# TAPN: Test Automation's Pyramid of Needs

**Trabalho Técnico**

**Anderson Rodrigues, Arilo C. Dias-Neto, Allan Bezerra**

Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Av. Rodrigo Otávio, 6.200 – Manaus – AM – Brasil

anderson@icomp.ufam.edu.br, arilo@icomp.ufam.edu.br,
allan.bezerra@icomp.ufam.edu.br

***Abstract.*** *Testing is an essential activity to ensure quality of software systems, but it is expensive and time consuming. Thus, testing automation would be an alternative to improve test productivity and save costs. However, many organizations refuse to use test automation or had failed on implement it because they do not know how to deal with the implementation of a test automation strategy fitted to their goals and expectations. Most of them underestimate or have no knowledge about test automation factor of success. In addition, although there are many works and maturity models focused on improving the testing process, few ones focus on test automation issues. The main contribution of this paper is to propose a hierarchical model called Test Automation's Pyramid of Needs (TAPN). TAPN is inspired in the Maslow's Pyramid of Needs administration theory and it comprises five levels that influence on the success of test automation initiatives in software organizations. TAPN intends to help organizations to build their test automation strategy using good practices in test automation captured from the technical literature.*

## 1. Introduction

Testing is the one of the most effective ways to ensure some level of quality to a software product. However, it is expensive and a time-consuming activity. Testing consumes between 20-50% of a software project's budget. Even so, millions of dollars are spent in correcting defects after delivering the software to the customer [Lee et al. 2012][Kasurinen et al., 2010]. Many efforts have been dedicated on developing frameworks to improve the software testing process aiming to reduce costs and losses [Van Veenendaal, 2010][Andersin, 2004]. More recently, test automation has been considered a good alternative to reduce even more costs and increase the efficiency of the test activities [Cervantes, 2009][Karhu et al., 2009]. However, the technical literature shows that even after decades, a big portion of software organizations still resist to automate their software testing activities or have failed to implement it. Moreover, the rate of software organizations with success in their strategy for test automation is below 50% [Lee et al. 2012][Kasurinen et al., 2010].

Software organizations tend to create unrealistic expectations about test automation and underestimate its implementation process as well [Pettichord, 1999]. Current testing process maturity models, such as TMM [Burnstein et al., 1996], TMMi [Van Veenendaal, 2010] and TPI [Andersin, 2004], do not address test automation with enough details. This practice is usually dealt only as a support tool, without suggestions to how it should be applied to a software project and how to measure his impact. To the

best of our knowledge, TAIM [Eldh et al., 2014] is the first attempt to propose a model to handle factors that impacts test automation, and define a measurement method to support software organizations in order to evaluate their progress regarding test automation. However, it is still an on-going work and the authors do not provide further information about the complete scenario.

Based on this scenario, in this paper we have tried to answer two main questions: (1) Which factors have the most significant impacts on the success of software test automation strategies? (2) How to support a software organization to have success in a test automation strategy implantation? Two main results are presented as consequence of these questions:

1. **List of success factors** that may have influence in the application of a test automation strategy in a software organization, collected from a systematic mapping study that investigated these factors in technical literature;
2. **A set of good practices** that support software organizations to start or evaluate their software test automation strategies also collected from the technical literature. These practices were hierarchically organized into a model that comprises levels of a pyramid, called *Test Automation's Pyramid of Needs* (TAPN), inspired on the *Pyramid of Needs Theory* proposed by Maslow (1943).

The remainder of this paper is structured as follows: Section 2 describes some significant test process support models (TPSM) proposed in the technical literature and how they deal with test automation. Section 3 describes the Maslow's Pyramid of Needs and how it has influenced our solution. Section 4 describes the Test Automation's Pyramid of Needs (TAPN), proposed in this work. Finally, Section 5 presents our conclusions and future works.

## 2. Related Works

The main objective of *Test Process Support Models* (TPSM) is to provide foundation for software testing process improvement observing and mapping best practices in testing activities. TPSMs are also known as Test Maturity Models and most of them were developed in late 90's [Kulkarni, 2006].

One of the greatest challenges in software development is to ensure maximum quality with minimum cost. Thus, software testing has become indispensable in software development. A TPSM helps to standardize the testing process, increase software quality and decrease production cost, which means to support software developed to archive the expected results through planned activities, and minimize the impact in the introduction of new technologies [Ramler e Wolfmaier, 2006].

In the next subsections, we discuss some TPSM available at technical literature and how they deal with software test automation.

### 2.1 Test Automation Improvement Model (TAIM)

*Test Automation Improvement Model* (TAIM) [Eldh et al., 2014] is an ongoing work that proposes an improvement model focused specifically on test automation. It is composed by 10 key areas (KA – comprising practices that handle with different aspects of the test automation process) and one general area (GA). Each KA may implement different aspects provided in the GA. The KA and GA presented in the TAIM are:

- **GENERAL AREA:** consists of Traceability; Measurements indicators;

Analysis tools; Standards definitions.

- **KEY AREAS:** (1) Test management; (2) Test requirements; (3) Test specification; (4) Test code; (5) Test automation process; (6) Test execution; (7) Test verdicts; (8) Test environment; (9) Test tools, and (10) Fault/Defect Handling.

The TAIM was partially validated using KA #4 (Test code) in [Eldh et al., 2004]. The author established "an initial measurable point that could provide valuable insight" for the model. Although TAIM brings relevant contributions in terms of *key areas* in test automation, it does not provide guidelines of how organizations could have better chances to succeed in test automation projects. Instead, TAIM aims to provide objective metrics that can be used to evaluate a test automation initiative. It is important to emphasize that according to its authors, TAIM is still an on going work and all information came from its introductory paper.

### 2.2 Test Maturity Model Integration (TMMi)

Software industry has largely accepted the *Capability Maturity Model Integration* (CMMI) as guidelines to improve the software development processes. The *Test Maturity Model Integration* (TMMI) is a framework developed by the test community as a complement of CMMI and brings the guidelines to improve specifically the testing process [Van Veenendaal, 2010].

TMMi comprises five maturity levels: (1) Initial; (2) Managed; (3) Defined; (4) Measured; (5) Optimization. To ascend to upper level, an organization must completely satisfy the requirements to the previous one. Additionally, each level is composed of process areas (PA's) with generic and specific goals related to different aspects of testing process. Goals are generally used as milestones to indicate the accomplished state of a PA. Generic goals are called this way because the same goal statement can be reused in multiple PA's, while specific goals are designed to specific PA's .

Test automation is not explicitly addressed in TMMi. It means that none of the practices was designed specifically to guide a test automation initiative. Despite this, test automation is recommended in many sections as an alternative that should be taken into account by the organizations. TMMi classifies test tools as *supporting resources* that may be used within several process areas. For instance, in Sub Practice 2.2 (Define Test Approach), the test approach is defined and test automation is presented as an alternative to execute regressing tests.

### 2.3 Test Process Improvement (TPI) Model

*Test Process Improvement* (TPI) model was developed based on the practical experiences of testing process development and assumes that a testing process has many aspects that should be address individually. In TPI model, these aspects are called *key areas* and each of them is focused on a specific subject, such as test strategy, test environment or life-cycle model. In addition, the accomplishment of each *key area* is verified through *levels of maturity* that show how mature the process is in a specific aspect (*key area),* which is made through the use of *checkpoints* (requirement). The more checkpoints are satisfied; more mature is the process at that level.

TPI model does not consider that all *key areas* have the same important for the performance of the whole process and some level of dependency may be observed between them. Therefore, a *Test Maturity Matrix* (TMM) is used. In it, rows represent *key areas* and they are labeled according to the TMAP [IQUIP, 2004] methodology in:

*Life cycle (L)* of test activities related to the development cycle, good *Organization (O),* the right *Infrastructure* and *tools (I),* and usable *Techniques (T)* for performing the activities. Columns represent *levels of maturity* from A to D and each higher level improves its prior in terms of time, saved money and quality. In addition, 13 scales of test maturity are used in order to address the relation between key areas: *Controlled* (1-5) for the control of testing process; *Efficient* (6-10) efficiency of the testing process and; *Optimizing* (11-13) continuous improvement of the testing process.

The TPI model addresses test automation as result of a test tool and reserves a *key area (Test tools)* with three possible *maturity levels*:

- A: Planning and control tools
- B: Execution and analysis tools
- C: Extensive automation of the test process

The test process quality is measured according the test tool use (how and what). Although this approach can be useful to organization that already uses some kind of test tool, offering no help for those that will start automated tests from scratch.

### 2.4 Automated Test Generation (ATG-TPI)

Recently, [Heiskanen et al., 2010] proposed a modified version of TPI named ATG-TPI (Automated Test Generation). It aims to complement the original TPI model to provide support for the assessment of a testing process, in terms of how mature it is when starting to use automated test generation practices, and how the organization can evolve in this testing field. ATG-TPI assumes that organization has achieved certain level of maturity and it is capable to produce *the necessary basis* (test artifacts) to start the test generation process. ATG-TPI introduced four *key areas* mainly related to build and maintain *test models*: modeling approach, use of models, test confidence, and technological and methodological knowledge. In ATG-TPI, modeling tests means to translate them into a computable form.

It also made changes in existent *key areas* and maturity levels. Most of these changes are related to introduce modeling activities along the process and to define regular test automation as an independent maturity level.

### 2.5 Test Process Improvement – Brazilian Model (MPT.BR)

MPT.BR is a maturity model developed by a Brazilian committee (comprising universities, organizations, and government) based on other reference models, such as TPI and TMMi. Its focus is on the testing process improvement using best practices related to product's life cycle [Softex Recife, 2011].

MPT.BR adopts five maturity levels to evaluate the testing process improvement: 1) Partially managed; 2) Managed; 3) Defined; 4) Defect prevention and; 5) Automation and optimization. Each level comprises process areas (PA's) that represent different activities into the testing process. Additionally, each PA should meet practices in order to allow a software organization ascend to an upper level. Such as in TMMI, a practice can be generic or specific. Same generic practices can be used in different PA; instead, specific practices are designed to specific aspects of the testing process.

MPT.BR addresses test automation in the maturity Level 5 through two process areas: *ATE* (*Test Automation and Execution*) and *GDF* (*Tools Management*). *ATE* comprises six specific practices from *ATE1* to *ATE6*. They are related to the definition

of test automation objectives, test cases selection, cost assessment, framework selection, incident management, monitoring and return of investment. *GDF* offers guidelines for selection, management, deployment and organization of test tools.

MPT.BR offers great contributions to the formalization of a test automation process. However, it does not provide a guide for organizations in an early stage of test automation.

### 2.6 Summary of TPSMs regarding Test Automation

Most of early TPSMs offer little documentation and it is difficult to visualize how they address test automation or how a software organization can be prepared to implement it. The main difference between TPSMs and the model proposed in this work is that while the TPSMs are concerned to improve the testing process, the proposed model is focused on helping organizations to fulfill the requirements needed to increase the success of their test automation strategies by gradually handle the economic, technical and management aspects. Thus, the proposed model can be applied in cooperation with TPSMs in a software organization.

Therefore, the objective of this work is to provide a framework that aggregates different test automation practices to be used in software projects. In order to provide evolving stages, the proposed model has also adopted a hierarchical approach that could be gradually achieved by organizations. We organize these stages according to the requirement needs of an organization/software project making a relationship with the degrees of needs that comprise the Maslow pyramid, described in the next section.

## 3. Maslow's Pyramid of Needs

Abraham Maslow was a fruitful psychology researcher mainly interested on human motivation. He proposed the *Theory of Human Motivation* [Maslow, 1943], which describes that humans are motivated according to the fulfillment of some needs, represented by a five level *Pyramid of Needs* showed in (Figure 1). Maslow believed that every human is naturally motivated to archive self-actualization. However, to reach this goal, he/she has to gradually achieve states of motivation by satisfying different kinds of needs. He listed five groups of needs:

- **Physiological needs:** these are the most basic needs and they are related with life support and primitive desires, such as breathing, eating, drinking, sleeping, and having sex.
- **Safety needs:** the perception of safety is related to feelings such as stability and protection. Since first years of life, humans need to feel safe. In the early years, parents are the safety reference, lately replaced by other cultural institutions that supply it such as a good job, a house, good neighborhood and retirement plan.
- **Belonging/Love needs:** humans are naturally social and they have need to care and be cared by others, to feel part of a community. This is represented by any emotional relationship, such as marriage or friendship or be part of a church or gang.
- **Esteem needs:** once a human feels part of a group, the need to be recognized by the other members grows. It may mean a search for fame, attention, reputation and even dominance.
- **Self-actualization needs:** according to Maslow, after fulfilled the previous needs at least to a considerable extent, a human can finally concentrate on developing

its potential at the higher level. At this point, ideals become much more significant and can lead to charity, political involvement and achieving dreams.

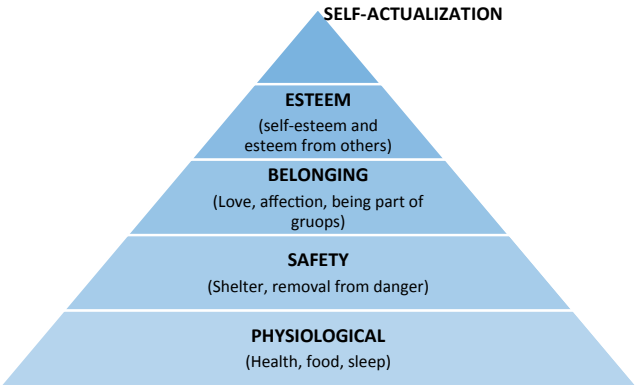


**Figure 1. Maslow Pyramid of Human Needs [Maslow, 1943].**

The Maslow Pyramid's contributions to the Test Automation's Pyramid of Needs (TAPN) proposed in this work (presented with details in the next section) are:

- **The pyramid representation:** The lower is the pyramid's level; wider is its area. It was not a simple coincidence that Maslow chose it to represent his theory. He believed that the more primitive is the need; stronger and more important/essential it would be for a person. The same principle is applied to TAPN. The lower levels contain the most essential practices to the success of a test automation strategy. As the software organization evolves in the pyramid's levels, it would be adding new practices that contribute to consolidate the test automation strategy.
- **Partial compliance:** Maslow also considered that people might perceive each of these needs in different level of importance and it can disrupt the fulfillment of the needs. The TAPN does not aim to be a rigid model. Instead, it may be a point of reference to software organizations.

## 4. Test Automation's Pyramid of Needs (TAPN)

### 4.1 TAPN's Definition

The objective of this work is to develop Test Automation's Pyramid of Needs (TAPN) structured in a model that can be used by software organizations to evaluate and improve their test automation strategies.

In essence, a test automation strategy follows the same life cycle of any other type of initiative in a software organization, with well-defined phases to pass through such as planning, executing, evaluating and tuning. Therefore, it should be run as such [Pettichord, 1999]. Hohn (2011) has emphasized that *economic*, *technical* and *managerial* aspects influence testing processes, which should be well planned since the early beginning.

As core requirements, TAPN should be:

- **Practical oriented**: it is designed from practitioner's point of view and aims to be objective and bureaucracy.

- **Test automation-driven:** take in account the specific characteristics, requirements, limitations, and factors of success.
- **Gradual:** offer gradual stages by which the organization achieves the main goal.
- **Easy to follow:** by organizations that still had not started a test automation strategy; and by organizations that already had started it into their projects.
- **Independent:** the TAPN are not tied to any existing model or tool and has no pretension to replace any of them, instead, it could be used as a complement;

In the conception of TAPN model, we have borrowed many characteristics from existing layered models already consolidated, such as TMM, TMMi, MPT.BR, TPI, ATG-TPI (cited in Section 2), which are focused on the software testing process improvement. However, they do not address test automation with enough details. Eldh et al. (2014) proposes to fulfill this gap with the TAIM, which provide interesting issues about test automation. Some frameworks developed also influenced the TAPN to deal with specific test automation aspects, such as test design [Rankin, 2002] and execution [Kim et al., 2009]. There are also those that propose frameworks to help organizations to apply test automation in different scenarios [Cervantes, 2009][Song et al., 2011].

### 4.2 Factors of Success in Test Automation Strategies

There seems to be a consensus among researchers and practitioners that software organizations mainly search for cost reduction and testing efficiency when investing in test automation [Ramler e Wolfmaier, 2006][Taipale et al., 2011][Rafi et al., 2012]. In order to develop the TAPN model, we have investigated which factors contribute to the success of a test automation strategy.

As first step, we have identified some unrealistic expectations about test automation. Then, we could exclude them from the proposed model's scope:

- Although there is a consensus that test automation can offer some benefits, it is definitely not a "silver bullet" to solve all the problems with testing activities [Hoffman, 1999][Karhu et al., 2009][Pettichord, 1999][Hayes, 2004].
- Manual testing cannot be 100% replaced by test automation, as well as human intervention cannot be totally eliminated from the testing process [Alam, 2007].

Going a little further, we have compiled a list of success factors extracted from the technical literature. They are listed (sorted by the number of references where each factor was identified) in Table 1.

Researchers and practitioners seem to agree that the success of test automation strategies strongly depends on early planning. In fact, Table 1 presents a hint about the importance of each factor in the test automation strategy. We have assumed that more referenced factors have stronger impact on the automation strategy and should have higher priority than others. In addition, each factor is influenced by three kinds of aspects: *economic*, *technical* and *management* [Hohn, 2011].

- **Economic aspects** are related to time and required resources (human and material); testing tool, hardware, labor and maintenance are possible costs in test automation [Ramler e Wolfmaier, 2006].
- **Technical aspects** are related to techniques, methods and tools; test case/data generation methods and testing tool acquisition criteria are typical technical issues in test automation.
- **Management aspects** are related to methods used to monitor, control and

evaluate testing activities; test logging, report generation and optimizing can be included in this category.

**Table 1. Factors of Success in Test Automation Initiatives**

| #ID | Factor of Success | Description | Reference |
|---|---|---|---|
| F01 | Early Planning | Researchers and practitioners seem to agree that test automation demands a careful planning. It should include, but not limited to, goal definitions, economic study (ROI) and resource acquirement processes. | [3][4][5][7][8][9] [11][12][13][14] [15][16][17][18] |
| F02 | Dedicate and skilled team | A common mistake about automation is that human intervention will be 100% dispensable or that existent test team can easily dedicate few hours in the automation process. Literature shows that test team and automation team should be distinct because they execute distinct works. Automation team must handle with tasks like test tool acquisition, configuration and installation, which demands high level of knowledge about the system and the environment. In addition, professional test tools can be very complex and may need "experts" to have them properly working. | 0[3][4][7][8][9] [10][11] [16][17] |
| F03 | Quality Control | It is critical to early identify if automation is having a positive impact in the test process. Therefore, it means to monitor results and make adjustment in order to attend the expectations. Quality control means to define performance metrics, establish reference values and to monitor if they are being achieved. | [2][10][11] [14][17] [19][20][21][22] |
| F04 | Tool Acquisition | Tool vendors used to present their solutions as silver bullets to all kinds of situations. However, literature shows that the day after story is very different in most cases. Issues such as learning curve, maintenance effort and customization must be taken in account to avoid future disappointments. | [1][4][6][9][11] [14][16][17] |
| F05 | Realistic goals | As previously mentioned, there is not 100% automation. However, according to the literature, organizations used to have a simple goal when deciding to automate their tests: to automate as much tests as possible. It can lead to unrealistic expectations and frustration. Establishing realist goals are related to differentiate between what is wanted to automate from what is feasible to automate. | [6][9][10] [11][17][18] |
| F06 | Maintainability | Test automation frequently require substantial maintenance; and test artifacts may change during the development cycle. Choosing a test tool easy to maintain and upgrade can save much money and time. | [2][4][9][10] [11][14] |
| F07 | Scalability | It is plausible to think that test artifacts tend to grow in number and complexity along the time, which might demand more resources (hardware and personal) in order to support the new demand. | [4][10][11] [13][17][18] |
| F08 | Reusability | The technical literature shows that organizations invest on test automation seeking for saving money. However, it is a long-term investment and will depend on how resources will be managed. The more they can be reused the bigger the return of investment. In addition, the papers also show that reusability has the potential to double the productivity when correctly applied. | [2][4][8][9][17] |
| F09 | Well Defined Test Process | A possible definition for test automation can be - the mechanical representation of a manual test activity. Therefore, automation depends on the correctness of manual processes and can have unpredictable results when they are not well defined or do not exist at all. | [3][7][10][11] [14][17] |
| F10 | Cost/Benefits Evaluation | Test automation demands high initial investment and long-term return. So, organizations should carefully evaluate if it is really worth to automate their tests. According to the literature, the smaller is the project, the less the need to automate test activities. | [2][7][10] [14][18] |
| F11 | Manageability | Even automated tests need to be managed to ensure they are achieving their objectives and fulfill quality standards. The establishment of performance indicators, monitoring techniques and report analysis are examples of managing activities. | [6][7][10] [14][17][18] |
| F12 | Resource Availability | Organizations must be sure that all necessary resources to automate their tests are available. Finding out that there are not enough resources to continue the project at its middle is tragic. | [6][10][17][18] |
| F13 | Testability Level of the SUT | Testability level indicates if a software has been designed or not to facilitate testing. Testability increase test design efficiency and facilitate automation. | [5][8][11][12] |
| F14 | Feasibility Assessment | It is essential for organizations to know in advance whether the test automation is feasible or not for their needs. They must be sure if their projects are technical and economic feasible. | [4][11][15][18] |

**Key:**
[1] ➔ [Lee et al. 2012] | [2] ➔ [Kasurinen et al., 2010] | [3] ➔ [Bertolino, 1991] | [4] ➔ [Karhu et al., 2009] | [5] ➔ [Alam, 2007] | [6] ➔ [Eldh et al. 2004] | [7] ➔ [Young, 2004] | [8] ➔ [Taipale et al, 2011] | [9] ➔ [Rafi et al, 2012] | [10] ➔ Hayes, 2004] | [11] ➔ [Pettichord, 1999] | [12] ➔ [Obele and Kim, 2014] | [13] ➔ [Bansal et al., 2013] | [14] ➔ [Bach, 1997] |

[15] ➔ [Oliveira et al., 2006] | [16] ➔ [Meng, 2011] | [17] ➔ [Graham and Fewster, 2012] | [18] ➔ [Ramler e Wolfmaier, 2006] | [19] ➔ [Mpt.Br, 2011] | [20] ➔ [Andersin, 2004] | [21] ➔ [Heiskanen, 2010] | [22] ➔ [Heiskanen, 2010] | [22] ➔ [Veenendaal, 2010]

For instance, testing tools should be acquired according to some criteria (*technical* aspect) and generate costs to the project (*economic* aspect).

### 4.3 TAPN's Structure and Levels

TAPN is inspired on Maslow's Pyramid of Needs that comprises five levels of human needs from bottom to top. Thus, TAPN also proposes five levels of test automation practices that can be achievable by an organization: *Execution*, *Generation*, *Management*, *Optimization* and *Reusability* (Figure 2). Each level implies the accomplishment of a set of practices that support one or more success factors that directly affect a test automation project. The model is not rigid and organizations can satisfy different practices distributed along the pyramid's level. This flexibility helps software organizations to evaluate what they need to do in order to increase their chances of running a successful test automation project.

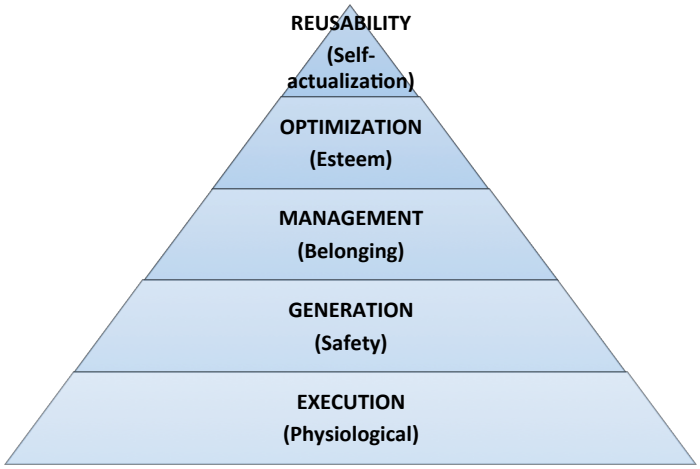


**Figure 2. TAPN's Levels.**

The Figure 3 represents the architecture of the model. Each level comprises practices and sub-practices that are influenced by economic, technical and management factors. The importance of each factor may vary according to the practice's characteristics.

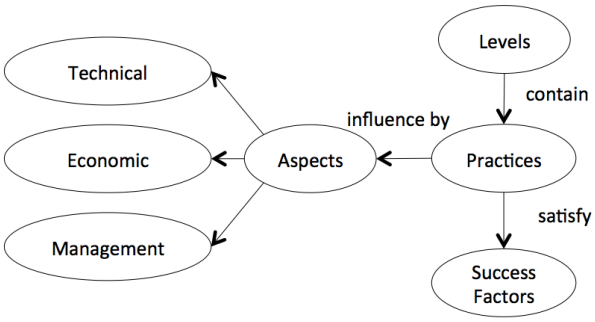


**Figure 3. TAPN Model's Architecture.**

The description of each level proposed for TAPN and its association with the levels that comprises the Maslow's Pyramid of Need are described below.

- **LEVEL 1: EXECUTION (*PHYSIOLOGICAL*):** In test automation, test execution is the most basic requirement (need). Lee et al. (2012) found that for the majority of software organizations, test execution would be the first goal when they start an automation project. This level provides the minimum resources for test automation in an organization. The practices for Level 1 are listed in Table 2.

**Table 2. Practices for Level 1: Execution.**

| Practice | Purpose | Factor of Success |
|---|---|---|
| Select tests to automate | Identifying which tests will be automated and their demands. | F01, F13 |
| Acquire the appropriate test tool and computational resources | Ensuring that the test tool will be able to automate the selected tests and all necessary computational resources are available. | F01, F04, F12 |
| Deliver test environment | Ensuring that test environment (*testware*) is correctly configured and operational. | F01, F12 |
| Define test team | Selecting the people who will be exclusively dedicated to the automation process. | F01, F04, F12 |
| Define an incident management methodology | Selecting a bug-tracking tool and define the incident recording process. | F06, F11 |
| Adopt a coding standard to write test scripts | Decreasing the time spent on future maintenances of the scripts. | F03, F06 |
| Verify test artifacts | Ensuring that test artifacts are validated and available for all involved in the project. | F03 |
| Validate test scripts | Ensuring that all test scripts are running correctly. | F03 |
| Evaluate results | Comparing reference values with test results. | F03 |

- **LEVEL 2: GENERATION (*SAFETY*):** In test automation, human intervention adds insecurity and should be avoided [Taipale et al., 2011]. In the previous level, an organization is capable of automatically running tests, but it is still widely dependent of human intervention, what may add risks and errors to repetitive test activities, such as test data generation and test case generation. This level represents a set of practices that should be followed by organization to generate automatically some test artifacts, such as test cases and data. The practices for Level 2 are listed in Table 3.

**Table 3. Practices for Level 2: Generation.**

| Practice | Purpose | Factor of Success |
|---|---|---|
| Select tests to be generated | Identifying the most suitable tests to be generated. | F01 |
| Build a test model | Ensuring that generated test will satisfy the test requirements | F01, F13 |
| Select test generation technique | Identifying the best generation methods for selected tests. | F01 |
| Keep test tool upgraded | Assuring that test tool is capable to support the test generation method. | F04, F12 |
| Keep computational resources upgraded | Assuring that computational resources can support test tool upgrades. | F12 |
| Keep team trained | Assuring that test team is productive and efficient. | F02 |
| Make use of | Test automation allows execution of high number of tests and | F07 |

| | | |
|---|---|---|
| automated oracles | manual evaluation can act as a bottleneck. The use of an automated test oracle can improve the speed of the test evaluations. | |

- **LEVEL 3: MANAGEMENT (*BELONGING*):** At third level, Maslow emphasizes the human needs to care and to be cared in a group. In test automation, an organization on Level 2 start to generate much more test artifacts that should be managed in order to assure the tests quality and comprehensiveness [Eldh et al., 2004]. This level represents a set of practices that helps the organization to take care (monitors and validate) of test artifacts and demands generated by the automation. The practices for Level 3 are listed in Table 4.

**Table 4. Practices for Level 3: Management.**

| Practice | Purpose | Factor of Success |
|---|---|---|
| Define automation plan | Making clear the organization's automation objectives, defining execution strategy, necessary resources, budget and schedule. | F01, F05, F13, F14 |
| Execute cost/benefit assessment | Analyzing if there will be gains (money and time) in replace the manual tests by automatic tests. | F01, F10 |
| Execute risk assessment | Knowing the risks to plan future countermeasures and avoid unexpected surprises during the project. | F01, F14 |
| Define criteria to select tests to automate | Since 100% of automated test is not feasible, defining criteria to select tests to be automated. The statement of objectives should be taken into account when defining the selection criteria. | F01, F07, F08, F14 |
| Define test tool selection criteria | The selection of a test tool has impact in the success/failure of a test automation project. The organization should choose the most appropriate tool to meet its needs within the expected costs. | F01, F04 |
| Seek adherence to the test process | It is not reasonable to think that test automation can completely cover testing process. Instead, it should be part of the process, improving activities that are better executed by a machine. | F09 |
| Define criteria to acquire computational resources | Computational resources must fully support the features of the chosen test tool. | F01, F12 |
| Define test team roles and skills | It is common to recruit personnel from the manual test team to the automation project. This practice aims to identify the necessary skills and define roles that must exist within the test team. | F01, F02 |
| Keep a team training program | Keeping the test team updated with the new technologies reduces the time to roll out new features in the test tool. | F01,F02 |
| Define metrics and reference values | Metrics allows organization to monitors a particular characteristic of an automation project. | F03 |
| Define a performance monitoring strategy | Metrics alone are not useful. They have to be monitoring and analyzed to guide the progress of the automation project. | F03 |
| Define maintenance plan | Either the SUT as the test tool may change during the project. So, planning how these modifications will be deployed may save time and resources. | F06, F07 |
| Define test reports | Reports may help to detect problems and take decisions. They should be designed to take maximum advantage of the defined metrics. | F03 |
| Keep a repository of incidents | Storing incidents may help organization to learn from them in the future and avoid repeating mistakes. | F03 |
| Keep register of used test tools | Registering the experiences on using different test tools may guide acquiring decisions in projects to come. | F03, F04 |
| Keep a database of human resources | Human resources are expensive to recruit and train. A database of personnel information supports the reuse of staff members in a more efficient manner. | F02, F03 |
| Keep a database of test artifacts | Test artifacts can be reused as a whole or as a basis for other constructions. Keeping a database of them may save time and | F06, F07, F08 |

| | | |
|---|---|---|
| | efforts when designing artifacts to a new project. | |
| Keep a script library | Code reusing allows the use of the same segment of code in multiples applications. This practice provides a script repository aiming the reduction of efforts on future projects. | F06, F07, F08 |
| Adopt a management tool | As more control you have of testing process, the faster decision making. Management tools can help test managers to monitor test activities and their results, organize personnel tasks and maintain a communication channel between technical staff and high administration. | F11 |
| Define a communication strategy | A good communication can enable better project visibility, coordinate personal efforts, reduce risks and improve understanding among diverse stakeholders. | F11 |
| Follow the schedule and keep spending within budget | In order to avoid frustrations about the test automation, organization must do their best to follow the schedule and keep expenses within the budget. | F10, F11 |
| Make use of standards | The use of standards brings consistency and stability to test artifacts and code. Coding standards make scripts easily to maintain and to be reused in projects to come. | F06, F07, F08 |

- **LEVEL 4: OPTIMIZATION (*ESTEEM*):** On Maslow's Pyramid, this level is dedicated to the human need to be recognized and self-improvement. In test automation, this level represents the practices that helps organization to self-improve its automation projects in order to reduce costs, and facilitate adaptations to new environment scenarios [Andersin, 2004][Taipale et al., 2011]. The practices for level 4 are listed in Table 5.

**Table 5. Practices for Level 4: Optimization.**

| Practices | Purpose | Factor of Success |
|---|---|---|
| Analyze most common incidents and propose solutions | Analyzing incidents history may help to detect bad behaviors, harming standards, suggest improvements and solution to problems. | F03, F08 |
| Propose improvements to the test tool for manufacturers | Registering experiences with test tools may result in improvement suggestions that can be proposed to manufacturers and facilitate even more the test process. | F03, F04, F07, F12 |
| Establish a preventive maintenance plan | Preventive maintenance can cost less than corrective maintenance. A preventive plan can result in cost reduction and more efficiency to the test process, once there will be less stops. | F03, F06, F11 |
| Establish a continuous training program to the test team | Test team must be always updated in terms of technology and process activities in order to decrease human errors and increase productivity. | F02 |
| Build an evaluative report for each completed project | Having an evaluative report at each project ending can offers important information such as cost/benefits, evidence of bottlenecks and total time consuming. | F03, F10, F11, F14 |

- **LEVEL 5: REUSABILITY (*SELF-ACTUALIZATION*):** Maslow defined self-actualization as the last level of the pyramid, a goal pursued by every human. In test automation, the initial investment is high and should be amortized as much as possible. According to many studies [Karhu et al., 2009][Burnstein et al., 1996][Eldh et al., 2004][Bansal et al., 2013], reusability is the best way to justify these investments. This level represents a set of practices that helps an organization to maximize the reusability of resource and experiences in test automation among

different projects. The practices for Level 5 are listed in Table 6.

**Table 6. Practices for Level 5: Reusability.**

| Practices | Puporse | Factor of Success |
|---|---|---|
| Reuse test artifacts | Test artifacts such as test cases and test data may be expensive to produce in terms of time and resources. Reusing artifacts in different projects can save money and decrease testing time. | F08, F10, F12 |
| Reuse test tool and computational resources | Using a test tool once use to be not worth for organizations. They are normally expensive and hard to configure and maintain. So, organizations should implement policies to reuse test tools in different projects. | F04, F08, F12 |
| Take advantage of the same test team in different projects | Taking advantage of the accumulated knowledge and experience in the test tool, reducing the tests start time and save with new hires. | F02, F08, F10 |

## 5. Conclusions and Future Works

So far, and for many years ahead, it seems that organizations will face the big challenge of balancing quality and costs. Testing is still the best way to assure some level of quality, but it is expensive and takes much time. In this scenario, test automation have recognized as a good alternative to save costs and increase efficiency in software testing. However, it also has been a source of frustration and failures to many organizations.

According to Lee et al. (2012) and Kasurinen et al. (2010), most of software organizations prefer not to automate tests or fail to automate. Moreover, according to these works, the success rate is below than 50%. The technical literature does not point a single cause for this phenomenon, but a union of many factors may influence on the success of a test automation project.

This work proposes a support model to increase the chances of success on developing, deploying and maintaining test automation initiatives. To achieve this, in a first moment, we identified and selected 14 success factors through a literature review that may positively influence on the execution of test automation projects. Then, these factors were used as reference to propose a set of good practices, which were hierarchically organized in levels of a pyramid (Test Automation Pyramid of Needs – TAPN). Thus, the practices can be gradually accomplished by the organizations.

As future works, we intend to validate our model by applying in software organizations through an assessment process. We expect to be able to classify their test automation initiatives according to the TAPN's levels and identify possible improvements.

During this research, we have found that the success in test automation does not come to those that automate more, but to those that automates better.

## 6. Acknowledgments

The authors would like to thank CNPq, FAPEAM, and INDT for the financial support for this research.

## 7. References

Alam, M. N., "Software test automation myths and facts", IMI Systems Inc., Dallas, 2007.

Anand, S. et al. "An orchestrated survey of methodologies for automated software test case generation". *J. Syst. Softw.* 86, 8, 2013, pp. 1978-2001.

Andersin, J., "TPI – a model for Test Process Improvement", Department of Computer Science, University of Helsink, Helsinki, 2004.

Bach, J., "Test Automation Snake Oil", In 14th International Conference and Exposition on Testing Computer Software, Washington-DC, 1997, p. 6.

Bansal, A., Muli, M., Patil, K., "Taming complexity while gaining efficiency: Requirements for the next generation of test automation tools", In: AUTOTESTCON, pp. 1-6, 2013.

Bertolino, A., "An overview of automated software testing", Journal of Systems and Software, vol. 15, no. 2, pp. 133-138, 1991.

Brunstein, I., Suwanassart, T., Carlson, R., "Developing a testing maturity model for software test process evaluation and improvement", In International Test Conference, 1996, pp. 581-589.

Cervantes, A., "Exploring the use of a test automation framework", In Aerospace conference, 2009 IEEE, 2009, pp. 1-9.

Eldth, S., Andersson, K., Wirlund, K., "Towards a Test Automation Improvement Model (TAIM)", in International Conference on Software Testing. Verification and Validation Workshops, 2014, pp. 337-342.

Graham, D., Fewster, M., "Experiences of Test Automation: Case Studies of Software Test Automation", Addison-Wesley Professional, 2012.

Hayes, L. G., "The Automated Testing Handbook". Software Testing Institute, 1995.

Heiskanen, H., Maunumaa, M. and Katara, M.Test Process Improvement for Automated Test Generation. Tampere: Tampere University of Technology, Department of Software Systems, 2010.

Hoffman, D., "Cost benefits analysis of test automation". STAR West, v. 99, 1999.

IQUIP, "TMap", TMap - Sogeti Nederland B.V., 2004, Available at: <http://www.tmap.net>. Accessed at March 10th, 2015.

Höhn, E. N., KITest: A Framework of software testing process knowledge and improvement (in Portuguese), Dissertation presented in Institute of Mathematical and Computing Sciences - ICMC-USP, July, 2011.

Karhu K., Repo T., Taipale, O., Smolander K. 2009. "Empirical Observations on Software Testing Automation". In Proceedings of the 2009 International Conference on Software Testing Verification and Validation (ICST'09). IEEE Computer Society, Washington, DC, USA, pp. 201-209.

Kasurinen, J., Taipale, O., Smolander, K., "Software test automation in practice: empirical observations", Advances in Software Engineering, 2010.

Kim, E. H., Na J. C., Ryoo S. M., "Implementing an effective test automation framework," in Computer Software and Applications Conference, 2009. COMPSAC'09. 33rd Annual IEEE International, 2009, pp. 534-538.

Kulkarni, S., "Test Process Maturity Models–Yesterday, Today and Tomorrow", in Proceedings of the 6th Annual International Software Testing Conference, Delhi, 2006.

Lee, J., Kang, S., Lee D., "Survey on software testing practices", IET software, vol. 6, no. 3, pp. 275-282, 2012.

Lott, M., Fecko, C.M., "Lessons learned from automating tests for an operations support system", In: Software: Practice and Experience, pp. 1-23, 2002.

Maslow, A. H., "A theory of human motivation", Psychological review, vol. 50, no. 4, p. 370, 1943.

Meng, X. F., "Analysis of software automation test protocol", In Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on, 2011, pp. 4138-4141.

MPT.Br, "Melhoria do Processo de Teste Brasileiro - MPT.br", SOFTEX RECIFE, 2011.

Obele, B. O., Kim, D., "On an Embedded Software Design Architecture for Improving the Testability of In-vehicle Multimedia Software", In IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2014, pp. 349-352.

Oliveira, J. C., Gouveia, C. C., Quidute-Filho, R., "Test Automation Viability Analysis Method", In Proceedings of the 7th IEEE Latin American Test Workshop (LATW'2006), 2006, p. 6.

Pettichord, B., "Seven steps to test automation success", Proceedings of the International Conference on Software Testing, Analysis, and Review, 1999.

Rafi, D. M., Moses, K. R. K., Peterson, K., Mäntylä, M. V. "Benefits and limitations of automated software testing: systematic literature review and practitioner survey". In Proceedings of the 7th International Workshop on Automation of Software Test (AST '12). IEEE Press, Piscataway, NJ, USA, 2012, pp. 36-42.

Ramler, R., Wolfmaier, K., "Economic perspectives in test automation: balancing automated and manual testing with opportunity cost", In Proceedings of the 2006 international workshop on Automation of software test, 2006, pp. 85-91.

Rankin, C., "The Software Testing Automation Framework", IBM Systems Journal, vol. 41, no. 1, pp. 126-139, 2002.

Song, H., Ryoo, S., Kim, J. H., "An Integrated Test Automation Framework for Testing on Heterogeneous Mobile Platforms", In ACIS International Symposium on Software and Network Engineering (SSNE), 2011, pp. 141-145.

Taipale, O., Kasurinen, J., Karhu, K., Smolander, K., "Trade-off between automated and manual software testing", International Journal of System Assurance Engineering and Management, vol. 2, no. 2, pp. 114-125, 2011.

Veenendaal, E. Van, "Test Maturity Model integration (TMMi)", TMMi Foundation, 2010.

Young , D., "Test Automation: An architectured approach", Schwab Performance Technologies, Raleigh, 2004.