection (RTS) technique. Regression testing means rerunning test cases from existing test suites to build confidence that software changes have no unintended side-effects.

Each testing activity performed by the Test Execution Phase contains specific tools to be used.

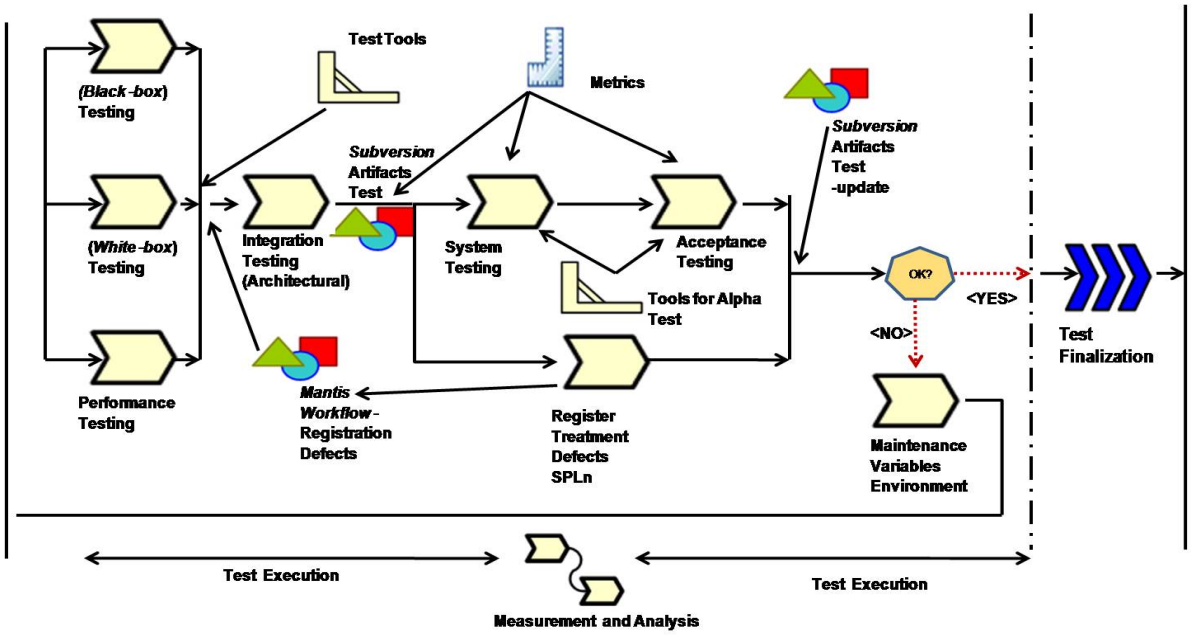

Figure 3. Test Execution

When implemented the tools that are suggested to be used correlated to this techniques are, for example: black-box - Selenium [14], JUnit [15]; and white-box - JaBUTi [16].

The Mantis (MT) tool is used to manage the process defects [17]. MT tool supports native integration with the Subversion (SVN) tool [18]. The SVN tool provides further support to the activities of changing control and continuous integration, ensuring that changes to the project to be built, are tested and reported in a short time. MT tool also achieves integration with the TestLink tool [19]. TestLink is a tool for executing and tracking test cases, organizing them into test plans.

The existence of these three integrated tools adds project-level reporting, analysis and management capabilities, especially when used in conjunction with requirements traceability information. It also allows full control on system defect management.

It is also important to mention that in the Test Execution Phase, quali-quant indicators and metrics are applied. Defined metrics will identify risk areas that requires more testing, provide a solution to potential problems, and identify areas for process improvement.

Before the Test Execution Phase is considered completed, two important test activities are performed: the System Testing and Acceptance Testing.

6. OTM3 within ESE

SE like other disciplines (i.e. Physics, Medicine, Manufacturing), requires the cycle of model building, experimentation, and learning. SE is a laboratory science. The researcher's role is to understand the nature of the processes, products and the relationship between the two in the context of the system. The practitioner's role is to build "improved" systems, using the knowledge available [20, 21, 22, 23].

Study is an act to discover something unknown or for testing a hypothesis. Studies can be experimental and observational. OTM3 identifies and helps the resolution of potential problems. Studies Classification are primary and secondary. A good example is systematic review, where OTM3 metrics data collection helps predict the long-term direction and scope for an organization and enables a more holistic view of

business and identifies high-level goals in each SPL [20, 21, 22, 23].

The components of the primary studies (four parameters were used, based upon the GQM template): 1) object of study; 2) purpose: i) characterize; ii) evaluate; iii) predict; iv) control; and v) improve; 3) focus – based on the standard ISO/IEC 25001 [23]; and 4) stakeholder point of view [20, 21, 22, 23].

The new standard ISO/IEC 25001:2007 [24] replaces the current ISO/IEC 9126 series and set the basis for OTM3 formulation on software quality testing, defect detection or prevention capability.

The GQIM paradigm [11], which is heavily used in SE for defining metrics, is the technique that supports this research. While the need for metrics has been recognized, implementation of structured measurement programs is lagging, especially in the ST area. OTM3 could measure the effectiveness of a Test Process. Efficient Test Process measurements are essential for managing and evaluating the effectiveness of a Test Process.

The goals of software production are: i) high quality product; ii) within budget constraints; and iii) a specified deadline. The Final goals are: i) cost reduction; ii) meeting deadlines; and iii) product quality improvement.

In this experimental study the main objective was: "What is the effect of the software testing techniques in a SPL strategy on product reliability and stability, given an environment of expert programmers in a new domain, with tight schedule constraints?"

Other questions: "In what order should the requirements be tested, especially when time is short?"; "Does the software response to an action in the time defined on the requirements?"; "How to test applications in SPL domains using an OTM3 approach?"; "What are Testing Coverage (TC) criteria to be applied for applications in each of the SPL domains?"; "How to measure the TC in OTM3 about non-functional requirement?"; "How are non-functional requirements descriptions used for Test Cases generation in OTM3?"; "How to reduce the Test Case set without decreasing the TC in OTM3 (i.e. eliminating redundant Test Cases)?"; "OTM3 approaches support the software maintenance (regression testing), but how to evaluate its maintainability?"; "How are the defects distribution per phase or per type of defects?"; "What are the origins of these defects?"; "How to avoid these types of defects in future projects using an OTM3 approach or other strategy to support the ST?"; "How many defects were detected using the OTM3 approach?"

Questions that cannot be answered by this experimental study, out of OTM3 context: "How effective is the training? Are the subjects really following the techniques?"

According to Pfleeger [25], one of the challenges would be to understand the chances where, under certain conditions, a particular tool or technique could contribute for the improvement of software development. In this sense, experimentation can be considered as determinant for the identification of the feasibility and effectiveness of the techniques applied to SE [26].

The purpose points out for the intended use of the measures: i) characterization - what is the current state of the object under study? ii) evaluation - is the current state good or bad? iii) prediction - how the quality focus can be predicted? and iv) improvement - is it possible to find cause-effect relationships involving the quality focus? [20, 21, 22, 23].

7. Strategy for the proposed experiments

As already stated the adopted strategy in the project was Experimentation Model; it involves an experimental study in two dimensions: 1) the maturity of software product (A); and 2) ST process improvement (B). The experimental primary study was based on the following scenarios: driven by understanding; predominantly qualitative analysis; qualitative or quantitative study; pure qualitative study. The study executed in laboratory (in vitro), has involved the project team, it were used to test hypotheses or get information on the study field.

To improve the software product for each SPL in the project, the aim of the experiment (A) was to propose the effectiveness and efficiency of software evaluation. Primary and secondary GQIM metrics probably could discover that failures could be traced back to defects in the requirements specification. The evaluation is performed by comparing the secondary GQIM indicators generated in each RTS.

The strategy adopted in the ICA-HMM project was to provide software development in a model driven architecture (MDA) approach, integrated with a Unified Process based on SPL components for each subsystem, according to Linden et al. [3].

The metrics and indicators from GQIM are being applied on the main phases of the operational software testing the OTM3 Model, the principal primary metrics observed on the 1st SPL as shown in Table 1.

Table 1. Project statistics SPL 1

Criteria Types	Project Statistics
Classes	(Entity, Manageable) =13, Enumeration = 1 and Controller = 4
Services	5
Value Objects	Value Object = 8 and Array = 4
State Machines	Machines=5, States=34 and Transactions = 41
DataBaseEntities	13
Object patterns	10
Lines Code	104,660 (.java, .xml, .css, .js, .properties, .sql and .xhtml); 75,011 excluding blank lines and comments 18,230 comments
IHM	CRUD = 8 and Customizes = 5

An additional a number of test cases between RTS 1 and 2, were due to a better specification of software components manually developed, and have been done by the development team. Assistance in the area of project

quality and the completion of incomplete test cases (blocked) is presented in Table 2 for both RTS.

The data presented in Table 2 represent the functional tests performed on the SPL 1 on the initial phase of the proposed framework.

Table 2. Comparison SPL 1 - RTS

Primary Metrics	Unit	Qty. SPL1 – RTS 1	Qty. SPL1 – RTS 2
Number of Use Cases	Cardinal number	8	8
Number Test Cases	Cardinal number	170	235
Number Test Cases - passed	Cardinal number	79	169
Number Test Cases – failed	Cardinal number	16	25
Number Test Cases – blocked	Cardinal number	75	41

The derived or secondary metrics based on GQIM, present in Table 3, should provide product quality attributes, such as to control the software process. Product metrics can be used for general predictions or to identify anomalous software components.

Table 3. SPL 1 - derived Metrics

Secondary Metrics	Indicator	Value SPL1 – RTS 1	Value SPL1 – RTS 2
# Tested Use Case/ # Total Use Case	Testing coverage Percent Use Case (PUC)%	6/8= 0,75 [75%]	8/8= 1 [100%]
# Tested Test Case/ # Total Case Test	Testing coverage Percent Case Test (PCT)%	95/170= 0,56 [56%]	194/235= 0,83 [83%]
# Defect Test Case/ # Total Case Test	Testing Find Defects (TFD)% (per defect case test)	16/170= 0,09 [9%]	25/235= 0,11 [11%]
# Defect Test Case removed (RTS 1-2)/ # Total Defect Case Test (RTS 2)	Effective Removal of Defects (ERD)	N/A	4/25= 0,16 [16%]
# Blocked/fail Test Case (RTS 1) / #(Blocked/fail Test Case (RTS 2))	Effectiveness of Detection of Defects (EDD)	N/A	(16+75)/ (91+66)= 0,58 [58%]
# ERD / # Total week Product lines	Defects Removal per week	N/A	0,16/4= 0,04 [4%]
# Tested Test Case/ # Total week effort to line product	Density of the Efficiency per week or per members team Test (DET)	95/6= 0,56 [56%]	194/4= 0,56 [56%]

The qualitative analysis of secondary metrics that identify the characteristics of efficiency and effectiveness of the software product provides better process control and prediction. Both can influence the decision-making and management control of product quality and process of software development, adhering to standard ISO/IEC 25001:2007 [24].

The purpose of experiment (B), software testing improvement, is still under development and is aiming at improving the process testing for each SPL and to evaluate the effectiveness and efficiency of the TCD inspections, when it comes to find the major faults in requirement specifications. The evaluation should perform a comparison of TCD inspections for the well tested and widely used technique CBR.

8. Discussion and Conclusion

The present proposed framework is under development. Some experiments were performed for a case study from a real project on the beginning of Test Execution phase. So far the results are encouraging, as seen on Tables 1, 2 and 3. Further experiments are on the planning phase.

It should be noted that the test artifact tracking relating to the Use Cases must be inspected and verified.

The revision by PBR must be planned, due to eventual activities deficiencies.

When using ESE the quality team must prepare in advance a strategy for its application control and evaluation that ensure the: i) maturity of the testing process; ii) preparation of a checklist to inspect for SPL activity; and iii) proper use of the proposed framework.

Measurements are collected and analyzed weekly, and the progress evaluated. Developers must adapt their behavior monthly. Finally the OTM3 is not geared to mathematically prove the correctness of the software; the testing process aims to provide the best quality of software products.

9. References

[1] Basili, V.R. “Software Modeling and Measurement: The Goal Question Metric Paradigm”, Computer Science Technical Report Series, CS-TR-2956 (UMIACS-TR-92-96), University of Maryland, College Park, MD. 1992.

[2] Linden, F.J., K. Schimid, and E. Rommes, “Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering”, Springer, 2007.

[3] Software Engineering Institute (SEI), “Software Product Lines”, Carnegie Mellon University, Pittsburgh, PA, http://www.sei.cmu.edu/productlines/, Last access: Oct 23, 2009.

[4] Pessoa, J.A. Dias, L.A.V. Cunha, A.M. “A Desktop Environment for River Hazards Monitoring”, Proceedings of the Sixth International Conference on Information Technology: New Generations, ITNG 2009, Las Vegas, NV, USA, April 27-29, 2009.

[5] Wohlin, C., P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, “Experimentation in Software Engineering: an introduction”, Kluwer Academic Publishers, USA, 2000.

[6] Basili, V., G. Caldeira, and H. Rombach, “Experience factory”, Encyclopedia of Software Engineering, Ed. J.J. Marciniak, Vol. I, Wiley, 1994, pp. 469-476.

[7] Seaman, C., “Qualitative Methods in Empirical Studies of Software Engineering”, IEEE Computer, Vol.25, No.4, July/August 1999.

[8] B. Brykczynski, R. Meeson, and D.A. Wheeler, “Institute for Defense Analyses Software Inspection: Eliminating Software Defects”, Proceedings of the 6th Annual Software Technology Conference, Salt Lake City, UT, April 15, 1994.

[9] N. Fogelström and T. Gorschek, “Test-case Driven versus Checklist-based Inspections of Software Requirements - An Experimental Evaluation”, 10th Workshop on Requirements Engineering, Toronto Canada, May 17-18, 2007, pp. 116 - 126.

[10] A. Aurum, H. Petersson, and C. Wohlin, "State-of-the-Art: Software Inspections after 25 Years", Published in Software Testing, Verification and Reliability, Vol. 12, No. 3, 2002, pp. 133-154.

[11] Walton, M. “Deming Management at Work”. Clearwater, FL, U.S.A: The Berkley Publishing Group, 1991.

[12] Park, R.E., W.B. Goethert, and W.A. Florac, “Goal-Driven Software Measurement – A Guidebook”, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1996.

[13] Software Process Engineering Metamodel - SPEM 2.0 available at http://www.omg.org/spec/SPEM/2.0/ Last access: Oct 23, 2009. .

[14] Selenium, available at http://seleniumhq.org/, Last access: Oct 23, 2009.

[15] JUnit, available at http://www.junit.org/, Last access: Oct 23, 2009.

[16] A.M.R. Vincenzi, W.E. Wong, M.E. Delamaro, J.C. Maldonado, “JaBUTi: A coverage analysis tool for Java Programs”, Simpósio Brasileiro de Engenharia de Software, Manaus, AM, XVII Simpósio Brasileiro de Engenharia de Software - SBES'2003, 2003, pp. 79-84.

[17] Mantis, available at http://www.mantisbt.org/, Last access: Oct 23, 2009.

[18] Subversion, available at http://subversion.tigris.org/, Last access: Oct 23, 2009.

[19] TestLink Community, “User Manual”, available at http://testlink.sourceforge.net/docs/testLink.php, Last access: Jan 10, 2010.

[20] S.L. Pfleeger (SLP), Rand Inc. “Evaluating Software Technology” - Tutorial at SBES’2002 and Software Engineering Thesis Workshop talk at SBES’2002.

[21] V.R. Basili (VRB), UMD/USA “The Role of Experimentation in Software Engineering: Past, Present, Future” - Keynote speaker at ICSE 18.

[22] F. Shull, J. Carver and G.H. Travassos (FS/JC/GHT) “An Empirical Methodology for Introducing Software Processes”, 8th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-9), Vienna, 2001.

[23] G.H. Travassos and M.O. Barros (GHT/MOB) “Contributions of *In Virtuo* and *In Silico* Experiments for the Future of Empirical Studies in Software Engineering”, Workshop Series on Empirical Software Engineering, WSESE, ESERNET, 2003.

[24] ISO/IEC 25001:2007, “Software engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Planning and management”, http://www.iso.org/iso/catalogue_detail.htm?csnumber=35724, Last access: Oct 23, 2009.

[25] S.L. Pfleeger, “Albert Einstein and Empirical Software Engineering”, IEEE Computer, Oct., 1999, pp. 32-38.

[26] M.V. Zelkowitz, D.R. Wallace, D.W. Binkley, “Experimental Validation of New Software Technology”, Lecture Notes On Empirical Software Engineering, Chapter 6, World Scientific, 2003, pp. 229-263.