Towards a Maturity Model in Software Testing Automation

Ana Paula C. C. Furtado, Silvio R. L. Meira
Informatics Center – Cin
Federal University of Pernambuco – UFPE
Recife, PE, Brazil
{apccf, srlm}@cin.ufpe.br

Marcos Wanderley Gomes
SOFTEXRECIFE
Recife, PE, Brazil
marcos@recife.softex.br

***Abstract*—The practice of testing software is one of the ways to produce software with quality for demanding clients in the software market. The automation of Software testing may be seen as a solution for how to test the greatest amount of software within a project, due to the fact that the more the software is built, the larger the scope of testing is. Therefore, organizations that seek to guarantee that their software projects are being built according to the demands of their clients should follow an automation approach to testing. Thus, this paper puts forward a description of work in progress on the development of a maturity model for automating software testing that is being developed as part of a doctoral thesis. Besides presenting the overall expected structure of the maturity model, the plan for validating it is also set out.**

Keywords-software testing; automation; maturity models.

I. INTRODUCTION

Software Testing is an essential activity in today´s world of software development, given that customers are more and more rigorous about the quality of products being delivered to the market. It is necessary to test in order to minimize the risks of finding faults in the software while in clients' production environment.

Within this context, automating software testing appears as an alternative to manual tests in order to cover a broader scope of the functionalities tested within a shorter period of time. According to International Software Testing Qualification Board (ISTQB) [1], test automation is the use of software to execute or support testing activities, such as test management, test case, test execution and assessing results. Nevertheless, the automation of testing is an activity that can be introduced in order to gain productivity from the team and additional quality in the artifacts generated.

Another relevant perspective from which to approach excellence in software development is to use maturity models to support the continuous improvement of the processes within an organization. There are maturity models that cover the entire scope of development activities, such as Capability Maturity Model Integration for Development (CMMI-DEV) [2] and MPS.BR [3] (the acronym in Portuguese for Improving Software Processing: a Brazilian model) which is a Brazilian model that was developed with a view to the global software community considering it better suited to its needs. Nevertheless, there are three other models that were built specifically to build more discipline into testing, namely Testing Maturity Model – TMM [4], Test Maturity Model Integration – TMMI [5] and MPT.BR [6] (the acronym in Portuguese for Improving Test Processing: a Brazilian model), and thereby to support the introduction of testing in a more disciplined and prescribed manner.

However, it is observed that none of them discuss testing automation as an issue within maturity models. Organizations that seek to automate their testing have no support from maturity models which would help them to understand what the best practices of automation are and how to introduce these into their organizations.

Therefore, this paper sets out the overall structure of a maturity model for automating software testing that is being developed as part of a doctoral research study.

This paper is organized as follows: the next section gives an overview of the discipline of software testing and its main concepts. Section 3 gives the background to maturity models and comments on what they offer in terms of automating software testing. Section 4 explains the framework for the maturity model and Section 5 makes concluding remarks and suggests future lines of study.

II. SOFTWARE TESTING BACKGROUND

According to ISO/IEEE [7], testing is a set of activities conducted to facilitate discovery and/or evaluation of properties of one or more test items. Testing activities can include planning, preparation, execution, reporting, and management activities, insofar as they are directed towards testing.

Meyers [8] states that software testing is the process of executing a program with the intent of finding errors. The book, *A Guide to Advanced Software Testing* [9], states that testing can also be considered a support activity: it is meaningless without the development processes and does not produce anything in its own right: nothing developed entails nothing to test.

All such statements give a general idea of the definition of software testing and essentially lead to the same overall objective of software testing which is not to find every system/software bug that exists, but rather to uncover situations that could negatively impact the business. Nevertheless, note that the cost of finding and fixing bugs can rise considerably during the development life cycle. Therefore, the earlier in testing that bugs which are judged

 ISBN: 978-1-61208-367-4

likely to have moderate or serious impacts on later stages are identified and fixed, the better.

On the other hand, ISTQB [1] declares test automation as the use of software to perform or support test activities, e.g., test management, test design, test execution and results checking. According to ISO/IEEE [7], automated testing is often considered to be mainly concerned with conducting tests on scripted tests rather than having testers conduct tests manually. However, many additional test tasks and activities can be supported by software-based tools.

The activity of automating tests assumes that tools are used, and, according to Hass [9], the purpose of using tools for testing is to get as many as possible of the noncreative, repetitive, and boring parts of the test activities automated. The purpose is also to exploit the possibility of tools for storing and arranging large amounts of data.

Automation may help solve problems, especially those caused by:

- Work that is to be repeated many times;
- Work that it is slower to do manually; and
- Work that it is safer to do with a tool.

Another goal when introducing automation techniques into the discipline of testing is to increase the productivity of the team. Otherwise the cost of introducing automated practices would not be compensated for. Figure 1 is a graphical representation of the comparison of the cost of manual and automated testing.

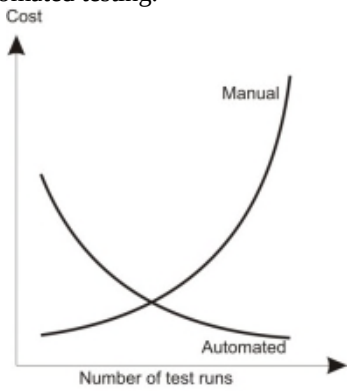

Figure 1. The cost of testing, by Hass [9].

Therefore, this section presented the main concepts of software testing used for this research. The next section comments on maturity model concepts used as references to implement a testing maturity model in automation.

III. SOFTWARE MATURITY MODELS

According to Prado [10], maturity can be defined as "a way to measure the stage that an organization is at in its ability to manage its projects." The main objective is to help improve the way software is being built.

In order to suggest a maturity level for automated testing, the main maturity models studied were CMMI-DEV [2], TMMI [5] and MPT.BR [6], which will be explained in the following sections.

A. TMMI

TMMI [5] is a maturity model that was produced by the TMMI Foundation which used TMM [4] as a reference. It aims to work as a complementary model to CMMI-DEV [2] and, therefore, it is organized in maturity levels, as presented in Figure 2.

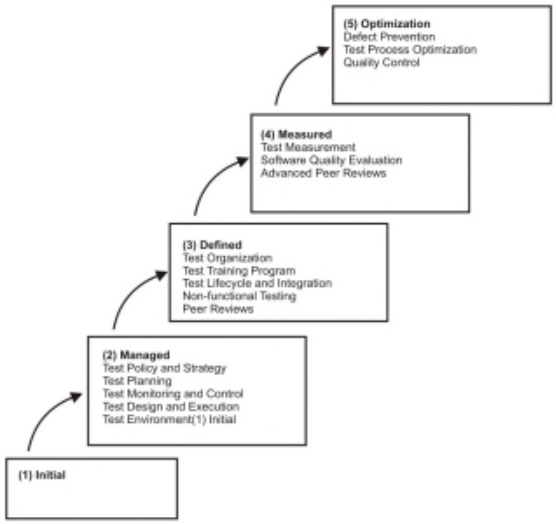

Figure 2. TMMI Maturity Levels [5].

The framework of the model consists of 16 process areas. However, the model states “TMMI does not have a specific process area dedicated to test tools and/or test automation”. The model comments that test tools are treated as a supporting resource (practices) and are therefore, part of the process area where they provide support.

B. MPT.BR

MPT.BR [6] approaches the enhancement of the testing process by using the best software testing practices throughout the product lifecycle. MPT.BR uses guidelines on how best to improve the software testing process throughout the lifecycle of the software.

It was developed to be introduced as a complement of MPS.BR [3], which focuses on software processing, but pays scant attention to testing disciplines.

The MPT.BR reference model presents five maturity levels, representing the stages for evolving a test process in the context of an organization. The maturity levels are shown on Table I.

The levels comprise 16 processes areas, one of which, AET (which is the acronym in Portuguese for Test Execution Automation), that specifically addresses testing automation, the objective of which is to establish and maintain a strategy for automating test execution activity, by defining its objective, defining a framework and assessing the Return on Investments (ROI).

There is another process area, called GDF (which stands for tools management), that mentions testing tools. Its objective is to manage the identification, analysis, selection and deployment of tools to support testing activities, in

 ISBN: 978-1-61208-367-4

general, within an organization. This process area does not mention any specific tool; it talks about the necessity to plan organizationally, to instantiate and to manage the use of tools within a project.

TABLE I. MPT.BR PROCESSES AREAS

Maturity Level	Objective
1 – Partially Managed	This contains the minimal requirements that a company needs to meet in order to demonstrate that the discipline of testing is applied to projects and that this takes place in a planned and monitored manner.
2 – Managed	This takes a broader view in which the scope of the project starts to be controlled by the management of change process. In addition, software testing patterns are defined and processes are monitored and controlled.
3 – Defined	At this level, testing becomes organizational. Defined software processes are adopted, quality Assurance is institutionalized in order to support process definition, responsibilities for test organization are defined and a measurement program is institutionalized in the organization. At this level, the software testing lifecycle is associated with the development one, where static and acceptance testing are formalized and systematic procedures are applied for test closure.
4 – Defect Prevention	This focuses on preventing defects and systematically improving the quality of the product. At this level, the organization has a process for managing defects, in which defects found are monitored.
5 – Automation and Optimization	The fifth maturity level sets out to establish a process for testing that continuously improves tests and automates them.

C. CMMI-DEV

CMMI-DEV [2] is a model that consists of best practices that address development activities applied to products and services. It addresses practices that cover the product's lifecycle from conception through delivery and maintenance.

The structure of the model comprises 22 process areas organized in 5 maturity levels, which are:

1. Initial;
2. Managed;
3. Defined;
4. Quantitatively Managed; and
5. Optimizing.

CMMI is a maturity model that can be applied by means of staged or continuous representation. In the former, the organization can improve a set of related processes by incrementally addressing successive sets of process areas. The latter enables organizations to improve processes corresponding to individual process areas, by making it possible to choose the ones that best fit the organizational environment.

Both representations use the same set of process areas, and there are 2 that specifically talk about testing, as shown in Table II.

TABLE II. CMMI PROCESS AREAS OF TESTING

Process Area	Description.
Verification	The purpose of Verification (VER) is to ensure that selected work products meet their specified requirements.
Validation	The purpose of Validation (VAL) is to demonstrate that a product or product component fulfills its intended use when placed in its intended environment.

Both process areas talk about practices on how to guarantee quality by means of testing activities (static and dynamic testing), but there are no recommendations on how to conduct automated practices for testing activities.

D. Automation Approach on Maturity Models

This section combines the maturity models that are used as main references to build the MPTA.BR. Table III summarizes the maturity models together with the approach of automation contained in each, if present.

TABLE III. MATURITY MODELS AND AUTOMATION APPROACH

Maturity Model	Automation Approach
CMMI	No automation approach defined, there are two process areas that talk about testng, namely, VER and VAL.
TMMI	Automation can be done in any process area but there is no guidance on how to do it.
MPT.BR	Level 5 presents two process areas, one of which is AET which talks about automation and GDF which mentions tools, in general, including automated ones.

The next section will present the proposal of the work in progress for developing a maturity model for automation

IV. MPTA.BR

The Maturity Model MPTA.BR (the acronym in Portuguese for Improving the Test Automation Process: a Brazilian model) aims to be complementary to MPT.BR, as it provides guidance to be used when developing automation processes within an organization.

In the current marketplace, maturity models, standards, methodologies, and guidelines exist that can help an organization improve the way it does business. According to CMMI-DEV [2], "the quality of a system or product is highly influenced by the quality of the process used to develop and maintain it."

The idea to develop a maturity model on automation arose from both technical research and a bibliographic review as well as from demand in the software development industry. Research specifically undertaken in organizations that have achieved a maturity level on MPT.BR [11] shows that they are interested in applying automation techniques to their testing processes.

MPTA.BR will follow the structure of MPT.BR, where each maturity level consists of a set of process areas, which can be understood as a group of related practices that, when implemented together, satisfy a specific objective. Each maturity level is also associated with generic practices that are applied to each process area.

 ISBN: 978-1-61208-367-4

A. Maturity Levels

The maturity levels were influenced by the organization of MPT.Br, together with a classification of tools for automation defined by Hass [9]. This describes an evolutionary track recommended for a company that aims to introduce testing automation processes. The maturity levels are:

1. **Managed**: its objective is to introduce the automation of planning and monitoring activities of the test project as well as configuration management practices.
2. **Designed**: This maturity level focuses on the definition of automated practices in test design and debugging activities, as well as troubleshooting and static analysis tools.
3. **Executed:** The objective of this level is to automate data generation, simulation, emulation, fault-sending, fault-injection and test case execution.
4. **Analyzed:** The objective is to use tools to support a comparison of results and indicators.

The suggested model improves the concepts presented in the other models to include specific guidance on how to introduce test automation in an organization.

B. Validation of MPTA.BR

The validation of the model is planned to occur through the following processes of Case Study and Survey.

The case study is run by adopting the model in selected and volunteer organizations following the steps below:

1. Making an initial diagnosis to identify gaps and to assess the automation practices that exist (if any) in the organizations;
2. Building an action plan to introduce the practices of the model in the organization;
3. Running a pilot project; and
4. Assessing the pilot project to identify if the intended objectives of the model were achieved, in fact, with the support of MPTA.BR.

On the other hand, a survey with specialists can be conducted by selecting a group of experienced professionals, both from the academic world and industry. Thereafter, it is necessary to run a survey in order to assess their opinion on the effectiveness of the model with regard to helping to introduce automation practices in organizations.

Both methods of validation can run in parallel and after collecting the results from both, the positive and negative aspects of the model will be assessed and the improvement opportunities consolidated in order to generate the final version of the model which the doctoral research study sets out to do.

Certain limitations to validate the model can be observed, such as the difficulty to find an organization to run the case study and to select/find the correct specialist to execute the survey.

V. CONCLUSION AND FUTURE WORK

It has been observed that software testing automation can be used to support organizations to achieve higher levels of quality in the products being developed by the software industry. Maturity models that are being used world-wide give little, or almost no guidance on how to implement automation in testing processes.

This work in progress is part of a proposal for a doctoral thesis that is being developed and was prompted by prior research and a review of the literature besides which it was noted from personal experience and observations that there is a demand from the software industry for a model of this nature. The objective is to propose guidelines using a maturity model on software testing automation in order to help organizations gradually introduce automation practices.

One of the threats that may arise from this research is related to the fact that automation might not be the solution for an organization´s needs and its introduction may make the process heavier than necessary.

Another relevant threat is that even defining MPTA.BR as the model to be implemented might make it more difficult than expected to run study cases in the real world environment because it is hard to convince organizations to introduce practices of a model that is under construction.

Automation may not be the solution for all software development projects because its incorrect use may lead to an increase in cost and not make sense in the end. As to future research, this model will be detailed with the information necessary and this will include making detailed descriptions of its structure and processes areas.

REFERENCES

[1] ISTQB, “Standard Glossary of Tearms Used in Software Testing”, Version 2.2, October 2012.

[2] CMMI-DEV, “CMMI for Development”, Version 1.3, CMU/SEI-2010-TR-033, Software Engineering Institute, 2010.

[3] MPS.BR, “Improving Software Processing: a Brazilian model ”, Softex, Available from http://www.softex.br/mpsbr/guias/, 2014.08.11

[4] TMM, “Test Maturity Model”, Illinois Institute of Technology, Available from http://science.iit.edu/computer-science/research/testing-maturity-model-tmm 2014.08.11

[5] TMMI, “Test Maturity Model Integration” Release 1.0. TMMi Foundation, Ireland. Available from http://www.tmmi.org/pdf/TMMi.Framework.pdf 2014.08.11.

[6] Softex Recife, “MPT - Improving Test Processing: a Brazilian model”. Available in http://mpt.org.br/mpt/wp-content/uploads/2013/05/MPT_captured on Apr. 27th 2014. Portuguese Version Only.

[7] ISO/IEEE 29119 – Part I International Standard, “Software and Systems Engineering/Software Testing, Concepts and Definitions, First Edition, 2013.

[8] J. Glenford Myers, “The Art of Software Testing,” John Wiley and Sons, ISBN 0-471-04328-1, 1979

[9] A. Hass, “A guide to Advanced software testing,” Artech House, INC.2008

[10] D. Prado, “Project Management in Organizations”, 2 ed. Belo Horizonte: Editora de Desenvolvimento Gerencial, 2003. Portuguese Version Only.

[11] A. Furtado, M. Gomes, E. Andrade, I. de Farias Junior, “MPT.BR: A Brazilian Maturity Model for Testing” The 12th International Conference on Quality Software (QSIC), August 2012, pp. 220-229, ISSN 1550-6002, ISBN: 978-1-4673-2857-9, DOI 10.1109/QSIC.2012.53

 ISBN: 978-1-61208-367-4