# Applying the Test Maturity Assessment (TMA) & Test Process Improvement (TPI) for the Improvisation of Test Maturity Models

**Fanish Lal S.[1] VR. Nagarajan[2]**
[1]Research Scholar [2]Assistant Professor
[1,2]Department of Computer Science & Engineering
[1,2]Sree Narayana Guru College, Coimbatore - 641 105, India

*Abstract*— Many of the software testing procedures and practices are not performed properly that leads to pessimistic results. The immaturity of the testing procedures will be ineffective in predicting the defects, cost and planning activities in testing. The state of the art practices are to be noted to merge all the kinds of test maturity models that are developed by various researchers for examining the drivers, challenges and outcomes of TMA/TPI. Various models are developed to enhance the maturity by running the test maturity assessment (TMA) and test process improvement (TPI). The survey helps to bring the positive outcomes while examining and upgrading the test maturity model. To attain the above feature, a study is conducted based on Multi vocal Literature Review (MLR) to analyse the facts about TMA/TPI. A MLA is the method of grey literature for posting the blogs. The present survey tells how TMA/TPI is used to determine the various test maturity models, to synthesize the drivers, issues and advantages in improving the maturity model.

***Key words:*** Maturity, Testing Procedures, Test Maturity Assessment & Test Process Improvement

## I. INTRODUCTION

The practice of software testing may be different in their competence and effectiveness; it may be different between various organizations and the teams. The high quality software product is developed by the companies who are efficient and very effective. But many companies are not fully grown up and they are organized in a ad hoc manner. This type of undeveloped practices gives raise to pessimistic results, i.e. the method used in testing will not be able to identify the defects and it may acquire more cost and filled schedule [4]. Hence the software teams follow test maturity assessment (TMA) to examine the competence, efficacy, and quality of companies, testing practices. Following to this, the testing engineers and the higher officials execute test process improvement (TPI).

The various methods and structures are framed by the researchers and the software practitioners to perform TMA and TPI in a organized manner. Some of the approaches are prescribed in the book called "Improving the Test Process: Implementing Improvement and Change". This type of book serves as the foundation for the International Software Testing Qualifications Board (ISTQB) expert-level certification on TPI. Combining with testing practitioners, the involved various TPI projects have come to know that they are facing many disputes in performing TMA and TPI. These disputes include:

- The necessity of TMA and TPI are increased between the company and their team members.
- Before starting the project, accurate planning of TMA and TPI actions are to be done.
- The disputes must be identified before starting it and queries must be answered.
- The pros and cons of TMA and TPI have to be analyzed.
- Measuring the achievement of TMA and TPI actions.

These disputes can be resolved by identifying the software engineering team utilizing the concept of TMA and TPI in order to help the software engineers [1]. This team has noticed the Multivocal Literature Review (MLR) and the MLR is a method of systematic literature review (SLR) that includes data that is taken from multiple sources i.e., scientific and gray literature and such as blog posts and presentation videos.

Nowadays, MLRs that are associated to software engineering are growing rapidly. They are typically appropriate for scrutinizing TMA and TPI, as well as the method is motivated connected to both academicians and industrialists. This survey discovers most of the test maturity models. The review identified many test maturity models and more practical experimentations [11]. Many articles are published based on this topics called as "agile development and developer motivation" and they are described with detailed work precisely. Most of the researchers have considered the work on the existence of TMA and TPI but they omitted the concept of both the academic and gray literature.

A recent survey was proposed on SLR in 2016 and it identifies eighteen TPI methods to describe the quick progression of this practice in software testing. According to many studies TMMi and TPI are the most familiar and widely-used models in software testing. TMMi uses the concept of TMM and it is related to Capability Maturity Model (CMM) and CMMI, and it was developed in the year 1998. The new version of TMMi specification is 1.0 and that is prepared and published by the TMMi Foundation in 2012.

The remaining sections are illustrated as follows: section 2 describes the general procedure for TMA and TPI, section 3 describes the assessment procedure, section 4 describes the software practitioner involvement, section 5 describes the drivers, section 5 and 6 describes the conclusion and references.

 

## II. A GENERAL PROCEDURE FOR TMA & TPI

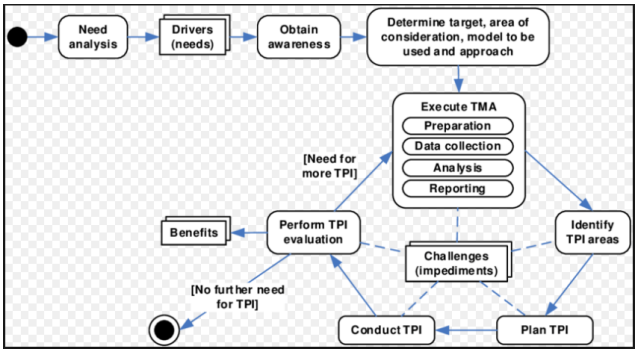


Fig. 1: A General Procedure for TMA & TPI

The general procedures and its framework are depicted in Figure 1 as the UML activity diagram. This procedure is simpler version in Test Process Improvement (TPI): A practical experimentation step-by-step procedure is provided to the structured testing: the process can be extended by utilizing the findings in the review [7]. Generally, the TMA-and-TPI begins with the requirement analysis phase to check its necessity by software team members or software engineers. The second step is to endorse alertness among other consumers and the management. The next thing is to examine the areas or the approaches used for TMA or TPI model. Another thing is, the TPI procedures are identified by starting the TMA procedures. Next phase is planning phase by team members or software engineers and they organize TPI and examines their results and its benefits. The process is repetitive when there is necessity of more TPI and it ends the process when the TPI is not required. The right procedures, examining the drivers, disputes and their advantages plays the vital role in this context.

## III. THE ASSESSMENT PROCEDURE

The MLR method uses the standardized procedure for SLRs in software engineering. The following questions can be addressed:

- What are the types of test maturity models have researchers and software practitioners have developed?
- What are the drivers utilized for TMA and TPI?
- What are the disputes of TMA and TPI?
- What are the advantages of TMA and TPI?

SLRs involve the combination of both gray literature and academic category and this incorporation is called as Multivocal Literature Reviews (MLR) in academic research in the year 1990 [14]. The main distinction between an SLR and MLR is SLRs takes the input only from academic peer-reviewed papers whereas MLRs takes the input sources from Gray Literature such as blogs, white papers, videos and web-pages. But MLRs identifies the requirement for "multiple" voices in spite of creating substantiation from academically obtained knowledge or sources. The MLR is elaborately defined as, "Multivocal literatures (MLR) are encompassed of all common accessible writings which is contemporary topic. This type of writings exemplifies the ideas or voices of different group of authors (academics, journalists, practitioners, local school districts policy centers, state offices of education, independent research and development firms, and others). The writings may occur in various forms. They reproduce diverse purposes, viewpoints, and data bases [9]. They represent diverse aspects of the given topic and combine diverse research or non-research methods".

The searches are performed in both the Google and Google Scholar databases. Our search methods or topics were

- Software test process improvement,
- Software test capability,
- Software test maturity, and
- Software test process enhancement.

To amalgamate the choice of opinion and experimental proof in the main sources with respect to the drivers, disputes, and advantages, qualitative coding (also called grounded theory) is used. A more detailed procedure of our MLR technique and qualitative coding is identified and solved the problems of the potential threats to our assessment's validity [5]. The main concept of maturing the software procedures called Test Maturity Assessment (TMA) and Test Process Improvement (TPI) are described in the next section.

### *A. Test Maturity Assessment (TMA)*

The Testing Maturity Model (TMM) of assessing is quite similar to the Capability Maturity Model and it was first developed by the Illinois Institute of Technology. Their aim was to use the concept similar to CMM and that provides a structural framework for evaluating the testing process maturity in an organization and that enhances the maturity by initiating the targets.

The five Levels are prescribed levels in the Testing Maturity Assessment Model and it is shown in table no 1.

| Level | Description |
|---|---|
| Level 1 – Initial | The outcomes are repetitive because of the usage of ad hoc methods by the organization and the quality becomes very poor |
| Level 2 – Definition | The definition of testing process are defined and the strategies, planning, methodologies and writing test cases are performed depending on the user needs |
| Level 3 – Integration | Example for this level is V model because the testing is incorporated into software life cycle. Here, the necessity for testing is because of risk management and the testing is done from independent and development domain |
| Level 4 – Management and measurement | The testing action occurs in every stages of the life cycle that comprises of requirement reviews and its design. Quality requirement is promised one for all the products of the company |
| Level 5 – Optimization | The process of testing is enhanced at every loop and this is attained with the help of tools and also aims to prevent the defects by lifecycle instead of detecting it. |

Table 1: Levels of Test Maturity Assessment Model

The set of goals and the processes are defined from level 2 that is used for practitioners and the TMM has been

 

substituted by the Test Maturity Model integration and it is handled by TMMI Foundation [2].

### *B. Test Process Improvement (TPI)*

The process of testing can be viewed from different angles by the test process improvement model i.e. the utilization of testing tools, testing specification methods and reporting. These factors are called as key areas in TPI model [10]. The maturity level is classified based on the key areas. All the given key areas are not uniformly imperative for the whole testing process performance and there may have dependency among the diverse key areas and their levels.

Hence Test Maturity Matrix is employed to validate the classification levels objectively, and also to check whether the how many Checkpoints are allocated to each level [6]. A checkpoint is a given need and the process is categorized at each level when the check points are conveyed by a testing process to the particular level.

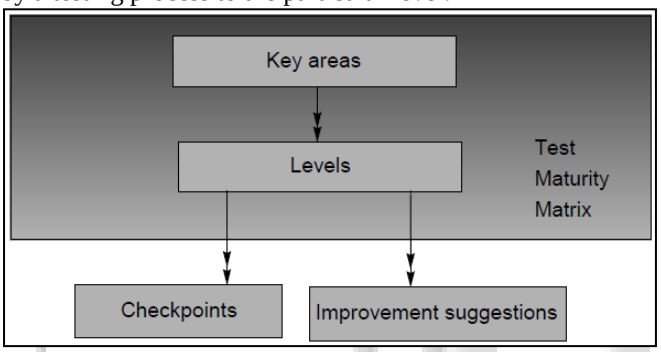


Fig. 2: TPI Model

In spite of associating the current testing process situation the additional levels and key factors are needed to define the situation and the intermediary steps to the particular situation. For the maturity level, an additional suggestion enhancement can be provided to attain the level of maturity. The TPI model has rapidly growing experience model and it affords a structural framework of allusion for examining the strong and weak points of the recent testing process and originate particular and practical enhancement activities for this testing process [3].

The main features of the TPI model are depicted in Figure 2.

## IV. SOFTWARE PRACTITIONER INVOLVEMENT

Figure 3 shows the number of surveys published by only the academic researchers and software practitioners or the collaborations between the researchers and practitioners. The graph shows that both academic researchers and software practitioners of using the TMA/TPI model ias gradually growing by the year 1990 [8]. The selection of source was partial in 2015 that includes only 5 sources because they were opted only in June 2015.

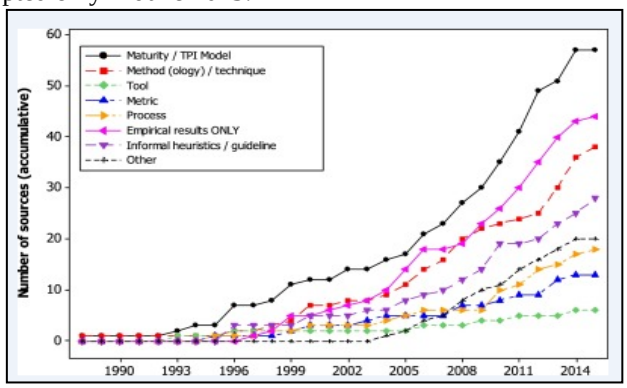


Fig. 3: Growth of TMA/TPI Model

In this survey, the most familiar techniques were used called as Test Maturity Model Integration (TMMi) and it's before version called as the Testing Maturity Model (TMM) and TPI and its new method, TPI Next. Many of the sources used TMMi and TMM for validation or primary models whereas few of the sources have used TPI and TPI Next. Few others used other models called Test-SPICE and the Test Management Approach (TMap) [15].

The group of nine test maturity models is described in the Table no. 2, where the three models are generic, next three models are for particular software development type and the next three are for particular purposes.

| Category | Model* | Staged or continuous? |
|---|---|---|
| Generic | Test Maturity Model Integration (TMMi) | Staged:<br>Level 1. Initial<br>Level 2. Managed<br>Level 3. Defined<br>Level 4. Measured<br>Level 5. Optimization |
| | Test process improvement (TPI) | Continuous:<br>1. Test strategy<br>2. Lifecycle model<br>3. Moment of involvement<br>4. Estimating and planning<br>…<br>18. Test process management<br>19. Evaluation<br>20. Low-level testing |
| | TestSPICE | Continuous. Comprises a set of KPAs based on ISO/IEC 15504, the Software Process Improvement and Capability Determination (SPICE) standard. |
| | | Staged: |

 

| | | |
|---|---|---|
| Targeted for specific development types or domains | Agile Quality Assurance Model (AQAM) | Level 1. Initial<br>Level 2. Performed<br>Level 3. Managed<br>Level 4. Optimized |
| | Agile Testing Maturity Model (ATMM) | Staged:<br>Level 0. Waterfall<br>Level 1. Forming<br>Level 2. Agile Bonding<br>Level 3. Performing<br>Level 4. Scaling |
| | TPI for Embedded Software and Industrial Characteristics (TPI-EI) | Continuous. An adaptation of TPI for embedded software |
| Targeted for specific test activities | Unit Test Maturity Model (UTMM) | Staged:<br>Level 0. Ignorance<br>Level 1. Few Simple Tests<br>Level 2. Mocks and Stubs<br>Level 3. Design for Testability<br>Level 4. Test-Driven Development<br>Level 5. Code Coverage<br>Level 6. Unit Tests in the Build<br>Level 7. Code Coverage Feedback Loop<br>Level 8. Automated Builds and Tasks |
| | Automated Software Testing Maturity Model (ASTMM) | Staged:<br>Level 1. Accidental Automation<br>Level 2. Beginning Automation<br>Level 3. Intentional Automation<br>Level 4. Advanced Automation |
| | Personal Test Maturity Matrix (PTMM) | Continuous. Comprises a set of KPAs such as test execution, automated test support, and reviewing |

Table 4: Test Maturity Models

The test engineers skill or knowledge maturity and capability enhancement can be measured by developing the methodology like TPI's for Embedded Software and Industrial Characteristics (TPI-EI), the Personal Test Maturity Matrix (PTMM) or the Unit Test Maturity Model (UTMM). After surveying the model description the various aspects are analyzed to determine the overlying models [13].

There are 13 testing maturity models which is equivalent to Capability Maturity Model Integration (CMMI) program and it falls under the two categories called as staged or continuous. The models like the Agile Quality Assurance Model (AQAM),TMMi, and the Automated Software Testing Maturity Model (ASTMM), the level of maturity in testing is allocated a grade on the origin of a set of particular goals and methods in staged models, whereas in continuous models, a group of KPAs are evaluated by the defined criteria's by set of methods TPI, TestSPICE, and PTMM.

Figure 4 depicts a sequential development chart of TMA and TPI techniques and their relationships and it tells how the present models are relevant to the preceding models [12]. The below given figure is based on the introduction model equipped for UML. The new TMA and TPI models can be extended and it is introduced in 1985. Various recent models are similar to older methods, i.e. the Metrics Based Testing Maturity Model (MB-TMM), is develped in 2001 and it is relevant to TMM.

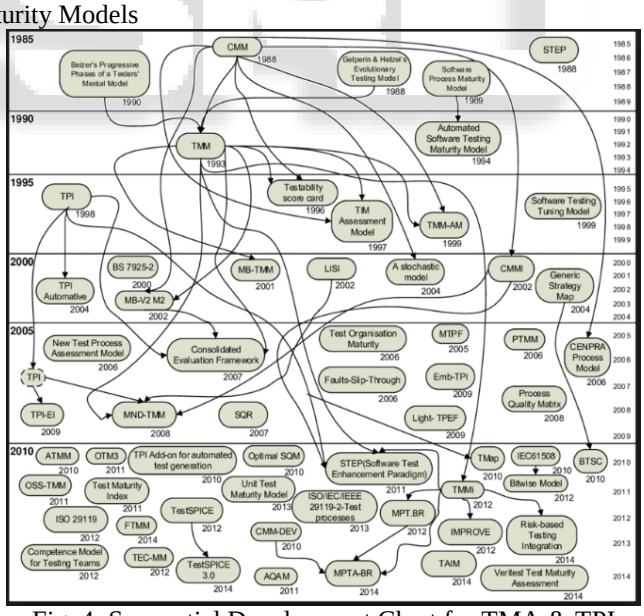


Fig. 4: Sequential Development Chart for TMA & TPI

## V. DRIVERS

After extracting the data, driver login happens in diverse forms and methods and hence the driver codes are organized and categorized into various categories.

a) Process and operational drivers includes
– The actions in testing and the performance are lacking
– Test efficacy is low

 

- Standards and the testing procedures are not up to the user expectation level
- The already available testing procedures are not satisfying the internal stakeholder
- There is no exit testing method
- Testing productivity needs to be improved
- The foundation for testing can be examined to develop a structure for clear testing
- Testing profile must be increased

b) Software quality drivers includes

- The faults are more because of poor quality in testing
- There should be a direct association among the well-developed final product quality and testing process quality
- The resource planning for testing is lacking and it have more influence on the quality of the software

The last thing is, there are many sources are cited for various other drivers. Most of the sources have said that the important step in initiating TMA and TPI was to attain customer's commitment [16]. To attain commitment, both qualitative and quantitative cost–benefit analysis of TMA and TPI functions are more important.

## VI. Conclusion

The execution of TMA and TPI is based on the actual benefits and the categories are classified based on the benefits. The first benefit is the operational benefit that includes:

- Development time is very short
- Development costs is very low
- Testing costs is low due to accurate planning
- Project release dates are satisfying the commitment level
- Administration failure is rare
- Cycle time of testing is minimum
- Risks can be identified easily and the management is very easy
- Training period is high enough

The second thing is Technical benefits that includes

- Defects are low in producing software quality
- High damage causing defects are very low
- Test automation is enhanced
- Test-case designs are improved by making use of new techniques

The third thing is Business benefits that includes

- Profit rate is more
- Customer satisfaction is increased
- Defect costs are reduced

## References

[1] V. Garousi et al., "A Survey of Software Engineering Practices in Turkey," J. Systems and Software, vol. 108, 2015, pp. 148–177.
[2] V. Garousi and J. Zhi, "A Survey of Software Testing Practices in Canada," J. Systems and Software, vol. 86, no. 5, 2013, pp. 1354–1376.
[3] M. Grindal, J. Offutt, and J. Mellin, "On the Testing Maturity of Software Producing Organizations," Proc. Testing: Academia & Industry Conf.—Practice and Research Techniques, 2006, pp. 171–180.
[4] G. Bath and E.V. Veenendaal, Improving the Test Process: Implementing Improvement and Change—x` a Study Guide for the ISTQB Expert Level Module, Rocky Nook Computing, 2013.
[5] R.T. Ogawa and B. Malen, "Towards Rigor in Reviews of Multivocal Literatures: Applying the Exploratory Case Study Method," Rev. Educational Research, vol. 61, no. 3, 1991, pp. 265–286.
[6] V. Garousi, M. Felderer, and M.V. Mäntylä, "The Need for Multivocal Literature Reviews in Software Engineering: Complementing Systematic Literature Reviews with Gray Literature," Proc. 2016 Int'l Conf. Evaluation and Assessment in Software Eng. (EASE 16), 2016, pp. 171–176.
[7] E. Tom, A. Aurum, and R. Vidgen, "An Exploration of Technical Debt," J. Systems and Software, vol. 86, no. 6, 2013, pp. 1498–1516.
[8] T. Dybå and T. Dingsøyr, "What Do We Know about Agile Software Development?," IEEE Software, vol. 26, no. 5, 2009, pp. 6–9.
[9] T. Hall et al., "What Do We Know about Developer Motivation?," IEEE Software, vol. 25, no. 4, 2008, pp. 92–94. 10. W. Afzal et al., "Software Test Process Improvement Approaches: A Systematic Literature Review and an Industrial Case Study," J. Systems and Software, Jan. 2016, pp. 1–33.
[10] C. Garcia, A. Dávila, and M. Pessoa, "Test Process Models: Systematic Literature Review," Software Process Improvement and Capability Determination, A. Mitasiunas et al., eds., Springer, 2014, pp. 84–93.
[11] T. Koomen and M. Pol, Test Process Improvement: A Practical Step-by-Step Guide to Structured Testing, Addison-Wesley, 1999.
[12] S.L. Cepeda, "CMMI—Staged or Continuous?," 2005; www.sei.cmu .edu/library/assets/cepeda-cmmi.pdf.
[13] G. Zockoll, A. Scheithauer, and M.D. Dekker, "History of Object-Oriented Methods and Notation," 2012; en.wikipedia.org/wiki/Unified _Modeling_Language#/media /File:OO_Modeling_languages _history.jpg.
[14] K. Rungi and R. Matulevic˘ius, "Empirical Analysis of the Test Maturity Model Integration (TMMi)," Information and Software Technologies, T. Skersys, R. Butleris, and R. Butkiene, eds., Springer, 2013, pp. 376–391.
[15] M. Felderer and R. Ramler, "Integrating Risk-Based Testing in Industrial Test Processes," Software Quality J., vol. 22, no. 3, 2014, pp. 543–575.

 