JISTEM - Journal of Information Systems and Technology Management
Revista de Gestão da Tecnologia e Sistemas de Informação
Vol. 9, No. 2, May/Aug. 2012, *pp.265-284*
ISSN online: 1807-1775
DOI: 10.4301/S1807-17752012000200004

PROPOSAL FOR A MEASUREMENT MODEL FOR SOFTWARE TESTS WITH A FOCUS ON THE MANAGEMENT OF OUTSOURCED SERVICES

Angélica Toffano Seidel Calazans
Ricardo Ajax Dias Kosloski
Luiz Carlos Miyadaira Ribeiro Junior
Centro Universitário de Brasília, Uniceub, Brasilia, Brazil
Universidade de Brasília, UNB, Brasilia, Brazil

ABSTRACT

The need for outsourcing IT services has shown a significant growth over the past few years. This article presents a proposal for a measurement model for Software Tests with a focus on the management of these outsourced services by governmental organizations. The following specific goals were defined: to identify and analyze the test process; to identify and analyze the existing standards that govern the hiring of IT services and to propose a Measurement Model for outsourced services of this type. As to the analysis of the data collected (documentary research and semi-structured interviews), content analysis was adopted, and in order to prepare the metrics, the GQM – Goal, Questions, Metrics – approach was used. The result was confirmed by semi-structured interviews. Here is what the research identifies as possible: to establish objective and measurable criteria for a measurement size as the input to evaluate the efforts and deadlines involved; to follow up the test sub-processes and to evaluate the service quality. Therefore, the management of this type of service hiring can be done more efficiently.

Keywords: Test process, hiring management, outsourcing, metrics, and measurements

---

Manuscript first received/*Recebido em* 22/09/2011 Manuscript accepted/*Aprovado em*: 12/04/2012

Address for correspondence / *Endereço para correspondência*

Angélica Toffano Seidel Calazans, Doctorate degree in Information Science from Universidade de Brasília (2008) and a master´s degree in Knowledge Management and IT from Universidade Católica de Brasília (2003). A post-graduate degree in Systems Analysis from UDF (1986) and in Client Server Platform (1996). Professor at Centro Universitário de Brasilia, Uniceub/BR, FATECS, Brasília-DF, Phone: 55 061 81167246, E-mail: angélica.calazans@uniceub.br

Ricardo Ajax Dias Kosloski, Professor at Universidade de Brasília – UnB. A post-graduate degree in Software Engineering, from Universidade Católica de Brasília - UcB (2003), a master´s degree in Knowledge Management and Information Technology also from UcB (2005). TiMétricas, Brasília-DF, Phone: 55 061 84063679, E-mail: ricardo.kosloski@metricas.com.br

Luiz Carlos Miyadaira Ribeiro Junior , Associate Professor at Universidade de Brasília, master´s degree in Computer Science from Universidade Federal de São Carlos and a doctorate degree from Escola Politécnica da Universidade de São Paulo (2007). Universidade de Brasília-UnB, Brasília-DF, E-mail: luiz.miyadaira@gmail.com

Published by/ *Publicado por:* TECSI FEA USP – 2012 All rights reserved.

1. INTRODUCTION

High market competitiveness and technological advances have increased the demand for better and better software, produced through predefined costs and deadlines. In turn, such factors as complexity, size, heterogeneity and the dynamism of computer systems have directly impacted the quality of these products. In this scenario, the test process becomes increasingly important due to the fact that its main objectives are product analysis, identification of defects and their possible elimination.

Software tests include the Verification and Validation processes. According to (Melhoria de Processo do Software Brasileiro, 2009), the purpose of Verification is to confirm that each service and/or product of the process or project satisfies the specified requirements while the objective of Validation is to confirm that a product or component will satisfy the intended use when applied to the production environment. The correct implementation of these processes results in economic gains such as: reduction in the levels of software defects, reduction in development costs and in product delivery time and the increase in efficiency of the software development process (Venkatasubramaniarn et Vinoline, 2010).

Despite these gains Juristo, Moreno And Vegas (2004) regard software tests as one of the most costly practices in the development process, which needs to be properly managed in order to avoid resource waste and delays in the software development project schedule, among other possibilities. Models such as COBIT and ITIL emphasize the need for the competent management of all IT resources, whether internal or external. This need is also reflected on the test activities, especially when the outsourcing of this process is considered.

According to Silva, Duarte and Castro (2009), "the outsourcing activity or information technology outsourcing has been showing significant growth rates in the IT services segment." And by taking the test context into account, Venkatasubramanian et Vinoline (2010) affirm that software development organizations are currently beginning to outsource test activities (through the use of test factories), in order to reduce costs and increase the quality and the reliability of software products. This has also been a trend in Brazil, especially among government agencies.

Thus, this paper´s purpose is to define a proposal for a Measurement Model for Tests considering the needs of the outsourcing process management by government agencies. A brief view of the test process is then presented in section 2. The laws, standards and models related to service hiring are briefly described in section 3, such as Law # 8666/93, Normative Instruction # 4 of 2010 and other models. In section 4, the research methodology is presented, and a few of the criteria for the measurement of test services are described in item 5. Measurements for the assessment of the quality of the service provided are shown in section 6 and the ways to measure product quality are in section 7. Conclusion and future papers are found in section 8.

2 TEST PROCESS

Testing software is more comprehensive than reporting impressions and non-conformities. The IEEE829 standard for software tests documentation specifies the way to use a set of documents defined in eight stages for the software tests, and for each stage to potentially produce its own type of document, as shown in Figure 1.

The Test Plan, according to this standard, contains the test's objectives and the global goals, while the Test Design Specification describes in detail and specifies how the Test Plan will be executed. The Test Specification Case describes situations which must be tested and the Test Procedure Specification describes the actions that must be performed by the software for the Test Case to be executed.

As for the Test Log (or evidence), it describes the executed tests, regardless of errors having been encountered or not. The Test Incident Report describes the failures that have occurred during the execution of the tests and, finally, the Test Summary Report (or executive) contains the summary of the test conditions executed, the failures encountered and the desired statistical tabulations.

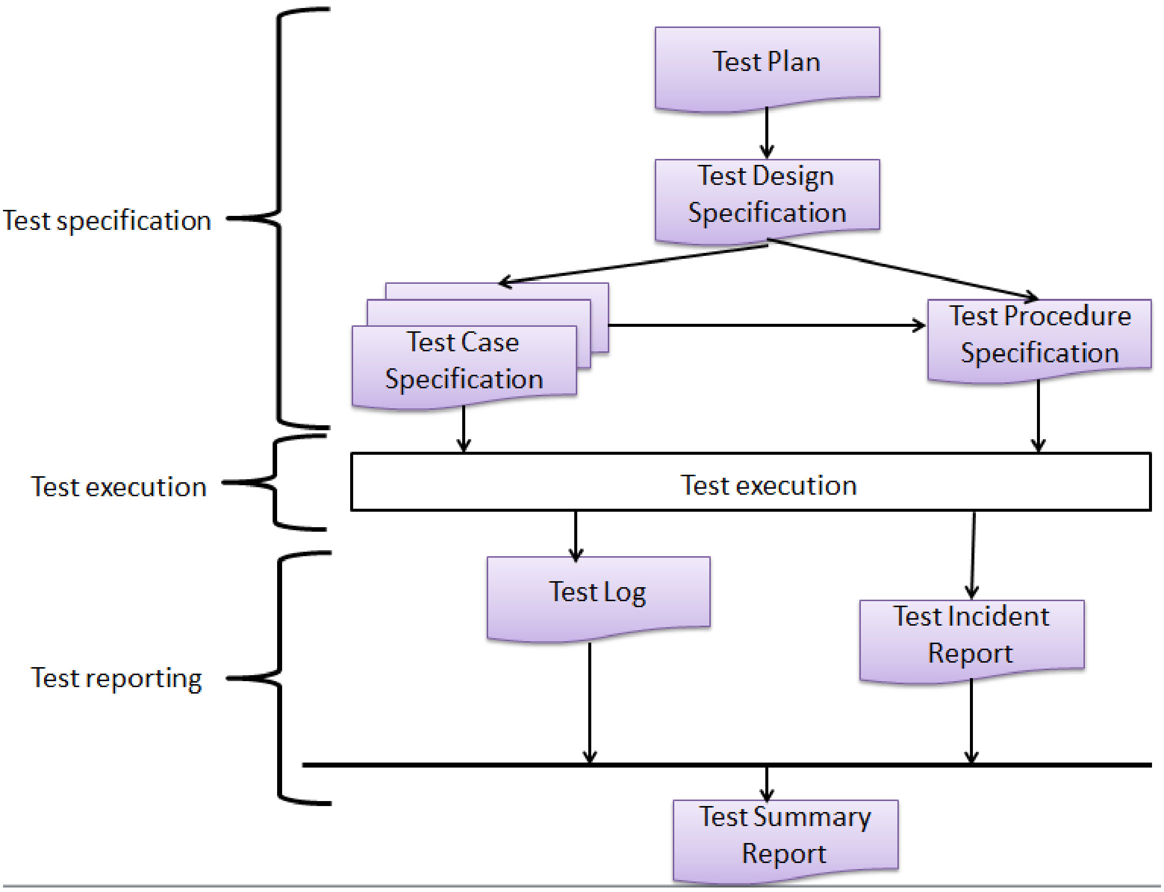


Figure 1 – Standard 829 for Software Tests Documentation

In addition to Standard 829, the "V" model software tests (Pfleeger, 2004) emphasizes the verification and validation activities for the purpose of preventing/detecting failures, and minimizing the risks of the project. For each stage of the software development process, a "V" model introduces one stage or the corresponding test level. In this model, the test planning and specification occur from top to bottom, that is, throughout the software development stages the tests are planned and specified. The execution of the tests occurs in the opposite direction, as can be seen in Figure 2.

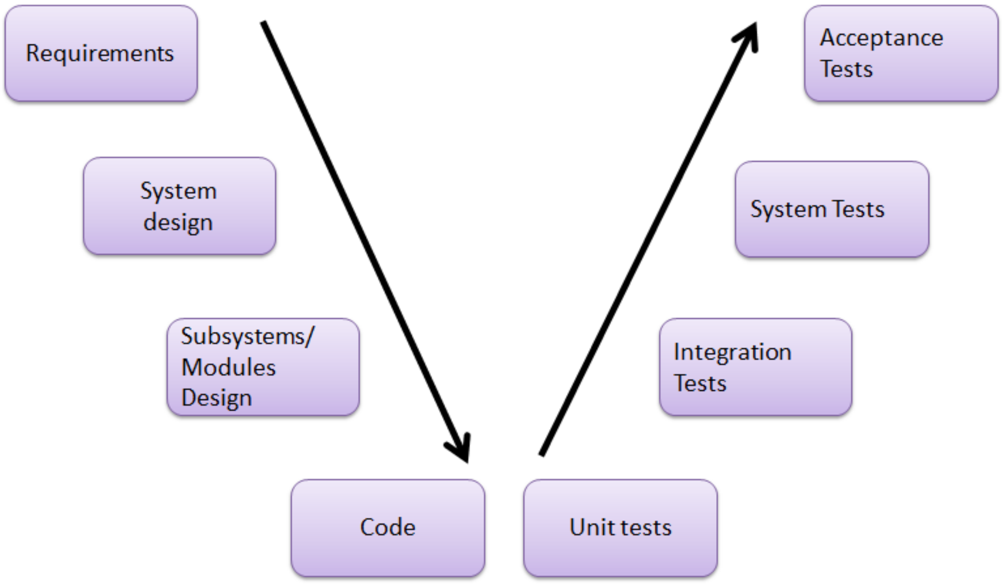


Figure 2 - V Model for software development (Adapted from Pfleeger, 2004)

As a complement to these definitions, Caetano (2008) cites the existence of two test techniques. The Structural Test technique, known as the White-Box Test, where criteria are used for the creation of test cases with the purpose of identifying failures in the software's internal structures. While the Functional Test technique, also known as the Black-Box Test, where criteria are used for the creation of test cases with the purpose of evaluating adherence or compliance of the implemented software in relation to the behavior described in the requirements.

In addition to these techniques, many authors (Sommerville, 2007; Pressman, 2000) identify several types of tests: functionality, usability, performance, security, regressions, load, and configuration, among others.

That is, the test activity involves multiple facets and identifying and defining them for future outsourced services also involves the analysis of already existing standards of hiring and of monitoring this type of service, which are briefly described below.

3 LAWS, STANDARDS AND MODELS LINKED WITH SERVICE HIRING AND MONITORING

In order to do this research, Law n. 8666/93 and Normative Instruction n. 4 of 2010 were briefly analyzed to also identify the applicable aspects of the hiring and monitoring of software test activities. Law n. 8666/93 establishes general rules about bids and administration contracts related to the works, the services, including advertising, purchases, liens, and rentals under the scope of the Powers of the Union, of the States, of the Federal District and of the Municipalities.

In addition to establishing the service hiring methods of Law n. 8666/93, this law also establishes, among other aspects, the need for monitoring of the contract when it cites in its paragraph 67 that

> "The execution of the contract shall be monitored and inspected by a specially assigned Administrative representative, as the hiring of third-parties is allowed, in order to assist them and provide them with information related to this assignment."

Paragraph 1 complements this article citing that

> "the Administration representative will write their own notes regarding the events related to the execution of the contract, by establishing the necessary means to correct existing failures or defects encountered".

The Normative Instruction n. 4 of 2010 (IN04, 2010), of the Logistics and Information Technology Department from the Ministry of Planning, establishes in its article 2, paragraph 20, that:

> "Acceptance Criteria: they are objective and measurable parameters used to verify whether an asset or service provided complies with the specified requirements."

In its article 15, paragraph 3 establishes that the service hiring strategy must contain, among other items:

- establishment of procedures and Acceptance Criteria of the services or assets provided, including metrics, indicators and minimum accepted values;

- previous quantification or estimation of the volume of the demanded services or the number of assets to be provided for comparison and control purposes;

- establishment of the quality assessment methodology and of the suitability of the Information Technology Solution to the functional and technological specifications;

Finally, in article 25, paragraph 3 which describes the monitoring of the services provided, the following items are then specified, among other items

> "quality assessment of the services or assets provided as well as justifications in accordance with the Acceptance Criteria established by means of a contract, assigned to Technical Inspectors and to the Petitioner of the Contract."

It is important to highlight that the Normative Instruction n. 4 of November 2010 (IN04, 2010) recommends the use of metrics in software solutions while the Court Decisions of the Federal Audits Court recommend the use of Unadjusted Function Points under contracts for the provision of systems maintenance and development services.

Also, by considering the service hiring context, Cruz, Andrade and Figueiredo (2011) present a service hiring process which complies with Normative Instruction #4. In the established process, in its stage 4 named Contract Management and in the

Perform technical monitoring activity, these authors describe the need for: monitoring the service order execution; managing risks, establishing corrective measures and making changes to the service order.

The description of these activities emphasizes the application of the constant monitoring of the service performance. The authors also highlight the need to evaluate the services provided by the Contracted Party in order to verify the "compliance with requested functional and qualitative requirements as well as quality criteria established in the processes of the work."

With a focus on the management of the service hiring process, Cobit (ITGI, 2007), which is one of the best known IT governance models, in its "Monitor and Evaluate" domain, highlights the need of the top management to ensure compliance with the IT processes by the external requirements, that is, the legislation and jurisprudence (ITGI, 2007).

In addition, COBIT, in this same domain, stresses the importance of IT processes to be regularly evaluated in order to assure quality and adherence to control requirements. There are other models which describe and emphasize the importance of the management of the service hiring process, among them, the CMMI-ACQ v1.2 (SEI, 2007), eSCM-CL v1.1 (ITSqc 2009a) (ITSqc 2009b), and the MPS.BR-Guia de Aquisição:2009 (Softex, 2009).

As such, by considering the test process characteristics, with its activities and products, Law # 8666/93, the instructions in Normative Instruction #4, the proposed service hiring process put forth by Cruz et al (2011) and the need for managing this process, the hiring of the Test factory should contain, at least, objective criteria to measure the demands, evaluate the quality of the services provided, and evaluate product quality in accordance with previously established criteria.

In the next sections, the conceptual model proposed and the research methodology will be presented as well as some types of metrics and measuring techniques, associated with the conceptual model proposed.

4 METHODOLOGY

The general objective of this work is to propose a Measurement Model for software tests by considering outsourced services, in order to make it easier for these contracts to be managed. In order to achieve this general goal, the following specific objectives were set:

- To identify and analyze the test process, its stages and activities;

- To identify and analyze existing laws and standards which govern the hiring of IT services;

- To analyze and propose a Measurement Model for outsourced test services.

The following data collection instruments were applied: research in documentation and semi-structured interviews. For the analysis of the data collected, content analysis was used (interview and documentation). In the documentary analysis, the following constructs were considered: aspects related to test processes, identifying stages, activities and products generated; the laws, standards, instructions and models concerning the test discipline.

Documentary research is the data collection method and aims to access the related sources, whether they are written or not. Written documentary sources include official, unofficial and statistical documentation. Non-written documentary sources include sources such as images and sounds, and iconography, among others. Documentary research sometimes leads to other research techniques such as observation, content analysis and others (Albarello *et al*, 1995).

By considering the data obtained through documentary research, a conceptual model was designed which represents the adopted concepts and the relationships between one another. The conceptual model built (Figure 03) was based on the confirmation that the management of test services, in order for it to be consistent with the laws, the standards in force, and the proposed models, and in order for it to be efficiently performed, it should use criteria to: measure the test services provided (size and effort), evaluate the quality of the service provided, and measure the quality of the product.

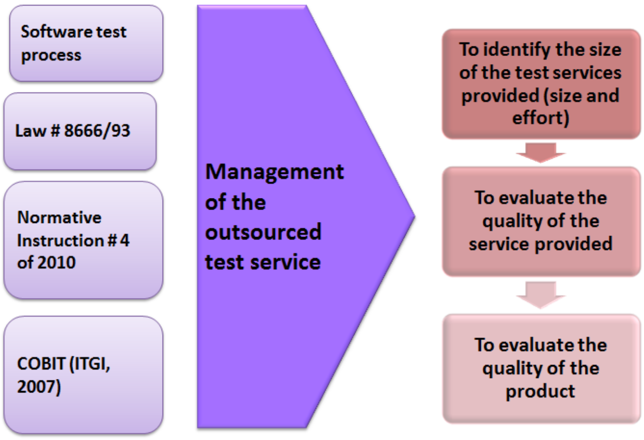


Figure 3 – Conceptual Model

That is, the construction of the model identified the need to, in order to manage the outsourced test services, propose a Measurement Model by taking the following needs into account:

- To establish criteria for the measurement of test services provided (size and effort);
- To establish the criteria for the evaluation of the quality of the service provided.
- To measure the quality of the product.

By considering these criteria, the GQM – Goal, Questions, Metrics – approach was applied in an attempt to identify the main goals, questions and the metrics related to the test services. The GQM approach was proposed by BASILI in the first half of the 90´s and has been used to provide metrics in accordance with the information needs related to the products, processes and resources used, establishing the basis for comparisons with future work (Basili & Rombach, 1994).

The GQM approach is used in relation to the assumption that an organization, in order to objectively perform measurements, must specify the objectives to be achieved by the established measurements. Such objectives direct the course of the questions which, after being refined, result in metrics, whose application will answer the established questions and, consequently, the identified measuring objectives (Basili & Rombach, 1994). The measurement model of the GQM approach works according to hierarchical levels among objectives, questions and metrics where:

-Conceptual Level – it is defined in the scope of the evaluation; that is, the object to be measured.

-Operational Level – questions that help characterize the object being studied are defined and how it must be seen within the context of quality.

-Quantitative Level – data sets to be obtained are defined, as related to each of the questions defined with the purpose of answering them in a quantitative manner; that is, as metrics.

The results of the data collected allow for an interpretation model related to the objectives set forth (Basili & Rombach,1994). The GQM paradigm provides a top-down method for the establishment of questions and metrics and a bottom-up interpretation model of the data.

The GQM approach contributes to the establishment or selection of metrics which achieve the objectives set forth by the organization and has been widely used by other models with a focus on continuous improvement. The CMMI model, for instance, says that the GQM approach is useful to select measurements that provide information about the business objectives of the organization (Chrissis, Konrad & Shrum, 2003).

In order to complement the research and aiming at the triangulation of the results, employees of a test factory were interviewed, in search of their perception of the services performed in a test contract. Also, semi-structured interviews were conducted with employees of a hiring company of a test factory, with the purpose of identifying the perception of their needs as related to the test activities hired.

The purpose of the interview is to obtain descriptions of the different aspects and the specific situations of a real-world phenomenon according to the interviewees´ view (Kvale, 1996). In the semi-structured interview, the interviewer obtains detailed information, as well data and opinions by means of a free-style conversation, following a previously prepared list of questions, supported by theories of interest to the research (Trivinos, 1987). Kvale (1996) cites five methods to analyze and interpret qualitative interviews:

Meaning condensation, meaning categorization, narrative structuring, meaning interpretation and generating meaning are generated by means of ad-hoc methods. The meaning condensation method was used in the research for the purpose of identifying common points in the perception of the participants.

Next, the research results are described aiming to identify measuring criteria of the test services provided (size and effort.)

5 METRICS FOR MEASURING THE SIZE AND EFFORT FOR THE TESTS

Chart 1 shows the comparison of some techniques and experiments identified to estimate the effort that will be put into the Software Test subject in a software development project. In this chart the metrics are succinctly described, and the advantages and disadvantages found. It is interesting to highlight that all the interviews carried out with the employees from the test factory and the employees who hire this service found the need for metrics to estimate the effort of the test. This is a necessity for both teams.

Technique	TPA – (Test Point Analysis)(Veenendaal And Dekkers, 1999)	
Description	**Advantage**	**Disadvantage**
Refines estimations by size at function points, considering impact factors: the test strategy and the productivity level. **Size** - Adds considerations to the function points about: complexity (number of conditions of the functionalities); interfaces (data kept by the functionalities) and uniformity (similarities among functions and their tests). **Test Strategy** - Takes into consideration the selection of components and their characteristics of quality; and the range of the tests. **Productivity:** scores of the variables according to predefined scales (testware, team size, etc). The productivity factor should come from the organization's base history. **Context of use:** Systems under development and/or in maintenance with specifications on test cases and measurements on function points	Functional metrics, with refinements from other physical characteristics. A well defined and detailed method found in the documentation; Contains a reference of productivity value (0.7 to 2.0 h/TPA) There are Free tools for calculating TPAs	Does not include the management of the test process (planning, monitoring and control) Refers to system tests and acceptance alone. Depends on the size at the function points and consequently they depend on the existence of the documentation Little information on productivity of the base history on the researched literature
Technique:	**FPA – Function Points Analysis(IFPUG, 2010**	
Description	**Advantage**	**Disadvantage**
Based on the functional size of the software to identify, from productivity values, the total effort required for its development. From the knowledge of a life cycle for the development/maintenance of the software and of a percentage of the distribution of the effort by its phases, subjects and/or activities, the estimation is refined for the test subject. **Context of Use:** Systems under development and/or in maintenance that have requirements (cases of use or descriptive requirements) to count function points. Knowledge of the life cycle, its phases, activities of the percentage of the distribution of the effort.	Functional metrics, widely used with plenty of well researched data on the productivity of the development/maintenance of the software and life cycles (with their percentage of the effort by phases/activities). Depending on the organization's base history it may include the management of the life cycle tests.	Depends on the system documentation to identify the size of the function points. It is a non-specific size of the measurement for the test activities. As such, it does not arrive at the level of detail of the life cycle of the tests and therefore it depends strongly on other measurements aside from productivity. For example: it depends on the identification of the percentage of the effort of the test subjects and its subdivisions by phases/activities.
Technique:	**Aranha and Borba (2007)**	
Description	**Advantage**	**Disadvantage**

(cc) BY

From a controlled natural language (CNL), the steps of the test cases are evaluated with respect to the functional and non-functional characteristics. The characteristics are identified and evaluated with respect to their relevance by experienced testers. The characteristics should also be evaluated with respect to their impacts (low, medium, high), generating scores from 0 to 10. The calibration of the model is done through the base history. **Context of Use:** . Systems under development and in maintenance that have the tests defined in the CNL.	It may be calibrated for all of the organization's projects or for project groups sorted by similar characteristics Used by Motorola	To obtain the details of the estimations, it depends on the controlled language to elaborate the test cases, which significantly complicates the preparation of the system's documentation
Technique	**Cognitive Information Complexity Measure–(CICM)(KUSHWAHA AND MISRA, 2008)**	
Description	**Advantage**	**Disadvantage**
This metric tries to measure the test's complexity through the complexity of the code. The authors work with some variables, such as: identifiers and operators, the information contained in the code, the size of this information, and the basic control information. The result of this metric is a measurement of the size of the test based on the complexity of the code. It is necessary, later on, a definition for productivity so as to identify the effort for this test. **Context of Use:** Applicable to systems already built, as it needs the source code.	Uses only the code to analyze the size of the test It could be applied to legacies without documentation	Requires the preparation of applications by language to execute the counting No information on productivity from base history in the literature researched, that is, the need for practical experiment to define productivity. The outdating of the code (considering the legacies), may interfere on the counting.
Technique:	**Estimation of the process of the software test based on the requirements (SANTRA, 2010)**	
=Description	**Advantage**	**Disadvantage**
It consists of the quantitative definition of the test cases based on the requirements. The author used a good quantitative base of requirements to arrive at an average of test cases by requirements. He also estimated the effort for the phases for the preparation and execution of the tests. **Context of Use:** New development and maintenance of the systems with defined requirements	A technique that is easy to apply. Only the defined requirements are needed	Needs practical experiment The lack of standardization of the requirements may interfere in the end result, but this may be resolved by segmenting the application of the metric by system, subject area etc.
Technique:	**Test Case Points (PATTEL ET AL ,2001)**	
Description	**Advantage**	**Disadvantage**
Test Case Points (TCP) is an approach to estimate functional test projects. This method estimates the test effort for each activity separately. This technique encompasses seven phases: Identify Cases Used Identify Test Cases Determine TCP to generate Test Cases	A technique that is easy to apply. Use cases and test cases are necessary	Needs practical experiment The lack of standardization of the use cases and of test cases may interfere in the end result, but this may be

Description	Advantage	Disadvantage
Determine TCP for automation Determine TCP for manual execution Determine TCP for Automated execution Determine total TCP (the use of an adjustment factor of up to 25%, considering the complexity of the domain, the integration with other devices, multi-language support, etc) The calculation of the effort is based on paradigms of productivity arising from the organization's base history. **Context of Use:** New development and maintenance systems with the defined requirements		resolved by segmenting the application of the metric by system, subject area etc.
Technique:	**Adjusted Use Case Points (AUCP)(NAGESWARAN ,2001)**	
Description	**Advantage**	**Disadvantage**
Consists of an adaptation of the metrics of use of case points (AUCP - Adjusted UCP) where there are scores and weights that need to be defined in the model by the test manager. The size of the AUCP tests are identified (corresponding to the planning, preparation and execution of the tests) and the effort is obtained through the identification of productivity multiplied by the AUCP Context of Use: Systems that are under development and/or in maintenance with updated specifications of case uses in order to hold a counting of the adaptation of the case use points for tests.	There is no need for a previous measurement It identifies the size of the test (AUCP) involving planning, preparation and execution of the tests	Little experimenting It depends on the existence of the documentation of the use cases, as these calculations are done considering this tool.

Chart 1 - Comparison of some techniques and experiments

5.1 Some final considerations on the estimations of the size and effort

Among the techniques mentioned for size measurement, the Test Point Analysis - TPA considers the most number of factors for the estimation, which presumes that this technique may give more consistent results to measure the size of a software test. For example, the complexity factor is obtained by the quantity of the conditions (IF-THEN-ELSE) of a function, which will directly influence the quantity of Test Cases.

The Function Point Analysis, for example, two similar functions may have the same size in FP, but if they have different complexities, the TPA technique will reflect the difference in the size of the functions.

The Test Case Points – TCP technique also seems to be more accurate with respect to the estimation of the size for the test process than the FPA, for it also considers by certain factors, the internal complexities of the functions.

It is important to highlight that all techniques use a productivity factor to derive the effort through the measurement of the size obtained by the technique. Thus, such a factor should be calibrated according to three characteristics:

- The method for measuring the size;

- The characteristics that influence the productivity of the project, such as technology, environment, team, etc;

- The strategy of the tests used, including the levels, types and test techniques as well as the test environment.

6 METRICS FOR THE EVALUATION OF THE SERVICES PROVIDED

In the context of outsourcing test services, one of the challenges to be considered is how to monitor the quality of the services provided. Also, how to validate whether the test activities and scopes of outsourcing were executed satisfactorily, especially if the tested software product is not of good quality.

Demanding only the predefined tools may be risky, for it does not guarantee the quality of the execution of the tests. As such, it is necessary to monitor closely the test process, from the strategy adopted, going through the range reached, to finalizing and follow-up of the defects that were found. To make comprehension of this subject easier, a mind map was created (Figure 04) with the most needed information. The documentary research identified that the various authors cited metrics for the monitoring of defects, effectiveness of the tests etc (Nirpal & Kale, 2011), (Caetano, 2008), (Pusala, 2006), (Kaur, Suri & Sharma,, 2007). It is interesting to highlight that the interviews emphasized the need for some of these metrics to follow up the service provided. The employees from the test factory as well as the hiring company cited the absence of this monitoring.

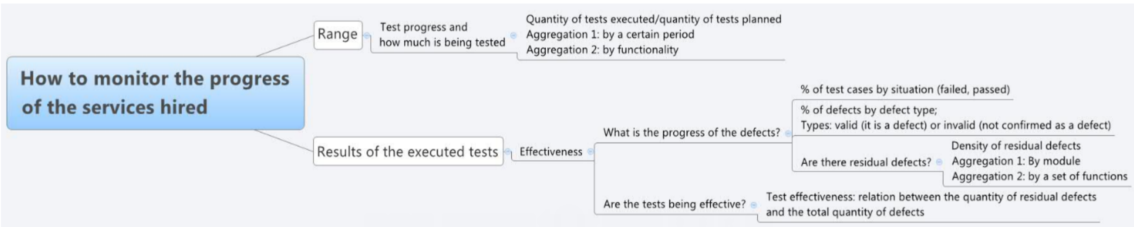
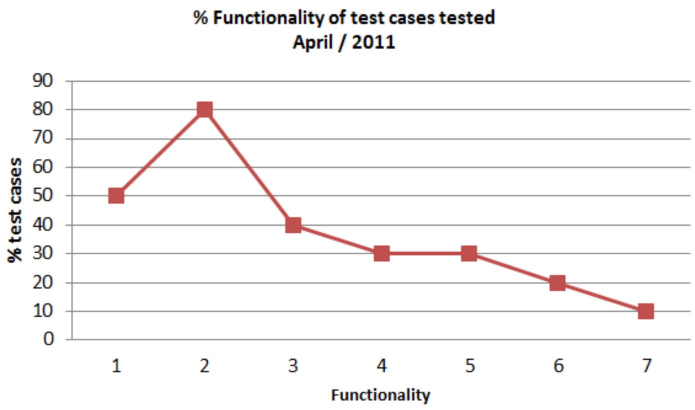
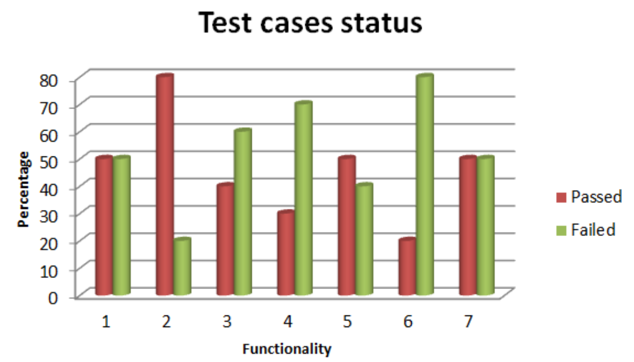


Figure 4 - Measurement models in tests - Follow-up of the tests

Next, some of the defined objectives, metrics and questions are described:

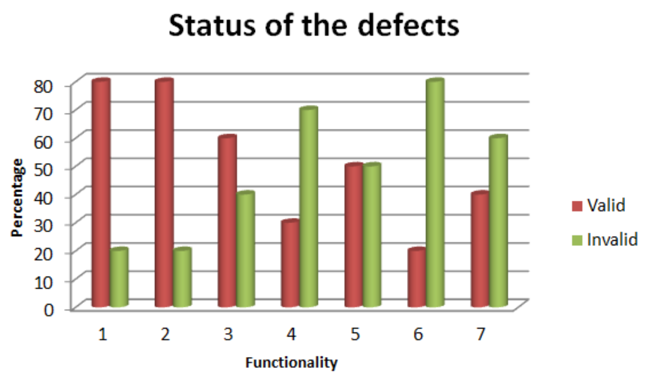
Objective 1: Monitoring the progress of the services provided	
Purpose: Follow-up Subject: progress Object: outsourced test services	
Question 1.1: How much is it being tested?	
Metrics 1.1a	**The aim of this metric is to verify proportionally how much is being tested**
Classification	Objective, quantitative and follow-up
Measurement base	Quantity of test cases tested (of a single functionality) relative to the total of test cases (functionality)
Measurements	M1.1.a=(Σ functionality of test cases tested/ Σ functionality test cases)*100

Indicator	To calculate the percentage of what is being tested relative to the total test cases during a certain period The objective of this indicator is to point out the volume of what is being addressed relative to what is planned during a certain period
Example of the indicator	
	
Analysis Model	The analysis model of this indicator is given by dividing the sum of the functionalities of test cases tested by the sum of all the functionalities of test cases prepared during a certain period The indicator must be represented by a line graph, and it will show the percentage variation of test cases tested As the test process becomes solid, the tendency is that the percentage stabilizes itself
Interpretation of the indicator	Aim - 90% Ideal value - 100% of range . Low percentages show the need to act to adhere to the test process. There is a high risk associated with low levels since the quality of the test and the product will be impacted.
Analysis procedure	Frequency - monthly Responsibility - test team Phase or activity in which it has to be analyzed - At any time
Question 1.2: What is the progress of the defects?	
Metrics 1.2a	**The objective of this metric is to identify the situation of what was tested**
Classification	Objective, quantitative and follow-up
Measurement base	Relation between the quantity of test cases that passed, failed and the total of test cases tested
Measurements	M 1.2a1 = (Σ test cased that passed/ Σ test cases)*100 M 1.2a2 - (Σ test cases that failed/ Σ test cases)*100
Indicator	The objective of this indicator is to point out the status of the functionalities of test cases tested, considering monthly periods of evaluation
Example of the indicator	



Analysis Model	The analysis model of this indicator is found by the division of the sum of the test cases executed by status and the total of test cases executed The indicator should be represented by a bar graph, and it shows the percentage variation by functionality of the statuses of the test cases.
Interpretation of the indicator	Low percentages of test cases that passed may show the need for actions of adherence to the development and/or test process. High percentages of case tests that failed, were blocked and re-executed may show the need for actions of adherence to the development and/or test process. It is interesting to see, in case of large variations among functionalities, the variables that are impacting the indicators positively or negatively.
Analysis procedure	Frequency - monthly Responsibility - test team Phase or activity in which it has to be analyzed - At any time
Metrics 1.2b	**The objective of this metric is to identify proportionally the occurrences of defects that are not confirmed (valid, invalid)**
Classification	Objective, quantitative and follow-up
Measurement base	Relation between the quantity of defects by type and the total defects identified
Measurements	M 1.2.b1 = (Σ valid defects/ Σ defects)*100 M 1.2b2 = (Σ valid defects/ Σ defects)*100
Indicator	The objective of this indicator is to point out the status of the test cases tested by functionality, considering the monthly periods of evaluation

Example of the indicator



Analysis model	The analysis model of this indicator is found by the division of the sum of the types of defects and the total of defects found by functionality The indicator should be represented by a bar graph, and it shows the percentage variation by functionality of the types of defects
Interpretation of the indicator	Aim - valid defects - 100% and invalid - 0% High percentages of valid defects may show the need for actions of adherence to the development and/or test process High percentages of invalid or abandoned defects may show the need for actions of adherence to the development process.
Analysis procedure	Frequency - monthly Responsibility - test team Phase or activity in which it has to be analyzed - At any time

7. METRICS FOR THE EVALUATION OF THE PRODUCT TESTED

Finally, it is most important to ensure that the product has the quality expected by all, in accordance with the criteria of the quality demanded. Such criteria of quality must be evaluated according to criteria that are also objective, that is, by metrics software. As such, the NBR ISO/IEC 9126 norm itself provides in parts 2, 3 and 4 the metrics for the evaluation of the criteria of quality, be it External, Internal, and Quality of Use of this norm.

A software product does not reach its complete stability in the first releases of the software. The most important fact is that the evolution of the defects be monitored as soon as possible and that the causes are addressed during the development process. This way, the NBR ISO/IEC 9126 norm itself provides a set of metrics for each of the characteristics of quality and their respective sub-characteristics. Such metrics aim to answer such questions as:

- How adequate are the evaluated functions?

- How complete are the functions relative to the specified requirements?

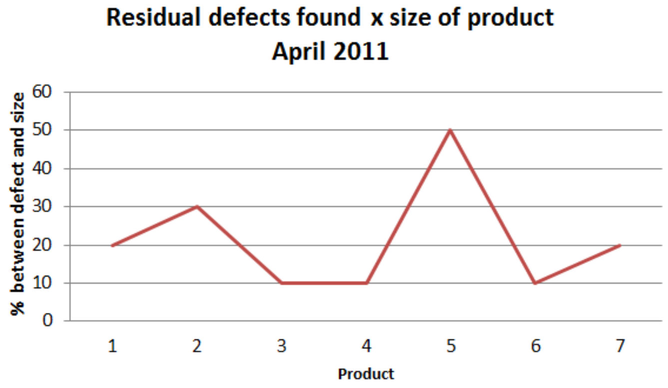
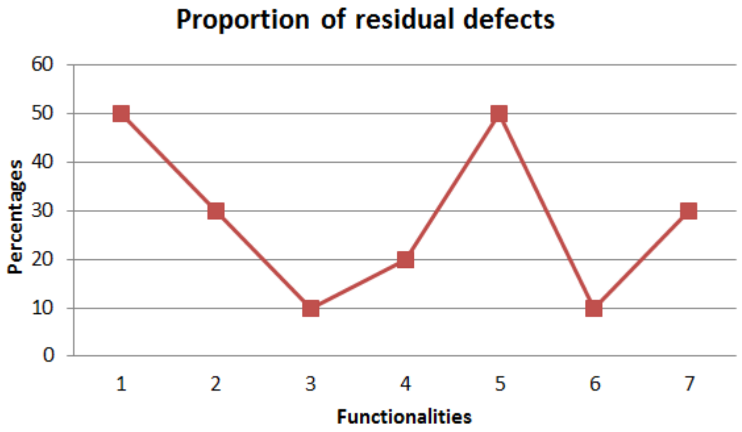
- How frequently do the users find incorrect results?

- How complete are the auditing records in reference to the accesses by users of the system and to the data?

It is important to highlight that the follow-up to the quality of the product should be in line with the criteria of the quality demanded and that the strategy of the tests should test the attributes that best represent the adherence to the desired and adequate level of quality. In this context, the criteria of quality relative to the non-functional requirements are normally forgotten or non-prioritized, for example, the requirements of Performance or Efficiency. Should there be any requirement of performance for some function, this attribute should be measured and validated by the test process.

The documentary research identified a few other measurement proposals of quality of the product, frequency of defects etc ((Lazic & Mastorakis, 2008), (Kaur et al, 2007)). These measurements, aside from the ISO IEC 9126 proposal, were also confronted with the results from the interviews, aiming to identify the most relevant ones. Described below are some metrics given to evaluate the product tested.

Objective 2: How to evaluate the product tested	
Purpose: Identify Subject: evaluation Object: product tested	
Metrics 21	**Identify the residual density of the defects**
Classification	Objective, quantitative and follow-up
Measurement base	Relation between the quantity of defects that the end-user found and the size
Measurements	M 1.1a - (Σ number of defects found by module or set of functions/size of the product)
Indicator	The objective of this indicator is to point out the proportion in the number of defects found after the test process and the size of the product
Example of the indicator	

	
Analysis model	The analysis model of this indictor is found by the division of the sum of the quantitative of the defects found by module after the test process by the size of the point of application. The indicator should be represented by a line graph, and it shows the proportion of defects by module.
I Interpretation of the indicator	Aim - 0 Ideal value - 0 It is interesting to verify in the case of large variations, among the modules, the variables that are impacting the indicators positively or negatively.
Analysis procedure	Frequency - monthly Responsibility - test team Phase or activity in which it has be analyzed - At any time
Question 2.2: Are the tests being effective?	
Metrics 2.2	**Identify the effectiveness of the test considering the residual quantity of the defects over the total of defects (residual and internal)**
Classification	Objective, quantitative and follow-up
Measurement base	Relation between the quantity of defects that the end-user found and the total quantity defects
Measurements	M 2.2 = (Σ number of residual defects / Σ number of internal and residual defects found)/100
Indicator	The objective of this indicator is to point out the proportion between the number of residual defects after the test process and the total defects
Example of the indicator 	
Analysis model	The analysis model of this indicator is found by the division of the quantitative of residual defects by the sum of the residual and internal defects The indicator should be represented by a line graph, and it shows the percentage variation of the residual defects
Interpretation of the indicator	Aim - 0 High percentages may show a need for actions of adherence to the test and development process.
Analysis procedure	Frequency - monthly Responsibility - test team Phase or activity in which it has to be to be analyzed - At any time

8 CONCLUSIONS AND FUTURE PAPERS

The general objective of this paper was to "propose a measurement model for software tests, considering the outsourcing of this service and the need to support the management of these contracts. In order to achieve this general objective, the following specific objectives were defined:

- Identify and analyze the test process with all its phases and activities;

- Identify and analyze the already existing laws, norms and models that regulate the hiring of services in IT;

- Analyze and propose a measurement model to outsource test services

This research allowed identifying the complexity of the subject of the tests through the study of the test process. The analysis of the already existing laws, norms and models allowed defining the conceptual model of the research that identified the need to manage the service for the outsourcing of the test services, the building of a Measurement Model considering the following criteria:

- Measuring the test services provided (size and effort);

- Evaluating the quality of the service provided;

- Measuring the quality of the product

Furthermore, it was evident that the outsourcing of any IT service also needs to consider the characteristics that influence the productivity of the project, like technology, environment, team, etc., and the strategy of the tests used, including their levels, types, and test techniques as well as the test environment.

Finally, after the analysis of the measurements found in the specialized literature, of the already existing norms, instructions, and models to manage the outsourcing of services in the governmental sphere, and of the application of the GQM methodology, it is possible to: establish a measurement in size, and consequently input, to estimate the effort and the time frame for tests demands monitoring the sub processes of outsourced tests, also by means of objective and measurable criteria; and to establish the criteria of quality, evaluating whether the end-product meets such criteria. As such, the management of this type of outsourcing would be made viable in a more efficient manner. It is important to highlight that the interviews held validated the identified needs as well as the proposed measurements.

For future papers, the implementation of the model, and of the proposed measurements to verify its applicability, is recommended.

ACKNOWLEDGEMENTS

Support by TiMetricas

REFERENCES

Albarello, L., Digneffe, F., Hiernaux, J., Maroy, C., Ruquoy, D., & Saint-Georges, P. (1995). *Prática e Métodos de Investigação em Ciências Socias*. Portugal: Gradiva.

Aranha, E., & Borba, P. (2007). An Estimation Model for Test Execution Effort. *First International Symposium on Empirical Software Engineering and Measurement – IEEE computer society.*

Basili, V., & Rombach, H. (1994). Goal question metric paradigm. *Encyclopedia of software engineering*. (2).

Law n° 8.666, of June 21, 1993 (1993). Regulates the art. 37, item XXI of the Constitution, establishing rules for bidding and contracts for Public Administration and other measures. Brazil. Retrieved 16 agosto, 2011, from http://www.planalto.gov.br/ccivil_03/Leis/L8666cons.htm.

Normative Instruction SLTI # 4, of Nov 12, 2010. (2010). Provides for the process of hiring the services of Information Technology for Public Administration Federal direct, autonomous agencies and foundations. Brazil. Retrieved 16 agosto, 2011, from http://www.governoeletronico.gov.br/sisp-conteudo/nucleo-de-contratacoes-de-ti/modelo-de-contratacoes-normativos-e-documentos-de-referencia/instrucao-normativa-mp-slti-no04.

Caetano, C. (2002). Gestão de defeitos. Engenharia de Software, year 1, 1ª. Edition

Caetano, C. (2008). Gestão de Testes Ferramentas Open Source e melhores práticas na gestão de testes. *Engenharia de software v.3.*

Chrissis, M. B., Konrad, M., & Shrum, S. (2003). *CMMI: Guidelines for Process Integration and Product Improvement*. Addison-Wesley

Cruz, C. S., Andrade, E. L. P., & Figueiredo, R. M. C. (2011). *PCSSCEG - Processo de contratação de serviços de Tecnologia da Informação para Organizações Públicas*. DF: MCT.

Fenton, N, & Pfleeger, S. (1997). *Software Metrics: A Rigorous and Practical Approach*. (2nd.ed.) Boston: PWS Publishing Company.

Institute of Electrical and Electronics Engineers IEEE. (2008). *Standard for Software & System. Test Documentation*. IEEE 829-2008.

International Function Point Users Group, IFPUG. (2010). *Manual de Práticas de Contagens de Pontos de Função*, v. 4.3.1.

Information Technology Governance Institute, ITGI.COBIT. (2007). *Control Objectives for Information and related Technology. 4.1.ed*. Retrieved 16 agosto, 2011, from http://www.isaca.org/Knowledge-Center/cobit/Pages/Downloads.aspx.

International Organization for Standardization and International Electrotechnical Commission. (2002). *ISO/IEC 9126:2002 Software quality.*

Information Technology Services Qualification Center, ITSqc. (2009a). *eSourcing Capability Model for Client Organizations (eSCM-CL).* v1.1, part 1. Retrieved 16 agosto, 2011, from http://www.itsqc.org/downloads/documents/eSCM-CL_Part1_V1dot1.html.

Information Technology Services Qualification Center, ITSqc. (2009b). *eSourcing Capability Model for Client Organizations (eSCM-CL).* v.1.1, part 2. Retrieved 16 agosto, 2011, from http://www.itsqc.org/downloads/documents/eSCM-CL_Part2_V1dot1.html.

Jones, C. (2007). Software Estimating Rules of Thumb. [Working Paper]. *Capers Jones*. Retrieved 15 march, 2011, from http://www.compaid.com/caiinternet/ezine/capers-rules.pdf.

Juristo, N., Moreno, A. M., &Vegas, S. (2004). Reviewing 25 years of testing technique experiments. *Empirical Softw. Eng., v. 9, n. 1-2, p. 7-44*.

Kaur, A., Suri, B., & Sharma, A. (2007, March). Software Testing Product Metrics - A Survey. *Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007)* RIMT-IET, Mandi Gobindgarh, 23.

Kushwaha, D. S., & Misra, A. K. (2008). Software Test Effort Estimation. *ACM SIGSOFT Software Engineering, v. 33, n. 3.*

Kvale, S. (1996). *Interviews: an introduction to qualitative research interviewing*. California: Sage publications.

Lazic, L., & Mastorakis, N. (2008). Cost Effective Software Test Metrics. *Wseas Tansactions on Computers. v. 7, n.6.*

Nageswaran, S. (2001, June). Test effort estimation using use case points. *Proceedings of Quality Week*, San Francisco, California, USA.

Nirpal, P. B., & Kale, K. V. (2011). A Brief Overview Of Software Testing Metrics. *International Journal on Computer Science and Engineering (IJCSE), v. 3 n. 1.*

Patel, N., Govindrajan, M., Maharana, S., & Randas, S. (2001). Test Case Point Analysis. [Working Paper] *Cognizant Technology Solutions*. Retrieved 15 march, 2011, from www.stickyminds.com/getfile.asp?ot=XML&id=2566&fn=XUS373692file1.pdf.

Pfleeger, S. L. (2004). *Engenharia de software: teoria e prática*. (2nd. ed.) São Paulo: Prentice Hall.

Pressman, R. (2006). *Engenharia de Software*. (6th ed.).Sao Paulo: Mgraw-Hill.

Pusala, R. (2006). Operational Excellence through efficient software testing metrics. [Working Paper]. *Point view –Infosys*. Retrieved 15 march, 2011, from http://www.infosys.com/it-services/independent-validation-testing-services/white-papers/documents/operational-excellence.pdf

Santra, A. (2010). A New approach for estimation of software testing process based on software requirements. *Journal of scientific & industrial research, v. 69, pp.746-749.*

Software Engineering Institute, SEI. (2007). *CMMI for Acquisition (CMMI-ACQ).* V. 1.2. Retrieved 01 march, 2011, from www.sei.cmu.edu/cmmi/tools/acq/download.cfm

Silva, M. A. da S., Duarte, R. G., & Castro, J. M. de. (2009). Outsourcing de TI e redefinição do papel da subsidiária: um estudo comparativo entre as subsidiárias brasileiras e indiana de uma multinacional americana. *Journal of Information Systems and Technology Management, v. 6, n. 2, pp. 173-202.*

Associação para Promoção da Excelência do Software Brasileiro, SOFTEX (2009). *MPS.BR – Melhoria de Processo do Software Brasileiro: Guia de Aquisição*. Retrieved

15 march, 2011, from www.softex.br/mpsbr.

Sommerville, I. (2007). *Engenharia de software*. (8th. ed). São Paulo: Pearson Addison-Wesley.

Trivinos, A. N. S. (1987). *Introdução à pesquisa em ciências sociais: a pesquisa qualitativa na educação*. São Paulo: Atlas.

Veenendaal, E, & Dekkers, T. (1999). *Test point analysis: a method for test estimation, Project Control for Software Quality*. Shaker Publishing.

Venkatasubramanian, A., & Vinoline, V. (2010). Software Test Factory (A proposal of a process model to create a Test Factory). *International Journal of Computational Intelligence Techniques*, *v.1*, *n. 1, pp.14-19.*