Software Test

A Comparative Analysis of Test Process Improvement Models TMMi, STEP, TPI NEXT & CTP

Author: Aimée Flora Jensen - 59259

Master's **T**hesis

Supervisor: Nina Boulus-Rødje

Roskilde **U**niversity - **D**epartment of **I**nformatics

Characters (including spaces): approx. 94.545

Deadline: **J**une **1**st, 2018

Abstract (DA)

I takt med, at software i stigende omfang bruges til flere applikationer af både kritisk og kompleks karaktér, vil der stadig være behov for anvendelige, effektive og robuste frameworks til optimering af software kvalitet. På nuværende tidspunkt, eksisterer der en række modeller til virksomheder, der måtte have interesse i at forbedre effektiviteten af deres softwaretest processer. Skønt dette, er tilgængeligheden af empirisk-funderet vejledning omhandlende, hvilken model der med fordel kan vælges i en given sitution mangelfuld. Derfor, vil denne afhandling gennem et litteraturstudie, foretage en grundig undersøgelse og sammenligning af de fire mest anvendte testprocesforbedringsmodeller (eng. TPI); TMMi, STEP, TPI NEXT og CTP. Ved at inddrage en række kilder med ophav i den akademiske verden og softwaretestbranchen, identificeres dét der kendetegner de fire TPI modeller, og udpeger både fordelene og ulemperne ved anvendelse af disse. Dermed søger denne afhandling at lægge et empirisk fundament, der kan bruges til at udvikle et kriteriebaseret beslutningstiltag til at vejlede virksomheder i processen med at udvælge den TPI-model, bedst egnet til at efterkomme deres behov og mål.
Resultaterne tyder på, at de fire TPI modeller kan differentieres afhængigt, på grundlag af deres relative fleksibilitet og modularitet, hvilket giver dem mulighed for at blive kategoriseret i to grupper: TMMi og TPI NEXT er mere omfattende, men også mere rigide, procesmodeller struktureret ved modenhedsetaper, mens STEP og CTP er mere fleksible og mindre præskriptive indholdsbaserede referencemodeller, der kan tilpasses en bredere vifte af organisatoriske sammenhænge.
Under litteraturstudiet, bliver en række alvorlige mangler i den akademiske litteratur identifceret og evalueret. Der foreligger et behov for yderligere akademisk forskning i dette stadig vigtige emne.

Emneord: Test Process Improvement; Testprocesforbedring; TMMi; STEP, TPI NEXT; CTP; softwaretest.

Abstract (ENG)

As software is used in increasingly broad, critical, and complex applications, the need for usable, efficient, and robust frameworks to enhance software reliability will continue to grow. Currently, a number of models are available for organizations interested in improving the efficiency and effectiveness of their software testing processes. **However, the availability of empirically grounded guidance regarding which model to choose for a given situation is lacking.** This thesis, therefore, uses a literature review methodology in order to conduct a detailed examination and comparison of the four most commonly used test process improvement (TPI) frameworks (STEP, CTP, TMMi, and TPI NEXT). Drawing upon a range of scholarly and industry sources, it identifies the defining features characterizing each of these frameworks and considers their relative advantages and disadvantages in common applications. In doing so, it seeks to lay an empirical foundation that might be used to develop a criteria-based decision-making framework to guide organizations in the selection of the TPI model best suited to their needs and objectives. The results suggest that these four models can be most readily differentiated based on their relative flexibility and modularity, which allows them to be categorized into two groups: TMMi and TPI NEXT are more comprehensive, but also more rigid, process models structured by maturity stages, while STEP and CTP are more flexible and less prescriptive content-reference models that can be adapted to a wider range of organizational contexts. In the course of the review, a number of serious gaps in the scholarly (and particularly empirical) literature are identified and evaluated. The need for additional scholarly research into this increasingly important topic is strongly underscored.

Keywords: Test process improvement; TMMi; STEP; CTP; TPI NEXT; software testing

Table of Content

1. Introduction

According the International Software Testing Qualifications Board (ISTQB 2016), software testing can be defined as:

> *"The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation, and evaluation of software products and related work products to determine that they satisfy specified requirements, or to demonstrate that they are fit for purpose and to detect defects" (ISTQB 2016 p. 77)*.

Perhaps unsurprisingly, software testing has long been regarded as an integral part of the software development process that supports software quality and functionality. As programs and applications continue to grow in scale, scope, and complexity—and as software is being deployed in a wide range of operational contexts to perform increasingly critical tasks—software testing is becoming even more vital (Burnstein, Homyen, Grom, & Carlson 1998; Pressman 2005; de Souza, de Almeida Falbo, & Vijaykumar 2015; Ammann & Offutt 2016). In fact, software development increasingly provides a basic conceptual and logistical foundation for development paradigms (e.g. agile development and test-driven development, or Test Driven Development) (Ammann & Offutt 2016; Butt et al. 2017; Madeyski & Kawalerowicz 2018).

It is not terribly surprising, therefore, that a significant share of project budgets in the software development space are dedicated to software testing practices (Myers 2004).

Although these figures vary significantly depending on a range of factors (e.g. project type, complexity, functional area, etc.), it is not uncommon for software testing activities to account for half of total project budgets, and upwards of a quarter of information technology budgets overall (Harrold 2000; Ng et al. 2004; Ammann & Offutt 2016; World Quality Report 2015, 2018). From a business perspective, these high costs generally deliver favorable returns on investment in terms of customer satisfaction and cost savings in maintenance, bug fixes, safety, reliability, and so forth (Slaughter, Harter, & Krishnan 1998; Bertolino 2007; Ahner, Wisnowski, & Simpson 2017). Nonetheless, it is not unreasonable to expect testing costs to continue to rise hand-in-hand with the growing complexity and criticality of software itself. Thus, there is a clear and well-documented need for more efficient and effective approaches to testing. Furthermore, because the process for choosing "the most suitable model for a specific organizational context" remains challenging and often opaque, there is a case to be made that efficiency and effectiveness can be boosted by simply developing guidelines for organizations seeking to select between the available models.

According to Bath and van Veenendaal (2013), broadly speaking the four most commonly-used Test Process Improvement (TPI) models are; Testing Maturity Model Integration (TMMi), Systematic Testing Evaluation Process (STEP), Test Process Improvement (TPI NEXT), and Critical Testing Processes (CTP) (p. 118).

With this in mind, rather than undertaking a general survey of the full range of test process frameworks that have been proposed, in the interest of clarity (and in keeping

with a pragmatic research philosophy - explained later on), this project is somewhat more narrowly delimited. Specifically, it is scoped to facilitate an in-depth examination of these four most common TPI models by way of the following three research questions:

1.2 Research Questions

1. What are the key features characterizing the TPI models TMMi, STEP, TPI NEXT and CTP?
2. What are the strengths, weaknesses, and common applications of each of these models?
3. What evidence-based criteria might be used to identify the model that is best-suited to a specific project (if any)?

Through a critical exploration of these research questions, this thesis aims to develop (or at least lay the groundwork for) a decision-making framework that can guide developers and managers in selecting a TPI approach for a given project type or domain. By carrying out critical comparative analysis of these widely-used and relatively well-studied TPI models, it seeks to synthesize the industry and academic literatures on the topic in order to derive criteria that might be used to choose which model should be used in a given project context.

In this way, it is hoped that this thesis will not only provide a practical guide to help organizations improve their testing processes, but also identify possible functional gaps between these four major models that future TPI research might seek to address.

2. Methods

2.1 Research Philosophy

According to Saunders and Tosey (2013), research philosophy is an important part of the research design process, and significantly informs a number of methodological choices. The foundational philosophy guiding this thesis is pragmatism: the research project described here defines its value, first and foremost, in terms of its "*practical consequences*" (p. 58). In this context, this means its ability to both inform testing practice and to suggest directions for future TPI research by academics. To this end, research design decisions should be made with an eye toward enabling "*credible, reliable, and relevant data to be collected that support subsequent action.*" (p. 58).

At the same time, despite its technical subject matter, this project is also shaped (if to a lesser degree) by interpretivism. This philosophy is illustrated by the way in which the project seeks to not only analyze selected TPI models in terms of their internal logic and structure, but also to attend to them as sets of prescribed behaviors and actions carried out in specific organizational, and therefore social, contexts (pp. 58-59; Dawson et al. 2003; Doolin & McLeod 2005; Tedre 2007).

2.2 Data Collection & Analysis

As suggested above, in light of the present project's pragmatic orientation, its focus on relatively abstract TPI models, and the apparent lack of a widely-accepted empirical foundation for the development of decision-making frameworks capable of guiding TPI

model selection, it was determined that a systematic literature review would offer the best methodological approach for addressing the research questions guiding this thesis.

With respect to sampling, it was originally my intent to rely primarily on recent, peer-reviewed scholarly sources. In the course of conducting background research for the project, however, a preliminary review of the academic and industry literatures revealed a surprising scarcity of sources treating CTP, TMMi, TPI NEXT, and STEP. Based on this finding, it was determined that a sampling approached based exclusively or even primarily on peer-reviewed scholarly literature would not produce a sufficiently large or diverse sample. Moreover, such an approach would be unlikely to effectively represent accepted industry best practices.

Consequently, the sampling procedure was modified in order to enable the incorporation of not only scholarly sources, but also industry data, including textbooks, testing certification curricula, industry best practice standards, and in some cases, corporate publications by major testing organizations—particularly those with a direct connection to TPI models studied here (e.g. Sogeti, which produced and oversees the TPI NEXT model).

In order to qualify for inclusion in the data-set, therefore, a potential source would have to satisfy the following broad criteria. First and most importantly, it had to represent a reputable source from one of the publication types described above, and had to have direct and explicit relevance test process improvement as a whole, or alternatively to one or more of the models that are the focus of this thesis.

Curricula for private tutoring organizations based on industry-accepted practice standards (e.g. ISTQB) were accepted to elaborate only certain concepts that could be verified. With respect to timeliness, candidate sources had to be published within the last four decades (since the rise of modern formal software testing practices). Finally, only English-language articles were considered.

One exception to these criteria involves publications by seminal authors in the field. Many of the progenitors of today's foremost TPI frameworks are still actively engaged in the industry. In some cases, they publish their writing in informal contexts, such as interviews, blogs, periodicals, or other industry publications. In the case of a piece authored by a widely-cited TPI leader like Rex Black or Erik van Veenendaal, informal publishers were not viewed as disqualifying.

Data collection began with a series of searches of common scholarly and non-scholarly databases e.g REX, ResarchGate, ACM Digital Library, Semantic Scholar, ScienceDirect, Google Schoolar, Google. Returned texts were screened first by title and abstract against the criteria for inclusion given above.

The remaining texts were screened more thoroughly by delving into the report bodies. After this two-tiered screening process, direct and reverse citation searches were performed (identifying the sources of included articles, as well as later articles citing included articles, respectively) in order to further expand the corpus for analysis. This produced the final sample. The data were analyzed through a process identifying the framework being discussed, methodological variables, and the presence or absence of empirical data and/or comparative orientation.

3. Literature Review

3.1 Background

The theoretical literature surrounding software development and testing theory is relatively rich and robust, comprising detailed evaluations of multiple methodologies (e.g. static and dynamic testing, box-based approaches) carried out at a range of levels (e.g. unit, integration, interfacing), leading to the development of a diverse array of typologies and techniques (Lewis 2000; Singh, Singh, & Singh 2010; Orso & Rothermel 2014). The literature focused on test process improvement specifically, however, is arguably both younger and more limited.

Important dimensions of industry and scholarly TPI discourses can be traced back to the emergence of more general process improvement frameworks. For the purposes of the present discussion, the Capability Maturity Model (CMM) and the Capability Maturity Model Integration (CMMi) are particularly relevant, as discussed below. Briefly, the CMM has its roots in Nola's (1973) stages-of-growth model for information technology organizations, and thus offers a staged maturity model comprised of a process model continuum, key process areas and their goals, common features, and recommended key practices (Paulk 2009). The CMM was succeeded by the CMMi, which offered a more integrated framework featuring (among other things) greater compatibility with agile methodologies, as well as greater emphasis on training and appraisal efforts (ibid.; Kneuper 2008; Samalikova et al. 2014).

3.2 Context

Over the course of the last three decades, the number of proposed process improvement frameworks has proliferated. Due to the complexity of the testing field and its diversity of applications, however, the result has often been a sprawling and heterogeneous body of process improvement literature. The tendency of process improvement research to disaggregate into a "*framework quagmire*" has been a documented issue for decades (Ahern, Clouse, & Turner 2008; Fig. 1).

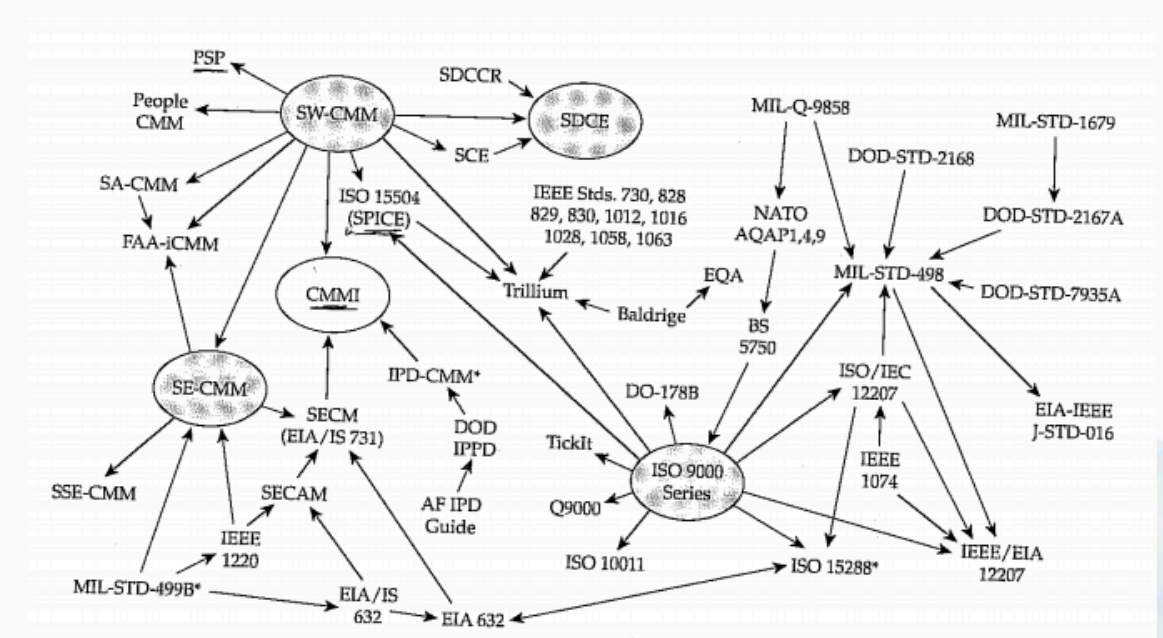

Figure 1: Visual representation of process improvement framework quagmire (Ahern, Clouse, & Turner 2001).

It is worth noting that since the above image was published, the number of proposed process improvement frameworks has grown substantially (Garcia, Davila, & Pessoa 2014).

This state of affairs has a number of implications for the present project, for organizations interested in implementing TPI, and for process improvement research in general—particularly when coupled with a general lack of empirical guidance (discussed in more detail in the following sections). On the one hand, it illustrates the need for evidence-based guidance when choosing a TPI model, particularly given the sheer diversity of the field. On the other hand, the figure above also illustrates that these models are frequently not proposed in isolation, but are developed in a way that is informed by previous models. Thus, even if one seeks to carry out a more focused and in-depth investigation (rather than a general survey of the field) by focusing on the most common models, as this thesis seeks to do, a full explanation of even a small number of models selected for analysis might require discussing a set of additional models that influenced their development.

3.3 Testing Maturity Model Integration (TMMI)

The TMMi framework originated as a community-driven initiative to develop improvement models for testing processes: noting that testing consistently accounts for a significant portion of project costs, the testing community set out to create a "detailed" TPI model (TMMi Foundation 2012, p. 6). The TMMi is almost unique among TPI models in that both its framework and the accompanying assessment model requirements for formal and informal assessments are "*publicly available and free of charge*", while other approaches often require hiring a certified inspector or

consultant using proprietary methodologies (Rungi & Matulevicius 2013, p. 379; Rasking 2011).

The TMMi is explicitly pragmatic and action-oriented: in addition to providing a systematic approach to evaluating the maturity of the test processes currently in place in a given organizational context, it offers guidelines that lay out gradual, sequential, and controllable improvement steps. These steps will be discussed in significantly greater detail below, but first some review is necessary. In order to understand the TMMi, it is important to attend closely to the contexts it was designed to respond to. This context includes the dearth of other models fit to function, as well as conceptual, anatomical, and practical elements drawn from or inspired by other process improvement models widely used in the software industry.

In order to understand the etiology of the TMMi model, then it might be useful to start with a short consideration of its predecessor, the Testing Maturity Model (TMM) (van Veenendaal, Hendriks, van de Laar, & Bouwers 2008).

3.3.1 TMM

Briefly, the TMM was developed and proposed by a team of researchers at the Illinois Institute of Technology led by Bob Carlson and Ilene Burnstein (Burnstein, Suwanassart, & Carlson 1996; Burnstein, Homyen, Grom, & Carlson 1998). Although the TMM is occasionally portrayed as a competitor to CMM, the researchers were actually interested in developing a complementary TPI framework, and this focus is apparent in the structure of the TMM.

The basic anatomy of the TMM involves a kind of classification system, which establishes six possible levels describing the maturity of testing processes:

1. Initial
2. Phased
3. Systematic
4. Integrated
5. Managed
6. Optimal

At the first level (Initial), quality standards are effectively absent, with organizations relying on ad hoc testing methods and protocols. Generally, this is accompanied by an unsystematic approach to the engineering process and a tendency to leap directly into the coding process upon receiving requirements and specifications, without a dedicated design process utilizing established, systematic methodologies (Swinkels 2000, p. 13).

Farooq and Dumke (2008) argue that the TMM represents "*the most comprehensive test process assessment and improvement model*", although the researchers appear to consider TMMi a subset of TMM in making this assessment (pp. 122-23).

At the opposite end of the spectrum from the initial maturity level, of course, is the optimal organization. At this sixth and final level, the organization utilizes consistent and repeatable software design methodologies which are well-understood and widely followed. This understanding, in turn, enables focused and reflexive discussions about specific aspects of these design and testing processes aimed at addressing failures and

deploying strategies to improve efficiency. As a side note, it is worth pointing out that for practical reasons, the TMM does not use in-depth research to facilitate its assessments. Instead, the TMM's assessment method is based on a scorecard questionnaire with 20 testability factors (Rungi & Matulevicius 2014).

Just as the TMM was designed as a complement to the CMM, the TMMi was designed as a complement to the CMMi (van Veenendaal 2012).

Having briefly characterized the models the TMMi is based on and designed to complement (the TMM and the CMMi, respectively), therefore, it is time to examine the TMMi itself.

3.3.2 TMMi

Like the TMM, the TMMi is a hierarchical model which seeks to tailor TPI guidance to a diagnosis of the existing system, relative to a number of discrete tiers (Fig. 2).

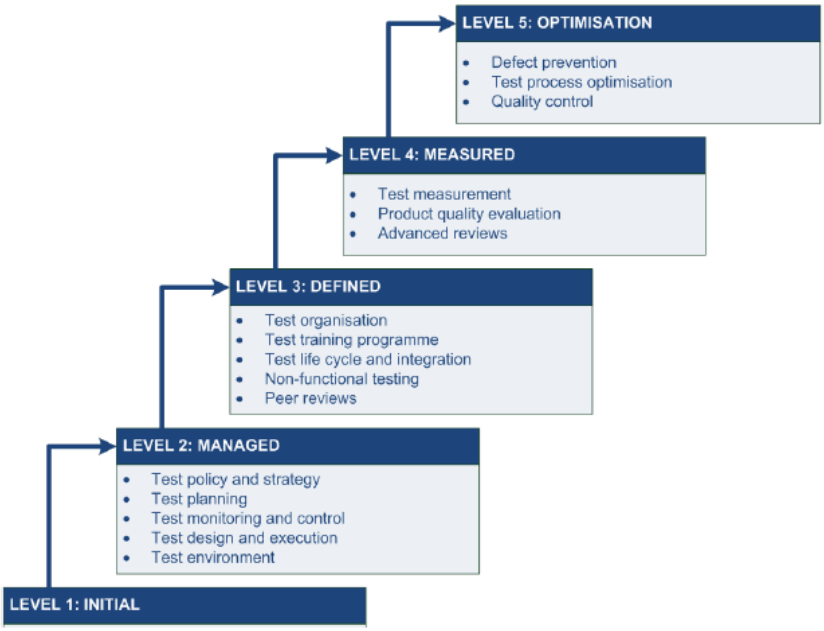

Figure 2: TMMi maturity levels (TMMi Foundation 2018)

It is worth noting that in the terminology both the TMM and TMMi, the bottommost stage of the model receives the designation Initial. This is because the model's hierarchical nature assumes a sequential progression through each stage: wherever an organization may be located on the ladder represented in the figure above, TMMi assumes that it began at a maturity level of 1 (ibid.).

Thus, each level has a distinct scope and focusing in clearly delimited process areas and encompassing both general and specific goals (Bris, Frantis, & Kolkova 2015).

It is only after attaining a given level that an organization is deemed ready to orient its activities towards achieving the next one, with the attainment threshold being set at achievement of 85% or higher of goal completion for the level in question (p. 806).

Although the levels in TMMi bear obvious similarities to those used in the TMM, CMM, and CMMi, the focus of the TMMi is slightly different, and thus in the interest of providing a comprehensive overview it is worth taking a moment to elaborate on each level here.

As in the TMM, at Level 1 (Initial), there is little differentiation between testing and more ad hoc debugging—or even, for that matter, between testing and other development processes. The approach is unsystematic, lacking both formal structure and dedicated documentation; tests are typically performed only when problems are encountered, in many cases when coding is in its final stages. Unsurprisingly, this can significantly compromise efficiency and significantly increases the risk of encountering costly and even critical problems late in development, where correcting them may be difficult or even impossible (TMMi Foundation 2012, p.10)

The fundamental criterion for qualifying for Level 2 (Managed) is the presence of a clear delineation between the processes of testing on the one hand and debugging on the other. As suggested in the figure above, this means that basic test policy is in place and formalized, including an explicit strategy. The use of test plans and methods must exist not only on paper, but must be carried into practice in the context of specific projects (ibid).

At Level 3 (Defined), testing moves beyond the project scale and begins to impact organizational structure; testing typically maintains an emphasis on dynamic techniques. This is to say that standards and protocols have now become standardized across projects "*throughout the organizations or organizational unit*"—while continuing to fulfill the criteria laid out in the previous level TMMi Foundation 2012, p.11):

> "*Level 2 is still being done, and teams are now organized, training programs exist, test is integrated into the development life cycle and integrated into all projects from early in development. Non-functional testing is planned and executed in all projects and reviews are used in each project as well*". (TMMi 2012, p. 10).

As the name suggests, at Level 4 (Measured) the measurement of testing processes and their outcomes are incorporated into standard practice: the focus shifts from simply putting resources in place to test software, to continually monitoring how efficiently and effectively that testing is carried out. Still, this stage is not only associated with testing efficiency, but also with desirable development outcomes (e.g. fewer product defects). At the same time, static techniques are increasingly incorporated into testing

practice, and "*advanced reviews*" become a mainstay of the development process, including from early project stages (TMMi Foundation 2012, p. 11).

Finally, when Level 5 (Optimization) is achieved, the TMMi does not offer further recommendations for altering the testing framework in place for an organization. Instead of specific prescriptions for new practices or techniques, the model simply suggests that organizations continue to monitor and assess existing processes with an eye toward ensuring "*continuing improvement toward defect prevention and optimized quality*" (ibid, p. 12).

Once again, perhaps the most obvious source of appeal of this evolutionary, staged maturity model for testing specifically is its ability to facilitate not only the assessment of the current maturity level, but also its action-oriented structure, which provides "*a clear improvement path*" for achieving the next level in the ladder (Rungi & Matulevicious 2013, pp. 377-78). However, TMMi's origin as a complement to CMMi also makes it appealing for reasons that lie beyond the testing domain specifically, and which structure its relationship to other areas of the development process. Specifically, the assessment procedure used for TMMi (and the results it produces) can be leveraged to aid in subsequent CMMi evaluations (p. 378).

Given this, it is not particularly surprising that TMMi has earned a place as one of the most dominant and commonly-used TPI models. For this and other reasons (including its community-driven etiology), TMMi also has been widely discussed and subject to multiple evaluations using a range of approaches.

One particularly useful examination of the TMMi is the empirical case-study based performance evaluation by Rungi and Matulevicious (2013). Briefly, the study involved employing the TMMi Reference Model and the TMMi Assessment Method Application Requirements (TAMAR), including its assessment protocols, in order to evaluate testing processes at a casino in Estonia and recommend improvements.

Data collection included surveys and staff interviews. While the researchers found TMMi to be a "*valuable source of best practices when planning improvements*", and although the model appears to have offered thorough and systematic TPI, a number of difficulties and shortcoming both internal and external to the model were identified. For one, while the model is useful in terms of guiding maturity progressions in testing processes, it is not a business model, and it cannot tell organizations what they aim to accomplish by implementing TPI. Thus, the researchers found that without detailed requirements from candidate organizations, including its expectations regarding the TPI tool and the specific benefits it would produce, managerial buy-in and continued cooperation can be difficult to obtain. Internal shortcomings included the identification of gaps in the assessment method requirements that could potentially undermine the model's "c*orrectness and reliability*": surprisingly, TAMAR failed to provide an operationalize assessment method for "achievement level calculation", which could create disconnects between the results of in-house informal assessments and those of external formal assessments (p. 389). This, in turn, could compromise the achievement of goals and, consequently, slow the progression up the maturity ladder.

In a similar vein, the researchers noted that the TMMi's comprehensive design and use of discrete maturity stages could discourage the introduction of more isolated, modular testing improvements:

> *"All of [TMMi's] processes are strictly divided between maturity levels. A continuous model would allow the organization to pick the process areas that are believed to bring the greatest benefit and arrange improvement activities based on capability levels rather than maturity levels"* (p. 389).

Such a model would arguably be better suited, for instance, to use in agile development environments and similar contexts.

Overall, the researchers concluded by stressing once more the importance of choosing "*the most appropriate model*" for a given organizational and task context, emphasizing that this determination may depend on a variety of variables ranging from "*work methodologies and needs*" to "*objectives and financial capabilities*" (pp. 389-90).

3.4 Systematic Test Evaluation Process (STEP)

According to Craig and Jaskel (2002), the Systematic Test and Evaluation Process (STEP) model was originally introduced in the mid-1980s not as a formal, standalone, or proprietary approach, but rather as simply "*part of the course material*" for an industry seminar series called Systematic Software Testing (p. 10). It proved to be unexpectedly popular, however, and generated sustained attention and undergoing an extensive series of revisions and field-tests through "*consulting engagements and the*

shared individuals of many individuals and organizations" collaborating to improve the scope of the framework:

> *"While retaining compatibility with [IEEE] standards,[1] this methodology has grown in scope and now stands as one of the leading models for effective software testing [... covering] the broad activity of software evaluation [...] the sub-discipline of software engineering concerned with determining whether software products do what they are supposed to do"* (pp. 10-11).

The contemporary practice divided evaluation into the sub-processes of analysis, review, and testing, the latter of which was widely viewed as the most complex—at least in part because it was treated as an activity that necessary followed the completion of the development phase— and thus STEP's primary area of emphasis with a focus on prevention (discussed in more detail below).

Secondary foci included "*defect detection and demonstration of capability*", although STEP was also notable at the time for the importance it assigned to more general processes like planning and project coordination (pp. 10-11). Thus, STEP played an important (though by no means conclusive) role in displacing the dated sequential build-then-test paradigm with the more integrated modern lifecycle perspective in which builds and rebuilds are pursued concurrently and in a manner that overlaps with testing (Fig. 3).

[1] The STEP model used as its basis for conceptual orientation and documentation a series of IEEE standards (e.g. "Std. 829-1983 Standard for Software Test Documentation") and is regularly revised in order to adopt changing or updated underlying IEEE standards (ibid.).

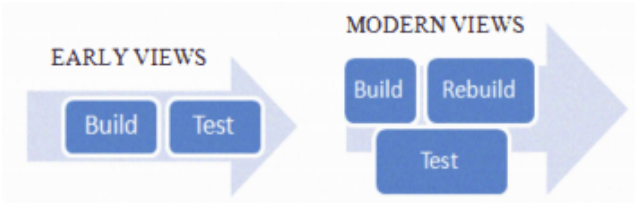

Figure 3: Divergent development paradigms (Sulaiman, Kassim, & Saaidin 2010, p. 219)

Although the early view in the figure above certainly persists today in many settings, it is important to keep in mind that when STEP was developed this approach represented the dominant, and in many cases the only, paradigm: typically, the beginning of the testing process was marked by the execution of actual tests, with the importance of planning, analysis, and design "*unrecognized*" at best (p. 11).

Unlike TMMi, the Systematic Test and Evaluation Process (STEP) framework is not a maturity model, and does not prescribe improvements in the form of a rigid sequence by which one set of improvements must be implemented before addressing the next. Instead, STEP can perhaps best be described as type of content reference model with both qualitative and qualitative components, rather than as a more traditional process reference model: TPI projects can be implemented "*in any order to priority of process areas*" (Ahmed 2016, p. 242; Fig. 4).

Interestingly, despite the fact that industry publications, including ISTQB testing, commonly refer to STEP as one of the predominant TPI frameworks currently in use, comparatively few independent (and particularly scholarly) analyses of the framework were identified in the course of this review (Bath & Van Veenendaal 2013).

This is partially due to the much-discussed methodological difficulties involved in seeking to systematically and objectively assess the impact of implementing any given TPI framework, from timescale selection to baseline establishment and benchmarking outcome measures. Nonetheless, this dearth of research and the reliance on descriptive and prescriptive accounts of the STEP framework represents a limitation for the current study that somewhat complicates the comparative analysis presented below. With this qualification in mind, then, let us turn our attention to considering the key features of the STEP model itself.

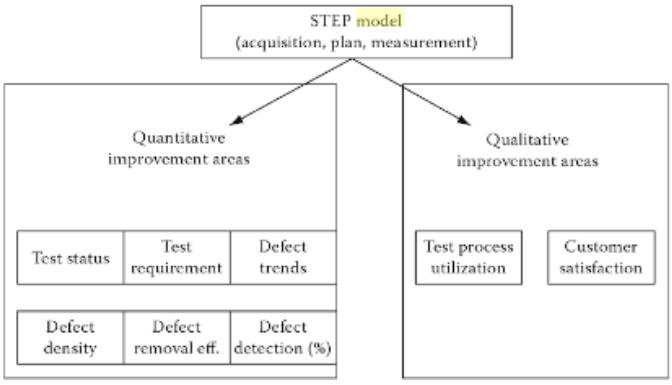

Figure 4: The STEP model contains both qualitative and quantitative components (Source: Ahmed 2016, p. 242)

Specifically, the STEP model recommends a number of characteristics that a given testing process should exhibit, such as a "*requirements-based testing strategy*", for instance; Ahmed offers a succinct review of the ideal testing process as described by the STEP model:

"Testing should start at the beginning of the software development life cycle. Test cases are used as requirements and usage models. Testware design is the basis for software design. Defects are

detected at their origin and should be removed at that point. Defects are systematically analyzed; testers and developers work together [...]" (p. 243).

Thus, in the prevention-oriented STEP model (in which writing tests precedes any actual coding) there is no part of the development lifecycle in which testing does not play an integral role, ensuring the testability of code and its ability to satisfy predetermined specifications (ISTQB 2012, pp. 59-61). In many cases, then, the STEP model is viewed as being in close alignment with, and well positioned to complement, Test-Driven Developmen approaches and agile methodologies like Scrum and EXTREME Programming: here, the software development process is rooted in requirements that take the form of "*test cases*", which source code is written to validate from the outset (Ahmed 2013, pp. 242-43).

As with TPI NEXT (below), STEP's areas of emphasis can be loosely grouped together into functional clusters, including areas like task-relevant actions, products, role definition, and so forth. These functional clusters form the basis of STEP's conceptual framework, rather than functioning as direct prescriptions designed to be translated directly into the anatomy of a specific organization or product team.

For instance, for descriptive clarity, STEP distinguishes between the functional roles of (1) manager, (2) analyst, (3) technician, and (4) reviewer, but it explicitly does not interpret these role categories as referring to a specific team composed of four employees.[2]

[2] Note: In the context of STEP, each of these roles is used to designate a collection of core activities rather than the identity of the entity that performs them. For each of these four roles, these activities might include: (1) facilitation and coordination of planning, as well as internal and external communication; (2) identification and management of needed inventory, design of evaluation protocols, collation and reporting of data needed to inform the planning process; (3) implementation of processes and execution of protocols, spot checks to ensure that this is done correctly, and providing progress updates and reports to other roles; and (4) examining, analyzing, and evaluating the resulting data in order to ensure efficiency, effectiveness, conformity to protocol, and alignment with objectives (ibid.).

Instead, STEP enables a range of interpretations, so that these categories can be applied whether all four roles are performed by the same individual, or by four separate departments (although in practice, of course, it is unlikely that either of these extremes would represent a particularly efficient arrangement). As Huisko and Kyyro (2015) note:

"STEP is not a tool-dependent model, nor does it expect the organization to have certain staffing and test groups [... However,] it does expect testers and developers to work together and to do their respective responsibilities" (p. 41).

Moreover, STEP is designed to be markedly proactive, emphasizing bug prevention and placing a premium on efforts to detect "*defects*" relevant to both the testing process and its subject at an early phase—occasionally requiring more significant upfront testing investments, but generally viewed as a cost-control measure within this model.

Indeed, Kaur and Sing (2014) stress once again that STEP was originally developed during a period in the mid-to-late 1980's in which prevention process models were considered an innovative and much-needed development in the testing industry, and when testing was commonly "decomposed" into a handful of distinct and strictly segregated phases (e.g. "*planning, acquisition, and measurement*") (p. 464).

In much the same way that it carefully compartmentalizes task areas through functional roles, STEP also discriminates between essential work products: namely, documentation, procedures, data, and support software (Craig & Jaskel 2002, p. 12). Similarly, STEP is organized into three major phases (strategy planning, testware

acquisition, and behavior measurement), each of which is associated with certain activity categories. In the first phase, a strategy is selected and specifies, and in the second, "*detailed test objectives*" are specified and test sets are "*designed and implemented*" (pp. 14-15).

One of the comparatively few available academic case studies examining a potential real-world implementation of STEP was carried out by Sulaiman, Kassim, and Saaidin (2010). The researchers set out to systematically analyze the suitability of STEP for the context of a Shared Banking Services (SBS) system in a well-developed banking organization in Malaysia.

In order to do so, they began by evaluating the organization's current testing processes, then critically assessed the STEP framework, and finally systematically compared these descriptive and recommended process accounts in order to identify aspects of the STEP framework that are well-suited to implementation in this context. The notion that testware and software are articulated as analogous and mutually-reinforcing concepts under STEP was seen as novel relative to a more linear status quo, and the authors underscore STEP's emphasis on designing, specifying, and building testware concurrently with the corresponding activities on the software development side.

Overall, the analysis identified a number of real and potential advantages and disadvantages associated with using the framework in this context.

In some cases, these advantages and disadvantages functioned as both sides of the same coin: it was not obvious how to maximize the advantage, or minimize a disadvantage, without eliminating a feature entirely.

For instance, Sulaiman et al. (2010) noted that, on the one hand, STEP's use of IEEE standard documentation practices enables thorough documentation, consistency, standardization, and increases the "*visibility*" of testing activities overall (p. 222). On the other hand, this reliance on external documentation protocols means that external changes (e.g. updates to IEEE standards) may require changing internal documentation protocols per STEP in order to "*adopt*" the external changes (p. 222). Other areas were more unambiguous, however: the authors note that STEP's adaptability to change can be a boon to efficiency in part because it enables the re-use of early-stage work throughout system life under certain circumstances. Overall, the STEP process was found to be both more comprehensive and more flexible than the firm's internal testing protocol, particularly with respect to integrating tests at early stages and encouraging thorough documentation and strategic thinking—as well as requiring the organization to more clearly define its goals, objectives, and specifications.

In this vein, it is worth noting that although STEP does not utilize a rigidly staged maturity model, it does divide the testing task into a number of distinct levels for the purposes of test planning to facilitate activity timing (Fig. 5).

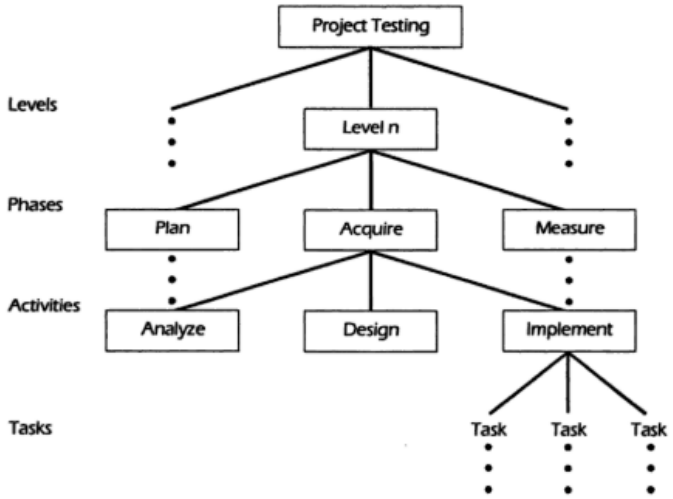

Figure 5: Correspondence between test level and activity timing in the STEP model (Craig & Jaskiel 2002, p. 13).

Rather than serving as a kind of marker of organizational status, a possible interpretation of maturity models) or prescriptive objective, however, levels are used to describe and tag specific testing environments and serve as shorthand for clarifying complexity:

"Simple projects, such as minor enhancements, may consist of just one or two levels of testing (e.g. unit and acceptance). Complex projects, such as new product development, may have more levels (e.g. unit, function, subsystem, system, acceptance, alpha, beta, etc.)" (p. 13).[3]

It is worth emphasizing once again that STEP, as a model, is oriented toward providing a starting point rather than a detailed plan or even a plan structure: each of STEP's components, activities, levels, phases, and so forth, are explicitly intended to be tailored, revised, and/or extended to "*fit*" a given test setting or objective (pp. 12-13).

[3] For instance, a unit test is typically understood in STEP as a designation signifying "the level associated with program testing in a programmer's personal development library", for instance (p. 13).

3.5 Test Process Improvement (TPI NEXT)

Much in the same way that it is difficult to give a complete account of TMMi without a brief introduction to TMM, in order to fully understand TPI NEXT it is necessary to mention its predecessor, known simply as *Test Process Improvement (TPI),* which was developed by Sogeti originally published in 1998 (Banga 2010)[4]. Briefly, Farooq and Dumke (2008) describe *TPI* as an "*industrial initiative*" aimed at carving out a space in the test process improvement world; the resulting framework comprised "*a maturity model, test maturity matrix, a checklist, and improvement suggestions*" (pp. 123-24). Fortunately, however, the relationship between *TPI* and TPI NEXT is rather more straightforward: the former was simply the first iteration of the model, whereas the latter represents a revised and updated version rather than a distinct model with a different orientation and emphasis.

Like TMMi, much of the TPI NEXT model, including assessment tools, is publicly available free of charge. However, because the testing suite is actively under development by Sogeti, occasionally in partnership with its parent firm Capgemini, there is arguably a larger expectation to access proprietary tools and services for a complete assessment and recommendation.

This is partially because partnering specialists are promoted as having access to a continuously-updated proprietary testing database; thus, it is suggested that simply

[4] For clarity, italics are used in this report to differentiate Sogeti's model, *TPI*, from the use of "test process improvement" (TPI) as a more general term referring to a functional area rather than a specific model.

attempting to apply the most recently released publicly-available version of the model in-house might generate suboptimal results compared to contracting Sogeti's testing service, for example (Sogeti 2018b).

Like TMMi, TPI NEXT is broadly structured as a maturity model, and in fact shares the first and final stage designations (Initial and Optimizing, respectively) with TMMi. However, the criteria for qualifying for a given maturity level differ between the models, so care should be taken to avoid confusing an Initial or Optimizing maturity level as ranked by TMMi with the levels by the same name under TPI NEXT, as discussed below. In TPI NEXT, the intermediate (second and third) maturity levels are designated Controlled and Efficient, respectively. TPI NEXT is generally characterized by its multifaceted and multidimensional construction of the testing process: it utilizes a series of maturity-level-specific checkpoints in order to evaluate sixteen key areas at each tier:

1. *Stakeholder commitment*
2. *Degree of involvement*
3. *Test strategy*
4. *Test organization*
5. *Communication*
6. *Reporting*
7. *Test process management*
8. *Estimating and planning*
9. *Metrics*

10. *Defect management*
11. *Testware management*
12. *Methodology practice*
13. *Tester professionalism*
14. *Test case design*
15. *Test tools*
16. *Test environment* (Banga 2010, p.1).

These key areas are loosely grouped into three categories ("clusters"): stakeholder relations (1-6), test management (7-11), and test profession (12-16) (Linker & Visser 2009). In addition, TPI NEXT offers "*improvement suggestions and enablers; the outcome of a TPI NEXT analysis is typically visualized in the form of a maturity matrix to highlight a desirable and logical* "improvement sequence" (Fig. 6).

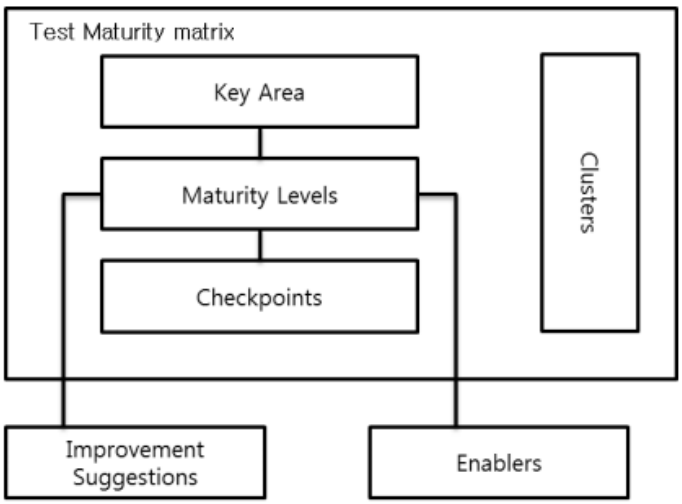

Figure 6: TPI NEXT Test Maturity Matrix Example (Kim & Kim 2014, p. 60).

Unlike TMMi, TPI NEXT offers explicit guidance for customizing goal-setting, objective definition, and execution plans in order to facilitate tailoring and integration with organizational priorities, needs, resource constraints, and capacities.

This intersection between the generic and the customizable is generally attributed to TPI's derivation from more general frameworks, and notably its tendency to construct test maturity and engineering principles using a pre-existing conceptual frameworks surrounding decision support systems.

Another consequence of the versatility of the TPI NEXT as a general framework has been the development of a number of context-specific subtypes of associated tools. For example, the Test Maturity Matrix (TMM) tool, which is commonly used to support audits, including prioritizing interventions and decision-making, is available in a variety of "*flavors*", each tailored to a different "*situation and environment*", such as versions adapted specifically to devops and other Agile environments (Sogeti 2018, p. 1).

In the context of Sogeti's current TPI suite, the more comprehensive and less modular TPI NEXT Model is generally viewed as the "*flavor*" best suited to "*more traditional environments and approaches*", but it is important to keep in mind that closely-related and conceptually similar variants exist to facilitate applications to other environments (p. 1).

Custom tools for facilitating TPI NEXT-based evaluations offer useful analytics surrounding checkpoints and cluster attainment, including benchmark functionalities that facilitate automated data reporting (Fig. 7)

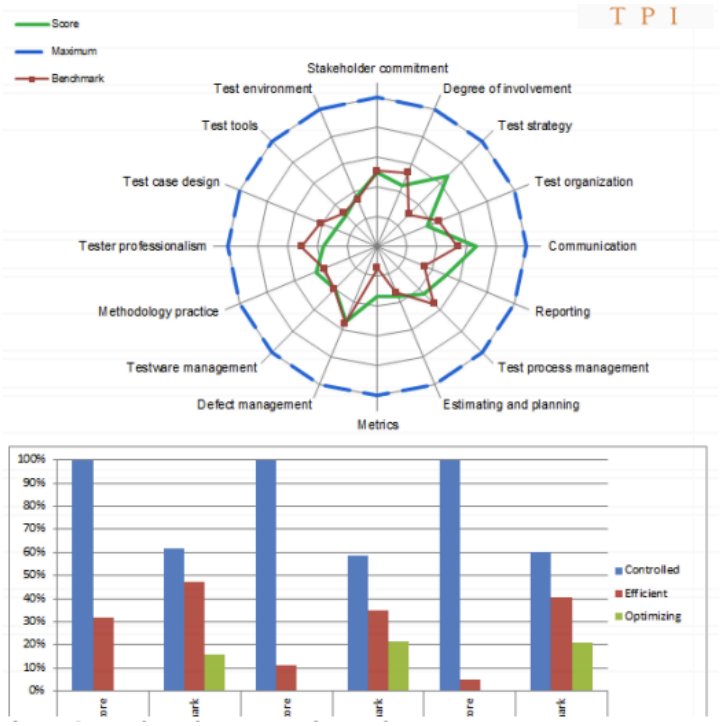

Figure 7: Example Benchmark connected overview from TPI NEXT Test Maturity Matrix Tool (Langebroek 2013, p. 20).

In practice, *TPI* as guided by TPI NEXT is an iterative process that begins with assessing a problem area or problematic outcome, evaluating it in terms of scope, and increasing awareness of the issue in the organizational context.

This is followed by the development and execution of a goal-oriented action plan, which lead back to a re-evaluation of the impact of the solution, beginning the cycle (evaluate scope, set goals, design improvements, develop plans, re-asses, etc.) anew (Aaltio 2013; Langebroek 2013).

3.6 Critical Testing Processes (CTP)

As the name suggests, the CTP model is predicated on the notion that not only are some testing processes more important than others, but in fact certain processes are *critical* to the success of testing efforts: if these processes are in place and functioning effectively, then the testing team will likely generate positive results overall, whereas if they are absent or non-functional, then no amount of managerial intervention is likely to improve their results to the desired level.

Specifically, CTP outlines a dozen general testing processes as critical. These processes are discussed in a little more detail below, but first it may be useful to review the origins and basic conceptual orientation of the model.

Briefly, CTP was designed by Rex Black, a software testing pioneer former president of the ISTQB—as well as the president of its American counterpart, the American Software Testing Qualifications Board, ASTQB (Thomas 2009; Bouguerra 2006). Black viewed existing TPI approaches as excessively bulky on the one hand, but has also vocally contested the context-driven testing view that *"there are no best practices, only good practices, noting that while significant differences exist between projects, it is important to resist the temptation of becoming* "*overly fascinated*" by them and losing sight of common problems (p. 1). Thus, Black views CTP as more "*context-sensitive*" than some approaches, without qualifying as a "context-*driven*" approach (p. 1). Common CTP training materials, therefore, commonly utilize context-specific exercises and training debriefs in order to encourage testing professionals to apply the framework to a range of situations, but in a consistent manner—making the

framework fit the situation without sacrificing its basic logic, rather than seeking to make a given testing context satisfy the prescriptions imposed by the framework (Black 2014).

Despite its name, CTP, like STEP, is perhaps best thought of as a content reference model overall, although it incorporates elements of process model approaches as well (Ahmed 2016). As with the other models, it features a tailoring process that is designed to assess current frameworks in place at the case organization, including identifying challenges, objectives, strengths, and weaknesses, as well as prioritizing process improvements, while seeking to encompass and integrate qualitative as well as quantitative foci of improvement (Fig. 8).

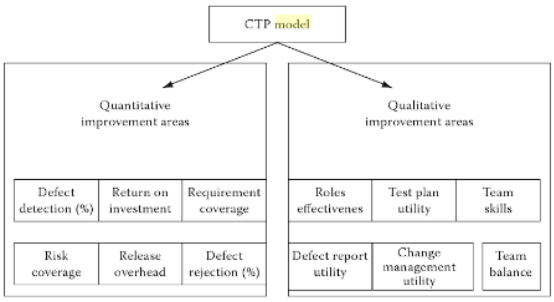

Figure 8: Two domains of the CTP model (Source: Ahmed 2016, p. 242).

This means that actually implementing CTP means beginning with an assessment designed to gather information that will be used to guide the improvement process as it unfolds (although preliminary information can be developed and updated as needed):

"Based on the assessment, a list of process areas to be improved is prepared and prioritized [...] per organizational needs. [...] A plan is

prepared to improve all the weak areas identified in the assessment" (p. 242).

In practice, then despite its modular approach to defining and addressing critical processes (discussed in more detail below), the implementation of the CTP model is typically conceptualized with reference to four sequential stages (Fig. 9).

Figure 9: CTP process steps (Bath & Van Veenendaal 2013, 3-28)

Thus, the first stage effectively requires critical reflection and analysis in order to develop an actionable understanding of the objectives and foci of the testing initiative itself, while the second stage emphasizes the contextual nature of that initiative, calling upon an analysis of key stakeholders, actors, and specific tests involved. In the latter two stages, in turn, testing is actually carried out, results are correlated, and then those results are used to inform the next iteration of the testing cycle by guiding "*adaptation and improvement*" efforts and other modifications to the improvement strategy(ibid.). While the CTP model is designed to offer primarily "*generic suggestions*" about how identified weak areas might be improved, maximizing the effectiveness and efficiency of the improvement initiative requires the implementation team, therefore, to "*tailor*" these recommendations to ensure their concordance with organizational needs (pp. 242-43).

This is, in part, because CTP was originally designed as a "*lightweight*", checklist-based TPI model to compete with more inflexible, burdensome approaches structured within a more comprehensive and intensive approach guided by an (arguably) more bureaucratic mindset (Black 2003; Van Zyl 2010; Gruner & Van Zyl 2011).

In contrast to comprehensive, highly prescriptive models like TMMi and TPI NEXT, CTP is designed to focus the testing team on a handful of test areas which "*simply must*" be done correctly using a framework that was functionally capable of being adapted to "*all software development lifecycle models*" (Black 2010, p. 1).

In designing the CTP model, Rex Black began with two simple definitions. First, he defined *process* as "*some sequence of actions, observations and decisions*"; second, he defined *testing* to signify "*the activities involved in planning, preparing, performing, and perfecting the assessing of the quality of a system*" (p. 1). With these definitions in hand, he identified four fundamental criteria that could be used to identify which of the myriad possible activities meeting the resulting definition of a *test process* could be reasonably considered to be critical that process. Understanding these four criteria is essential to understanding the CTP framework, so it is worth quoting Black at length here:

- *"Is the process repeated frequently, so that it affect the efficiency of the test team and the project team?*
- *Is the process highly cooperative [...] particularly cross-functionally, so that it affects test team and project team cohesion and cooperation?*

- *Is the process visible to peers and superiors, so it affects the credibility of the test team?*
- *Is the process linked to project success, in such a way as to affect project team or test team effectiveness"* (pp. 1-2)

Using these four criteria, Black then derived a working characterization of a critical test process. Specifically, a test process was considered critical if it meaningfully and substantively impacted the capacity of the testing team to perform key tasks like (1) bug identification (2) confidence building (3) risk management, and (4) information generation (p. 2). Ultimately, Black argues that 12 processes meet these essential criteria and merit being classified as critical areas of testing focus.

Somewhat confusingly at first glance, under CTP the first of these processes is testing itself, as broad, aggregate process manifested at a "*macro, strategic level*" and made up of eleven critical component processes:

- *Establishing context*
- *Quality risk analysis*
- *Test estimation*
- *Test planning*
- *Test team development*
- *Test system development*
- *Test release management*
- *Test execution*

- *Bug reporting*
- *Results reporting*
- *Change management (pp. 2-3).*

Generally speaking, CTP preserves the general iterative approach that underpins the other models analyzed here. However, CTP is also notable for its comparative flexibility.

To varying degrees, TMMi, TPI NEXT, and STEP all require a measure of tailoring depending on context, but CTP takes this concept a step further, arguably enabling significant improvements with respect to customization across scales, organizations, and even projects.

This feature can be attributed primarily to its integration of qualitative with quantitative attributes, its explicit attention to organizational strengths and weaknesses from the beginning of the assessment process, and its comparative modularity, which enables not only the prioritization of process improvements, but also permits them to be executed in different sequences depending on the demands of the moment (Black 2003).

With this in mind, it may be useful to briefly examine each of the critical testing processes identified in the model. The first component critical testing process (as opposed to the aggregate process of testing as a whole), *establishing context*, speaks to this emphasis on providing a non-prescriptive, adaptive TPI framework. This process requires explicitly linking testing to contexts at multiple scales through consideration

not only of the project constraints and objectives, but also organizational strengths and weaknesses, as well as other structures and systems currently in place (Hass 2014). Nonetheless, CTP restrains itself from providing even seemingly uncontroversial prescriptions, such as developing "*test policy and test strategy documents*", which is often a needed and appropriate step: Black (2003) points out that while this may be necessary in some contexts, in others it might be superfluous (p. 2).

As the name suggests, the quality risk analysis process orients the testing regime to ensure its responsiveness to major risks to the quality of the system as a whole—risks which, once again, are likely to vary significantly from case to case (Black 2014). Once again, in CTP, this is not a strictly quantitative process, but rather one which encompasses social activity as well, requiring consensus-building among key stakeholder groups regarding issues like "*what is to be tested (and how much) and what is not to be tested (and why).*" (Black 2003, p. 2).

Test estimation, in turn, closely linked to the process of test planning. Test estimation is a process which involves a series of cost-benefit analyses surrounding the implementation of a given testing regime in a specific situation.

By using systematic methods to derive an evidence-based Return Of Investment, it can enable stakeholder buy-in for business leaders with minimal software experience who may remain unconvinced of the importance of testing—or, in some cases, it can identify an over-zealous testing plan that is not warranted by the complexity or criticality of the project itself. Test planning is the product of this stakeholder

consensus-building process, and serves to ensure that test estimation analyses are accurate by clearly mapping responsibilities, activities, priorities, and so forth, against project objectives.

The development of testing teams and systems are also closely-related processes that run slightly against the modular grain of the CTP model (Bath & Van Veenendaal 2013). The former emphasizes the development of productive communicative and collaborative dynamics between testing and the broader project team, seeking to "continuously align team capabilities" with organizational values and critical skills (Black 2003, p. 2). The latter maps test coverage against critical quality risks, ensuring efficiency by balancing "*resource and time requirements against criticality of risk*" while emphasizing the perspectives of customers and end users, rather than developers, the business, or the project team (ibid.).

Test release management is the process by which reliable, functional products are released by the testing team into the "*test environment*" in order to advance the project in a way that ensures the project is developing as intended and on schedule (Black 2014, 5-7). The end of the project development phase is demarcated by the initiation of the test execution process:

> *"This process, the running of test cases and comparison of test results against expected results, generates information about bugs, what works, and what doesn't [...] this is where the value of testing is created" (ibid.).*

As a model, CTP is oriented toward continuous quality improvement; thus, it considers bug reporting to be an equally critical test process insofar as it generates insights capable of driving broader improvements to systems (Van Zyl 2010; Black 2003). While the previous process is often described as generating the value of the testing process, bug testing nonetheless can be thought of as a means by which a significant portion of that value is "*delivered*" to the project team—as well as a qualitative tool for building "*tester credibility with programmers*" (pp. 2-3). Much of the remainder of this value is delivered through the process of reporting results, which gives more senior managers and executives the insights needed to make informed decisions—while simultaneously building "tester credibility" with this group:

> *"The CTP model [...] covers the test levels and activities that are the responsibility of the independent test team. It acknowledges the influence on delivered product quality of other processes, such as reviews and unit testing, but also acknowledges that the influence of the test organization on these processes is often limited. It is unique [...] in that the improvements are explicitly targeted at improving stakeholder satisfaction in the test process along with the effectiveness and efficiency of the test process*" (Black 2014, 5-7).

Finally, change management completes the iterative cycle by encouraging a broader reflection on the project and its limitations through the selection of "*the right changes in the right order*" while focusing future efforts at applying lessons and developing new

strategies on activities with the highest Return Of Investment—ideally, though not necessarily, identified in the course of carrying out the other processes above (ibid.).

3.7 Previous Comparative Research

This review identified a paucity of peer-reviewed comparative studies of TPI models; empirical ones were particularly scarce. Much of the information available on the topic can be traced back to the models themselves and the ways they represent themselves in instructional manuals (Afzal, Alone, Glocksien & Torkar 2016). Two

examples of fairly rigorous comparative studies conducted by scholars are those of Kim & Kim (2014) and Afzal, Alone, Glocksien & Torkar (2016). Briefly, Kim & Kim (2014) argue that if simply assessing test activities and complementary CMM variables are not, in and of themselves, sufficient for achieving holistic or comprehensive test process improvement, then one approach might be to adapt the TMMi model by integrating it with elements of TPI NEXT. However, as their work illustrates, achieving this kind of integration requires a systematic approach, since the "*different mechanism[s]*" offered by these models necessitates a complex mapping process (p. 59). The mapping rules involved, for example, keyword comparison, and higher level concept comparisons used in conjunction with a correlation analysis and series of evaluation rules utilizing a series of comparative statements in order to identify common elements between the frameworks. An example evaluation rule comparator

might be "*An element of TPI NEXT and an element of TMMi have the same meaning*", whereas an alternative comparator might be that element from one has a more comprehensive meaning than the corresponding element from the other (pp. 62-63). The mapping process revealed significant overlap between the functional components of these models; thus, they can likely be differentiated primarily on the rigidity of their prescriptions and the relations of those components to one another.

Other independent comparisons are frequently dated, and thus fail to effectively address current forms of the most dominant models. Swinkels (2000), for instance, offers a comparative evaluation of TMM and TPI, parent frameworks to two of the models examined here (TMMi and TPI NEXT, respectively).

The comparison noted that while both models served the same purpose (to "*identify the current state of practice in key areas*" and "*improve the testing process"*) and should be classified as staged maturity models, the structure of those models is very different: notably, TMM was found to neglect a number of key areas that TPI addresses in detail, including testing and office environments, reporting, and even testware management (pp. 32-37).

More recently, Farooq and Dumke (2008), in turn, develop an evaluation framework and apply it to TMM and TPI in order to carry out a comparative assessment, coming to a conclusion closely resembling Swinkels' nearly a decade prior:

> *"Although [TMM] addresses more process improvement aspects in comparison to the [TPI] model [...] it lacks adequate guidelines on many process improvement issues "(p. 126).*

This gives some insight into the present problem, although it is unfortunately not direct insight, since the subsequent iterations of the models ostensibly sought to improve on these shortcomings. As indicated above, the TMMi involves development in a more action-oriented direction to facilitate improvement, while TPI NEXT has arguably increased its comprehensiveness to encompass more process dimensions. For practical purposes, the most explicit guidance with respect to comparing or choosing between the TPI models discussed here can be found in software testing curricula. The ISTQB (2011), for instance, supports the basic dichotomy between process models on the one hand and content reference models on the other as a strategy for classifying the four models considered in this thesis—a common approach suggested by several authors in the preceding subsections. Once again, TMMi and TPI NEXT belong to the former category, while CTP and STEP belong to the latter.

Notably, the ISTQB curriculum offers some of the only explicit decision-making criteria for testing managers aimed at guiding choices between TPI models identified in this review.

However, the methodology used to generate these criteria has not been made explicit. Additionally, potentially in an effort to avoid endorsing specific models over others (and thus appearing biased), these criteria pertain specifically to the decision between selecting a process model or a content reference model generally; they do not offer much direct guidance in selecting between multiple frameworks in the same category. Briefly, the ISTQB suggests that process models are particularly useful in settings where multiple projects are underway, in part because these models offer tools

designed to facilitate between-project comparisons through benchmarking, provided that those projects are substantively "*similar*" (5.1, p. 36). Because process models are conceptually similar to (or, as in the case of TMMi, are structurally rooted in) more general process improvement models like CMM or CMMi, process-oriented TPI models are also indicated when compatibility with these more general models is desirable. Additionally, we would be remiss if we overlooked the basic value statement of process models: an explicit and clearly-defined "*starting point*" for the test process improvement effort, followed by a road map to improvement. Finally, process models can be useful when it is necessary to succinctly communicate information about the testing process to external actors: for example, it may be useful for "*marketing purposes"* to provide a clear and accepted "*measure of test maturity*", a task for which more flexible content reference models are ill-suited (5.1, p. 36).

Other criteria identified by the ISTQB are more circular, however. For instance, it is suggested that process models may be "*best applied*" if it is "*company policy [...] to attain a specific maturity level (e.g. TMMi Level 3).*" (5.1, p. 36). Similarly, the handbook suggests that process models may be an attractive choice of the organization maintains a pre-existing respect for and openness toward process models. It is unlikely that self-reflexive criteria like these will be especially useful in practical settings: if company policy states that achieving a given TMMi level is an objective, then test managers probably do not need external guidance from researchers to suggest that CTP might not be the ideal model to achieve it.

Major criteria that might be suggestive of a context where a content-reference based approach might be preferred include the absence of a pre-existing formal test process, or alternatively it has been determined that "*discontinuous, rapid improvements and changes to existing test processes*" are in order; additional criteria include statements like:

"An assessment to identify costs and risks associated with the current test process is needed. Improvements do not need to be implemented in the order specified by [process models], but rather in the order determined by business needs

Tailoring is required to ensure the test process fits the company's specific context "(5.1, pp. 36-37).

Once again, it is worth noting that the logic underpinning of the second criterion in the quote above is fundamentally circular: if prescriptive orientation and a staged structure are not desired, then it stands to reason that the models that dispense with this features would be preferred.

To be clear, pedagogical materials of this nature may be a valuable (if imperfect) resource in a practical context. However, they should not be viewed as equivalent to peer-reviewed scholarship, since the processes that produce its content are often opaque and may permit certain biases that may be difficult to identify. Despite these academic criticisms, however, sources of this nature merit serious consideration insofar as they represent accepted industry best practices.

That said, it is worth emphasizing once again that scholarly treatments of TPI frameworks are so scarce—even in comparison to similar analyses of process

improvement more generally (Farooq & Dumke 2008) and Afzal, Alone, Glocksien & Torkar (2016).

4. Analysis & Discussion

Broadly speaking, it is difficult to overlook the significant shortcomings in the research literature reviewed here when considering it as a whole. Because these limitations in data type, scope, and quality directly impact the present study's ability to evaluate its research questions, it may be useful to reverse the standard sequence and begin by discussing some of these limitations before moving on to a more direct analysis of the TPI models examined here.

Despite the wide and growing usage of these TPI frameworks, empirical data surrounding their use and impacts—even including simple implementation case study reports—is intermittent at best. Taken together, the body of academic literature treating the actual use and utility of just these four TPI frameworks—which currently dominate the testing landscape and represent the most widely-used frameworks by general agreement—is remarkably sparse. Indeed, considering (a) the hefty investments that have been (and continue to be) made by firms of all types with respect to improving their test processes, (b) the considerable growth in the development and complexity of test process improvements frameworks, and (c) the fact that explicit efforts to design, implement, and revise systematic TPI approaches have been underway since at least the 1980's, the research literature exploring the actual use of those frameworks is at a surprisingly immature stage of development. In other words, the growth of the scholarly research literature on this topic has quite clearly failed, at least so far, to keep pace with the growing complexity, diversity, and

increasingly widespread use of TPI frameworks overall. Despite these models' reliance on organizational, operational, and information theory, the TPI domain is industry-driven and unwaveringly praxis-oriented.

At the same time, however, despite (or perhaps precisely because of) the diversity of actors that have participated and which continue to participate in driving and shaping the development of the TPI sphere, the actionable information that is publicly available is heavily siloed, and often provided by sources that are not exactly neutral or objective. Moreover, important features of the frameworks themselves appear to be derived from previous approaches which may have received equally little scholarly scrutiny. Investigating this heritage for each framework, and seeking to evaluate the literature surrounding each iteration

Recall, for example, that STEP was originally introduced as part of a proprietary seminar curriculum—on top of which, important parts of its structure and approach were based on external IEEE standards. CTP, in turn, was designed by Rex Black, a former ISTQB and ASTQB president and software entrepreneur, who continues to run Rex Black Consulting Services (RBCS) to provide training and consulting in software testing, which promotes the use of the CTP framework and ISTQB curricula. The TMMi framework originated as a community-driven initiative, and while it is publicly available and comparatively open-source in this regard, its structure and content is nonetheless managed by the TMMi Foundation, which (among other things) offers knowledge certifications (e.g. TMMi Professional, Assessor, Lead Assessor) in partnership with for-profit testing organizations like Pearson VUE ("Our Foundation"

2018). Additionally, the TMMi Foundation is almost certainly the pre-eminent source of case studies regarding the effects of the model's implementation on testing processes and organizations alike. Finally, *TPI* was originally developed by the private firm Sogeti, a Capgemini subsidiary, for whom TPI NEXT serves as a profit vehicle with a proprietary methodology, a proprietary database and toolset that can purportedly significantly enhance the impact of implementation—not to mention consulting services and the firm's "*global team*" of certified TPI specialists (Sogeti 2018a,b). While the TPI NEXT model can be downloaded for free as a PDF, the methodology used to produce the model, as well as updates between versions, is not publicly available or open-source in the way that TMMi has historically been, for instance.

Presumably, Sogeti has a vested in seeking to develop a structure that generates the best results for its clients, but there may be competing incentives as well (e.g. artificial complexity designed to upsell potential clients and encourage them to hire Sogeti consultants).

This state of affairs in which models are closely tied to promoting or managing organizations should already be expected to be conducive to the development of information siloes. However, the disproportionately slow growth of the academic literature is a major contributing factor as well. These organizations dominate the information landscape relating to their respective models, and consequently a substantial the literature reviewed above can be traced back to manuals, guidelines, rationales, and related information developed and by the promulgating organization itself. On top of this, in many cases the organizations associated with each model

appear to produce the bulk of research relating to those models, particularly case study information. While it is true that these organizations are well-positioned to collect and analyze such data and should be encouraged to continue to do so, they also have a vested interest in maintaining the perceived legitimacy and efficacy of their respective models. Consequently, comparative research in the TPI space (including the present project) risks being impacted by a higher proportion of potentially cherry-picked data or studies with an elevated risk of bias. The magnitude of this hypothetical effect is difficult to assess, of course, but it is not unreasonable to assume that it could be mitigated with more active participation by academic researchers undertaking to examine these models in a more objective and value-neutral way, particularly in real-world, practice-oriented contexts.

Additionally, the close ties between organizations and the models they promote and endorse may exert a kind of chilling effect on comparative research. The actors best positioned to carry out this research—the organizations themselves, who generally have access to the most detailed and up-to-date information about the use of their models—rarely have a meaningful incentive to do so. For one thing, data would have to be shared between these organizations. For another, a comparative analysis of results would require significant coordination not only between each organization, but also between the firms that might represent the subjects of the case study (for example)—which is to say, potential clients.

Because independent comparative analyses by scholars are few and far between, and because the major TPI models reviewed above are updated frequently in comparison,

results rapidly become dated, and the publication of such research has generally failed to keep pace with a growing and evolving industry. For instance, comparisons of *TPI* against TMM, or of TPI NEXT against TMMI, might be useful in a limited way, but they would also likely magnify already significant problems surrounding generalizability, reliability, cross-case applicability, for example. Additionally, when scholars have carried out independent comparative analysis of multiple TPI frameworks, they have typically adopted some variation of the more theoretical methodological approach manifested in the present study—which is to say, focusing on the structure of the models themselves and guidelines surrounding their implementation rather than on more empirical case studies aimed at collecting primary data about the impacts of implementation, costs and benefits, and so forth. One common approach involves mapping TPI models either against one another to identify similarities and differences, or against a firm's internal testing protocols. As in the current study, the shared characteristics of the existing research is likely a result of efforts to navigate the various limitations inherent in the extant topical literature.

That being said, the reasons for the slow and patchy growth of scholarly research into the use of major TPI frameworks does not appear to be a topic which has been subjected to detailed analysis. In light of the exploration carried out in this thesis, however, it is possible to make certain inferences about key features of this challenging research space. In addition to the kinds and sources of data that are currently available and the limitations associated with it (discussed in detail above), for instance, the very nature of TPI research, and especially comparative TPI research,

faces a number of distinct methodological challenges. This is especially true for primary research designs.

That said, the specific nature of these challenges depends on the framework itself. A few examples are given below in this section, but in order to properly contextualize this category of limitations, it is time to undertake a specific critical comparison of each of the models analyzed over the course of this study.

With this in mind, then, let us turn our attention to the first guiding questions of this study: namely, what are the key features characterizing each of the four TPI models examined above, and how do these translate into strengths, weaknesses, and common applications?

One criterion that might be used to facilitate the categorization of the models discussed above is simply to ask whether the TPI approach offered requires improvements to be made in a pre-defined, hierarchical sequence. This is the basic paradigm of TPI NEXT and TMMi, whose approaches carefully trace lines of dependency between various TPI-relevant activities.

It is not the case, however, for either STEP or CTP. Of these, CTP is explicitly designed for modularity and flexibility in terms of its application, allowing TPI efforts to be applied in any order based on the needs of the moment: its emphasis is on identifying critical processes that testers absolutely need to do well in order to succeed, and then setting out ideal versions of those processes in order to facilitate benchmarking and assessment on an as-needed basis. STEP offers a comparatively more thorough and systematic approach, but its loose guidelines offer firms and testing teams a general

starting point that requires significant tailoring for effective implementation in a given context. Indeed, in his introduction to *Systematic Software Testing,* one of the seminal texts of the STEP model, CTP progenitor Rex Black (2002) emphasizes the sheer breadth of the highly integrated approach while also stressing the model's own expectations regarding tailoring:

> *"Systematic Software Testing offers a complete roadmap for test professionals looking to institute or improve the way they test software [... Throughout, the authors] remind the reader that the correct answer to any hard question about how to tackle some thorny testing problem is, "It depends"" (pp. 10-11).*

Thus, while STEP and CTP share a common interest in flexibility, modularity, and continuity, they nonetheless fill different niches. CTP works to distil TPI down to its most essential component processes, whereas STEP presents a more integrated overview. Both allow and even require significant discretion on the part of organizations and test managers with respect to their implementation—and both do so to a degree that is not matched by the other two frameworks.

That being said, TMMi and TPI NEXT are not found at equivalent locations along the staged-continuous access. In fact, it should be noted that TMMi differs markedly from the other three models in the rigidity of its staging: TPI NEXT, along with STEP and CTP, are generally viewed as accommodating a substantively higher degree of continuity with respect to prioritization and flexible ordering of improvement sub-processes in this regard ("IDEAL" 2018). Unlike STEP and CTP, however, TPI NEXT

does not offer post-assessment tools or methodologies designed to "*evaluate which process improvements will yield maximum returns*", arguably a major advantage of the flexible and markedly iterative nature of the former two models (p. 1). In lieu of these methods, however, it should be noted that TPI NEXT and TMMi's more comprehensive models do offer more generalized and prescriptively-structured "process improvement road maps", which organizations typically need to adhere to comparatively closely in order to achieve certification at the target maturity stage. TMMi is likely at its most effective in contexts where goals and expectations are clearly defined: it does not provide a quick fix or efficiency assessment, but a guideline for developing robust internal best practices for testing and the improvement of testing processes.

Its staged model is not designed to suggest small, targeted fixes, but rather to guide the overall development and maturation of the entire testing domain. While some gaps exist in its assessment protocols (particularly between formal and informal approaches), overall TMMi should be utilized in contexts where organizational needs, goals, and expectations are clear, and where a premium is placed on systematic and comprehensive TPI—and where a long-term investment in TPI is a possibility. On the other hand, TPI may also be attractive for smaller organizations with limited resources whose testing practices may be closer to the Level 1 (Initial) type. This is because the TMMi model as well as its assessment protocols are publicly-available, free of charge. If a young organization wishes to improve its testing processes (or at least gain evidence-based guidance in the form of a template for structuring testing activities or

identifying key areas) but cannot afford to hire an expensive consultant, then TMMi might represent a useful starting point because it can be internally implemented, without relying on external consultants or databases, thus lowering the upfront costs of test process improvement.

At the same time, due to its extremely comprehensive, detailed, and highly structured nature, it is also true that such an organization would probably only be advised to implement TMMi with an eye to future development. In other words, TMMi does not offer a lean framework, and it is designed to maximize its impact when organizations follow its road map as far as possible: it is for organizations that envision themselves as developing a high level of maturity and making significant investments in refining testing processes and integrating them thoroughly with the development life cycle. Organizations who are interested in implementing a TPI model not as a catalyst for organizational growth and development, but rather to simply beef up certain narrowly-defined aspects of their current testing process in order to meet the demands of a particularly critical or unusually complex project, are likely to find TMMi's approach to be rather unwieldy. The TPI NEXT model offers a somewhat more dynamic and customizable tool, but overall it would face many of the same issues as TMMi for an organization of this kind. In such cases where a highly targeted, minimalist approach is desirable, CTP is likely to be the most attractive candidate.

The first and most obvious methodological issue, discussed above, involves how researchers should respond to the role of proprietary information in frameworks like TPI NEXT. In case study research, if the researcher is coordinating the TPI

intervention, then would forgoing access to Sogeti's TPI NEXT database and relying exclusively on publicly available information provide a reasonably fair assessment of the model, for instance? A related issue that is common to case study research more generally centers on the confidentiality concerns of the case organization. A maturity model might result in a less-than-ideal final rating for the case organization even after implementation, for instance. Case organizations would also have to be willing to give researchers wide latitude in the type of data they set out to collect, as well as significant influence over their testing framework. After all, even more modular, continuous approaches like CTP do not merely call changes to existing testing processes, but generally require some kind of re-evaluation of how testing relates to more general development, managerial, and business processes.

In a similar vein, it is highly unlikely that implementing two different TPI frameworks in the same organization to compare their costs and benefits would constitute a methodologically sound approach. For this reason, comparative process improvement studies generally seek to utilize different but comparable firms as case organizations. There are two major problems associated with attempting to carry this approach over into a comparative TPI study, however. First, how would the comparability of firms be assessed initially?

After all, each framework carries with it its own approach to evaluating the existing testing regime and understanding its baseline. Would the researchers use, say, TMMi to screen some number of candidate firms for comparability, but then disregard that evaluation for half of them and employ TPI NEXT instead (for example)?

Second, while the degree varies, each of the frameworks analyzed here requires an extremely high degree of tailoring to meet organizational strategic objectives and performance goals. If CTP were employed in two apparently comparable organizations, for instance, this tailoring process could result in a scenario in each made very different changes to its overall testing process compared to the other—potentially reflecting little more than managerial preference. Which would be representative of CTP's "*true*" intervention? This question becomes even more problematic in the case of a study comparing a CTP-based TPI intervention in Organization A to a STEP-based TPI intervention for Organization B. Seemingly the only way to overcome these barriers in order to meaningfully enhance the validity and reliability of results with direct comparative research involving highly tailorable models would be to utilize large samples and larger datasets—a very difficult prospect in case study research. An alternative would be to conduct more general comparisons based on benchmarking and internal performance data, but again, cross-organizational variabilities, differences in how each model defines and constructs the testing process, and the difficulty in calculating testing ROI between a small number of projects on a relatively short timescale, can all create significant methodological complications.

5. Conclusion

This thesis has sought to offer a systematic, wide-ranging, and evidence-based examination of the four models that enjoy the broadest use in today's test process improvement landscape: Testing Maturity Model Integration (TMMi), Test Process Improvement NEXT (TPI NEXT), Critical Testing Processes (CTP), and Systematic Test and Evaluation Process (STEP). This exploration was carried out by way of a literature review methodology. Using a range of sources drawn from the scholarly, industry, professional, and popular literatures, it identified the key structural and functional features associated with each of these models, with an emphasis on features which can be used to differentiate them from one another (Research Question 1). It also sought to critically evaluate the strengths and weaknesses of each of these models, since there are certain trade-offs made in each, and to the greatest extent possible, to identify some of their most common or characteristic applications (Research Question 2). Finally, based on this information, it identified several evidence-based considerations that might be used by organizations seeking to determine which model should be employed for a specific project or context (Research Question 3).

With respect to the first research question, several different approaches to characterizing these TPI models were identified. In the most general terms, for instance, they can be differentiated from one another structurally on the basis of flexibility, continuity, modularity, and comprehensiveness. It is also possible to differentiate them from entirely different paradigms, however. One example would be

to distinguish these models based on their origin and nature of current management, from being open-source, community-driven, and non-profit-managed (TMMi) or having been designed by a single testing researcher and managed partly in connection to a related consulting enterprise (CTP), to having roots in industry documentation standards and organic growth following introduction in a professional seminar curriculum (STEP) or emerging as a corporate framework with licensed consultants needed to access proprietary databases (TPI NEXT).

These origin stories may be relevant dimensions for evaluating the reliability of the literature and availability of information about these frameworks—and future researchers should attend to them closely—but for the purposes of the current study functional, structural, and practical considerations are arguably somewhat more relevant.

In most treatments of the topic, STEP and CTP are grouped together as relatively continuous content reference models, in contrast to TMMi and TPI NEXT, which are largely framed as staged maturity process models (although the rigidity of this staging can vary significantly). There are occasional discrepancies in this general division, but in the main they appear to be based on technicalities, and the general notion that the former two models can be more flexibly implemented than the latter seems to be widely supported by the models' proponents and by scholars who have assessed them. Broadly speaking, the scholarly literature (limited though it is) and the analysis presented in this thesis are consistent with the general recommendations from pedagogical texts (e.g. ISTQB curricula) regarding the different optimal uses of process

and content-reference models. Process models should be a preferred choice for test managers in organizational contexts involving multiple projects, and particularly where there is a need to continuously evaluate test process outcomes using between-project benchmarks. Similarly, where software development process improvement models like CMMi are already in place, the implementation of process models is indicated to enhance compatibility. Staged maturity process models are also desired when there is a need for a clear diagnosis of the current state of the testing process against external criteria, followed up a road map to improvement. Continuous, content-reference models, on the other hand, are often preferable to their more unwieldy counterparts when formal testing processes are missing or when a sharp departure from the status quo is required. When implementing TPI efforts to accord to business goals rather than testing comprehensiveness is a decisive factor, these more lean and modular models should be selected.

Guidance regarding which model to choose within each of these dichotomous groups is somewhat more difficult to come by, however. That being said, this review has produced some basic findings that might be used to guide decision-making in this regard.

In fact, differences—and sometimes significant ones—were identified within each of these pairs (TMMi and TPI NEXT versus CTP and STEP) in terms of flexibility and tailoring. Although CTP is, in some sense, rigidly prescriptive insofar as it defines certain processes that it takes to be absolutely vital to the success of a testing regime, for instance, it is widely known for its modularity, allowing organizations to schedule

and prioritize improvements as needed without departing from the framework itself. Most accounts take CTP to be more flexible in this regard than STEP, despite the fact that the latter requires significant tailoring in order to implement it in a given organization setting. Conversely, of the staged models, TMMi is widely regarded as the most inflexible, strictly defining detailed hierarchies and outlining dependencies between testing subprocesses.

To group these models by comprehensiveness, in turn, generates an approximately inverse ranking. Here, TMMi arguably looms above the competition: its inflexible design enables the model to offer an extremely comprehensive, thorough, an detailed blueprint not only of an optimal test process design, but also of the interaction between testing and development—and even define how to get there. At the other end of the spectrum, CPT requires a high degree of organizational discretion in terms of prioritizing TPI improvements, and the guidance it is able to offer in this regard is quite general and even minimal. Similarly, the tailoring required to implement STEP in a manner that is compatible with organizational context can risk leaving certain organizations with limited guidance. A similar difficulty potentially exists with TPI NEXT, although in this case it can presumably be mitigated through the hiring of certified model experts with access to the database. That said, this review was unable to identify independent empirical research to support the notion that such an investment meaningfully improves TPI outcomes.

On the whole, then, strengths, weaknesses, and common applications of these models can be assessed both in terms of their categorization as process models versus content

models, as well as between models belonging to the same category. Unfortunately, however, evidence-based criteria that might be used to identify the model best suited to a specific project are difficult to derive from the data analyzed in this study. Simply put, too much literature traces back to manuals describing the models themselves, and empirical evaluations that would be needed to produce the evidence for such a decision-making matrix are few, far between, and often dated, comparing predecessors of the frameworks currently in use today.

In lieu of robust empirical evidence, then, test managers must rely on the results of theoretical studies, organizational needs (as well as strengths and weaknesses), project constraints, and industry best practices (as encapsulated by curricula like those offered by the ISTQB). The guidance given by these sources effectively amounts to the recommendations given above regarding process versus content reference models, and within each of those groups, differences based on flexibility and modularity.

An unexpected but highly significant finding of this project, however, centers on the surprising lack of a systematic, objective evidentiary basis that would enable these research questions to be addressed more conclusively. As it is, this thesis's efforts to address do so have relied heavily on extrapolation and inference on the one hand, and on the other hand, on representations of these TPI models by the very organizations responsible for producing, managing, and promoting them. This issue has been subjected to a thorough and detailed discussion in the analysis section above, but given its significance, these gaps should be emphasized. The notion that TPI theory and

practice stands to benefit significantly from efforts to fill these gaps through targeted, rigorous research is virtually beyond debate.

In closing, then, it is worth returning once more to the importance of software testing and the need to continue to improve test processes in systematic and evidence-based ways.

Today, software is integrated into the fabric of daily life to an unprecedented (and still growing) degree, including in extremely sensitive and critical tasks. At the same time, the growth of computing power and the expansion of the Internet both globally and through Internet of Things (IoT) applications will continue to drive the need for efficient development approaches capable of complex but highly reliable products. This needs to be accomplished without ballooning the already-significant share of project and IT budgets devoted to testing. In order to achieve these goals in the face of the trends above and accelerating technological change, firms of all types will need to continue to invest in improving the efficiency and effectiveness of their test processes in a systematic and sustainable way.

To support them in this effort, however, additional research is strongly indicated, particularly from the academic community. Independent, objective analysis not affiliated with organizations managing dominant TPI frameworks are needed. Ideally, this research agenda should prioritize primary empirical data regarding the implementation and outcomes associated with available models, including fair and reliable assessments of costs and benefits. Similarly, comparative analyses based on primary and secondary data alike might be used to foster the development of more

robust, detailed, and comprehensive decision-making frameworks for organizations seeking to choose the best model for a given project or context. Even if significant progress is made, however, it should be noted that this will represent an ongoing project, as many TPI models are continuously changing in response to both industry needs and updates to maintain currency. Even if research fails to keep pace with these changes, however, there is adequate room for studies to make headway with respect to filling the significant and pervasive gaps in the peer-reviewed literature discussed here. Overall, therefore, further scholarly attention to this important and understudied topic is strongly indicated.

6. References

- Aaltio, T. (2013) Test process improvement with TPI NEXT: what the model does not tell you and what you should know. Accessed 22nd May 2018. https://www.slideshare.net/VLDCORP/test-process-improvement-with-tpi-next-what-the-model-does-not-tell-you-but-you-should-know

- Afzal, W., Alone, S., Glocksien, K., & Torkar, R. (2016). *Software test process improvement approaches: A systematic literature review and an industrial case study*. Journal of Systems and Software, 111, 1-33.

- Ahmed, A. (2016). *Software project management: a process-driven approach*. Auerbach Publications.

- Ahner, D., Wisnowski, J., & Simpson, J. R. (2017). Automated software testing in the DoD: current practices and opportunities for improvement. *Journal of Defense Analytics and Logistics*, *1*(1), 88-91.

- Ahem, D., Clouse, A., & Turner, R. (2001). *CMMI Distilled.* New York: Addison-Wesley.

- *Advanced Level Syllabus Test Manager.* (2012). International Software Testing Qualifications Board (ISTQB). Accessed 22nd May 2018. https://www.istqb.org/certification-path-root/foundation-level/foundation-level-material-for-download/foundation-level-exam/category/7-advanced-level-documents.html

- Ammann, P., & Offutt, J. (2016). *Introduction to software testing*. Cambridge University Press.

- Banga, K. (2010). Review: TPI NEXT – Business driven test process improvement by Gerrit de Vries et al. (2010). London: UTN. Accessed May 22nd 2018. https://www.bcs.org/content/conWebDoc/35411

- Bath, G., & Van Veenendaal, E. (2013). *Improving the Test Process: Implementing Improvement and Change-A Study Guide for the ISTQB Expert Level Module*. Rocky Nook, Inc.

- Black, R. (2003). *Critical Testing Process: Plan, Prepare, Perform, Perfect*. Addison-Wesley Longman Publishing Co., Inc.

- Black, R. (2010) *Critical testing processes: an open source, business driven framework for improving the testing process.* RBCS US. Accessed May 22nd 2018. https://rbcs-us.com/site/assets/files/1170/critical_testing_processes.pdf

- Black, R. (2014). *Advanced Software Testing-Vol. 2: Guide to the ISTQB Advanced Certification as an Advanced Test Manager*. Rocky Nook, Inc..

- Bougerra, Y. (2006) Interview with Rex Black. *British Computer Society.* January 2006. Accessed May 22nd 2018. https://www.bcs.org/content/ConWebDoc/3061

- Bris, P., Frantis, M., & Kolková, M. (2015). Software quality control with the usage of ideal and TMMI models. *MM Science Journal*. December 2015, 799-807

- Bertolino, A. (2007, May). Software testing research: Achievements, challenges, dreams. In *2007 Future of Software Engineering* (pp. 85-103). IEEE Computer Society.

- Burnstein, I., Homyen, A., Grom, R., & Carlson, C. R. (1998). A model to assess testing process maturity. *Crosstalk*, *11*(11), 26-30.

- Burnstein, I., Suwanassart, T., & Carlson, R. (1996, October). Developing a testing maturity model for software test process evaluation and improvement. In *Test Conference, 1996. Proceedings., International* (pp. 581-589). IEEE.

- Butt, F. L., Bhatti, S. N., Sarwar, S., Mohsen, A., & Saboor, A. (2017). Optimized Order of Software Testing Techniques in Agile Process-A Systematic Approach. *International journal of advanced computer science and applications*, *8*(1), 347-354.

- *Certified Tester: Expert level syllabus: Improving the testing process.* (2011). International Software Testing Qualifications Board (ISTQB).

- Craig, RD & Jaskel, SP (2002). *Systematic Software Testing.* Artech House.

- Dawson, R., Bones, P., Oates, B. J., Brereton, P., Azuma, M., & Jackson, M. L. (2003, September). Empirical methodologies in software engineering. In *Software Technology and Engineering Practice, 2003. Eleventh Annual International Workshop on* (pp. 52-58). IEEE.

- De Souza, E. F., de Almeida Falbo, R., & Vijaykumar, N. L. (2015). Knowledge management initiatives in software testing: A mapping study. *Information and Software Technology*, *57*, 378-391.

- Doolin, B., & McLeod, L. (2005). 12 Towards critical interpretivism in IS research. *Handbook of critical information systems research*, 244.

- Farooq, A., & Dumke, R. R. (2008). Developing and applying a consolidated evaluation framework to analyze test process improvement approaches. In *Software Process and Product Measurement* (pp. 114-128). Springer, Berlin, Heidelberg.

- Garcia, C., Dávila, A., & Pessoa, M. (2014, November). Test process models: Systematic literature review. In *International Conference on Software Process Improvement and Capability Determination* (pp. 84-93). Springer, Cham.

- Gruner, S., & Van Zyl, J. (2011). Software testing in small IT companies: A (not only) South African problem. *South African Computer Journal*, *47*(1), 7-32.

- Harrold, M. J. (2000, May). Testing: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 61-72). ACM.

- Hass, A. M. (2014). *Guide to advanced software testing*. Artech House.

- Huisko, M & Kyyro, M. (2015). Agile testing: improving the process: the case of Descom. *Business Information Systems Theses* Jamk University of Applied Sciences School of Businesses and Service Management, 1-89.

- Kaur, M., & Singh, R. (2014). A Review of software testing techniques. *International Journal of Electronic and Electrical Engineering*, *7*(5), 463-474.

- Kim, K., & Kim, R. Y. C. (2014). Improving test process for test organization assessed with TMMi based on TPI NEXT. *International Journal of Software Engineering and Its Applications*, *8*(2), 59-66.

- Kneuper, R. (2008). *CMMI: Improving Software and Systems Development Processes Using Capability Maturity Model Integration*. Rocky Nook.

- Langebroek, T. (2013). *User manual: TPI NEXT test maturity toolbox.* v. 1.2.1. Sogeti. Accessed May 22nd 2018. http://www.tmap.net/sites/default/files/TPI_NEXT_Test_Maturity_Matrix_tool_User_Manual_v1_2_1_0.pdf

- Lewis, W. E. (2000). *Software testing and continuous quality improvement*. Auerbach publications.

- Linker, B., & Visser, B. (2009). TPI® NEXT: Test Process Improvement Next.

- Madeyski, L., & Kawalerowicz, M. (2018). Continuous Test-Driven Development: A Preliminary Empirical Evaluation Using Agile Experimentation in Industrial Settings. In *Towards a Synergistic Combination of Research and Practice in Software Engineering* (pp. 105-118). Springer, Cham.

- Myers, G. J., Badgett, T., Thomas, T. M., & Sandler, C. (2004). *The art of software testing*. Hoboken, NJ: John Wiley & Sons.

- Ng, S. P., Murnane, T., Reed, K., Grant, D., & Chen, T. Y. (2004). A preliminary survey on software testing practices in Australia. In *Software Engineering Conference, 2004. Proceedings. 2004 Australian* (pp. 116-125). IEEE.

- Nolan, R. L. (1973). Managing the computer resource: a stage hypothesis. *Communications of the ACM*, *16*(7), 399-405.

- Orso, A., & Rothermel, G. (2014, May). Software testing: a research travelogue (2000–2014). In *Proceedings of the on Future of Software Engineering* (pp. 117-132). ACM.

- "Our Foundation" (2018). TMMi Foundation. Accessed May 22nd 2018. https://www.tmmi.org/

- Paulk, M. C. (2009). A history of the capability maturity model for software. *ASQ Software Quality Professional*, *12*(1), 5-19.

- Pressman, R. S. (2005). *Software engineering: a practitioner's approach*. Palgrave Macmillan.

- Rasking, M. (2011, May). Experiences developing TMMi® as a public model. In *International Conference on Software Process Improvement and Capability Determination* (pp. 190-193). Springer, Berlin, Heidelberg.

- Rungi, K., & Matulevičius, R. (2013, October). Empirical Analysis of the Test Maturity Model Integration (TMMi). In *International Conference on Information and Software Technologies* (pp. 376-391). Springer, Berlin, Heidelberg.

- Samalikova, J., Kusters, R. J., Trienekens, J. J., & Weijters, A. J. M. M. (2014). Process mining support for Capability Maturity Model Integration-based software process assessment, in principle and in practice. *Journal of Software: Evolution and Process*, *26*(7), 714-728.

- Saunders, M. and Tosey, P. (2013). The layers of research design. *Rapport, Winter*, *2013*, 58-59.

- Singh, S., Singh, G., & Singh, S. (2010). Software testing. *International Journal of Advanced Research in Computer Science*, *1*(3).

- Slaughter, S. A., Harter, D. E., & Krishnan, M. S. (1998). Evaluating the cost of software quality. *Communications of the ACM*, *41*(8), 67-73.

- *Standard Glossary of Terms Used in Software Testing.* (2016) V3.1. International Software Testing Qualifications Board. Accessed May 22nd 2018. https://www.istqb.org/downloads/send/20-istqb-glossary/186-glossary-all-terms.html

- Sulaiman, N. A., Kassim, M., & Saaidin, S. (2010, June). Systematic test and evaluation process (STEP) approach on shared banking services (SBS) system identification. In *Education Technology and Computer (ICETC), 2010 2nd International Conference on* (Vol. 5, pp. V5-219). IEEE.

- Swinkels, R. (2000). A comparison of TMM and other test process improvement models. *MB-TMM Project Report*, 12-4.

- Tedre, M. (2007). Know your discipline: Teaching the philosophy of computer science. *Journal of Information Technology Education: Research*, *6*, 105-122.

- Thomas, MW (2009). Tech-savvy entrepreneur bolsters clients' computing powers. *San Antonio Business Journal.* 19 July 2009. Accessed May 22nd 2018. https://www.bizjournals.com/sanantonio/stories/2009/07/20/story7.html

- *TPI Downloads: Test Maturity Matrix Tool.* (2018a). Sogeti TMap. Accessed May 22nd 2018. http://www.tmap.net/tpi-downloads

- *TPI: A global team of testing process improvement specialists.* (2018b). Sogeti. Accessed May 22nd 2018. https://www.sogeti.com/solutions/testing/tpi/

- *TMMi Model.* (2018). TMMi Foundation. Accessed May 22nd 2018. https://www.tmmi.org/tmmi-model/

- van Veenendaal, E., Hendriks, R., van de Laar, J., & Bouwers, B. (2008). Test Process Improvement using TMMi. *Testing Experience: The Magazine for Professional Testers*, *3*(19), 21-25.

- van Veenendaal, E. (ed.) (2010). *Standard Glossary of Terms Used in Software Testing.* V2.1. International Software Testing Qualifications Board.

- van Veenendaal, E. (ed.) (2012). *Test Maturity Model Integration (TMMi): Release 1.0.* TMMi Foundation. Accessed May 22nd 2018. https://www.tmmi.org/wp-content/uploads/2016/09/TMMi.Framework.pdf

- Van Zyl, J. (2010). Software Testing in a small company. Techincal Report. University of Pretoria. Accessed May 22nd 2018. http://www.up.ac.za/media/shared/Legacy/sitefiles/file/44/1026/2163/8121/innovate5/softwaretestinginasmallcompany.pdf

- "What is the IDEAL model for test process improvement?" (2018). ISTQB Exam Certification. Accessed May 22nd 2018. http://istqbexamcertification.com/ideal-model-for-test-process-improvement/

- *World Quality Report 2014-2015 (6th ed).* (2015). Capgemini, MicroFocus, & Sogeti. Accessed May 22nd 2018. https://www.capgemini.com/wp-content/uploads/2018/01/world-quality-report-seured-2017-181.pdf?utm_source=pardot&utm_medium=email&utm_content=none_none_none_report_none&utm_campaign=optimize_wqr

- *World Quality Report 2017-2018 (9th ed.*) (2018). Capgemini, MicroFocus, & Sogeti. Accessed May 22nd 2018. https://www.capgemini.com/wp-content/uploads/2018/01/world-quality-report-seured-2017-181.pdf?utm_source=pardot&utm_medium=email&utm_content=none_none_none_report_none&utm_campaign=optimize_wqr

7. Appendix

Figure 1

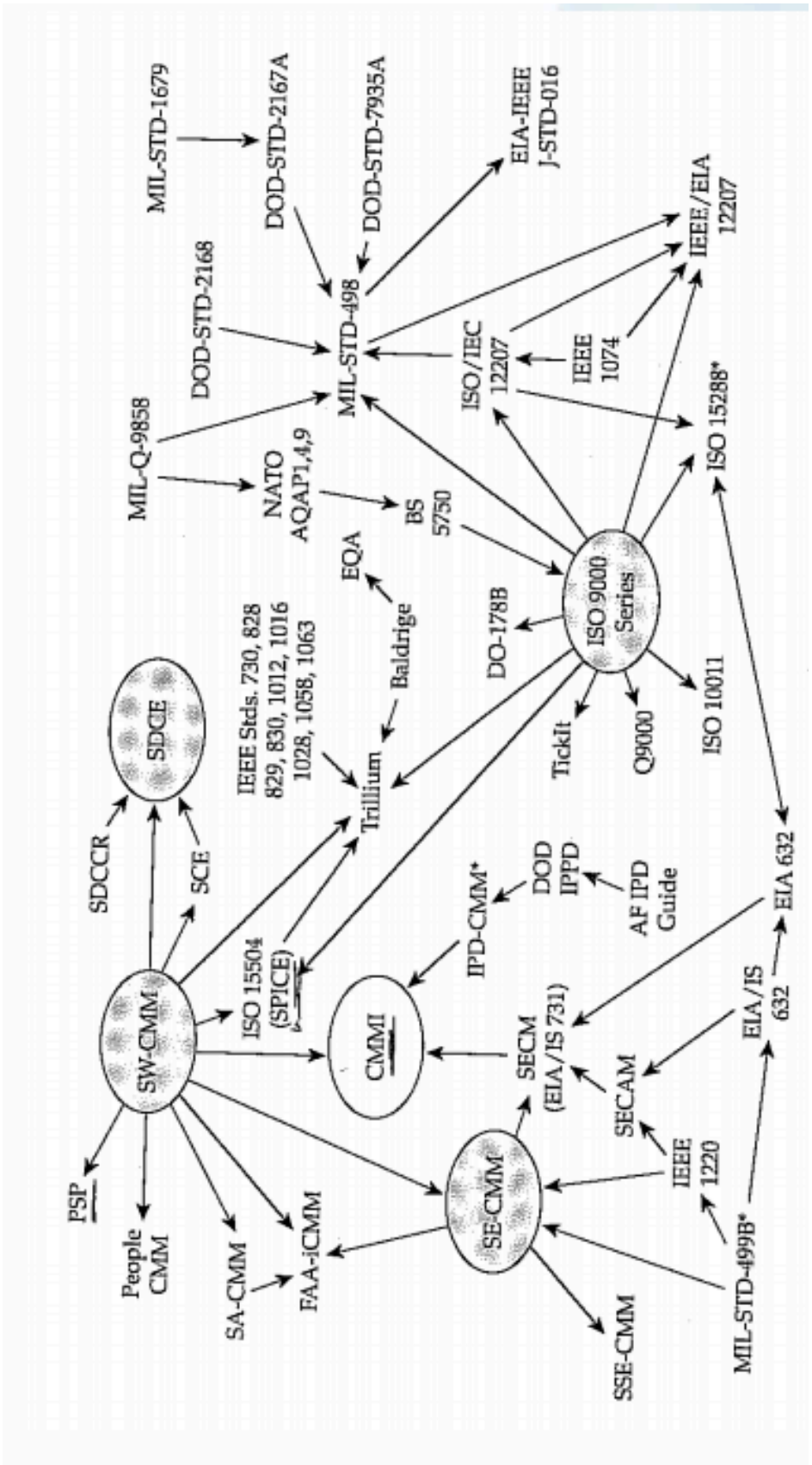

Figure 2

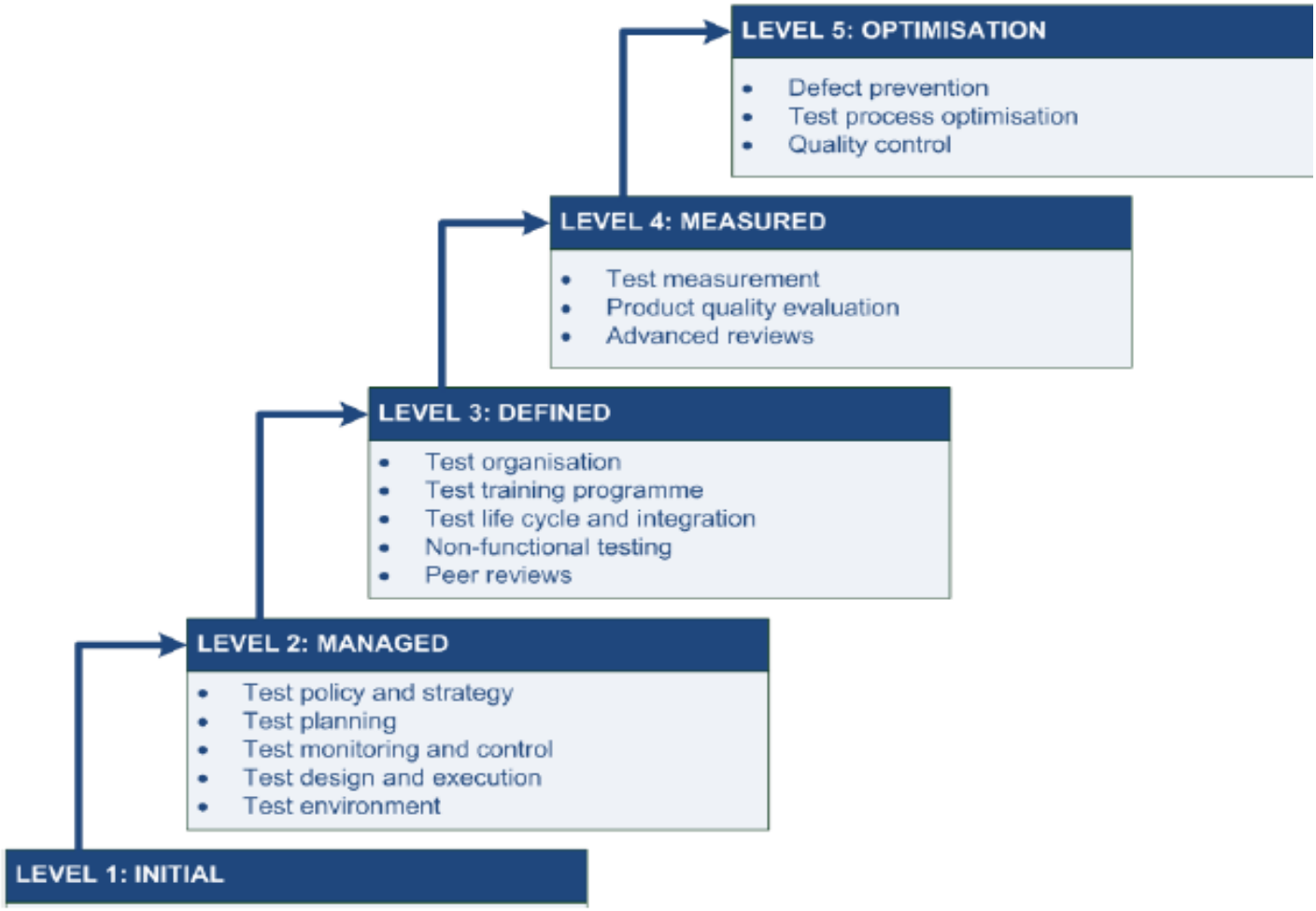
LEVEL 5: OPTIMISATION
Defect prevention
Test process optimisation
Quality control
LEVEL 4: MEASURED
Test measurement
Product quality evaluation
Advanced reviews
LEVEL 3: DEFINED
Test organisation
Test training programme
Test life cycle and integration
Non-functional testing
Peer reviews
LEVEL 2: MANAGED
Test policy and strategy
Test planning
Test monitoring and control
Test design and execution
Test environment
LEVEL 1: INITIAL

Figure 3

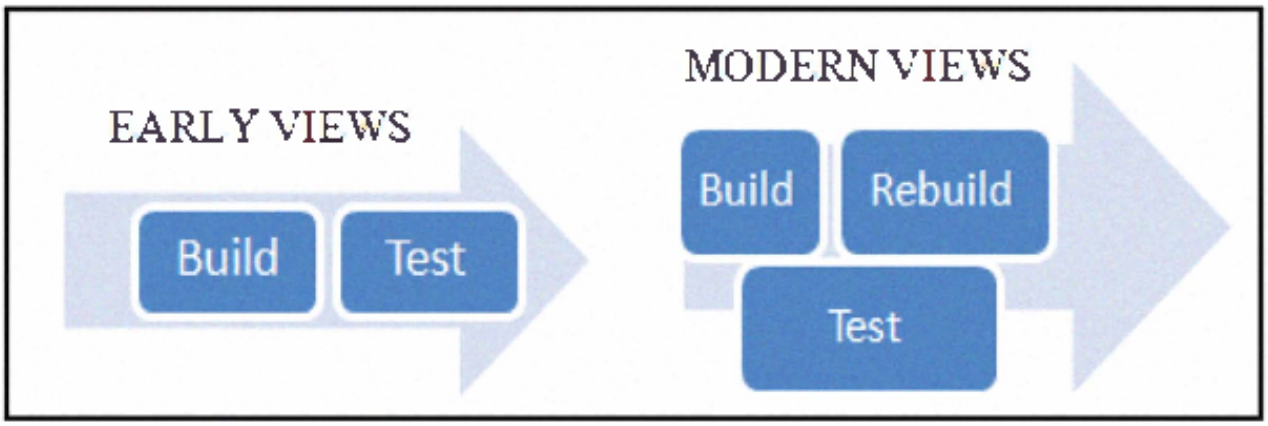

Figure 4

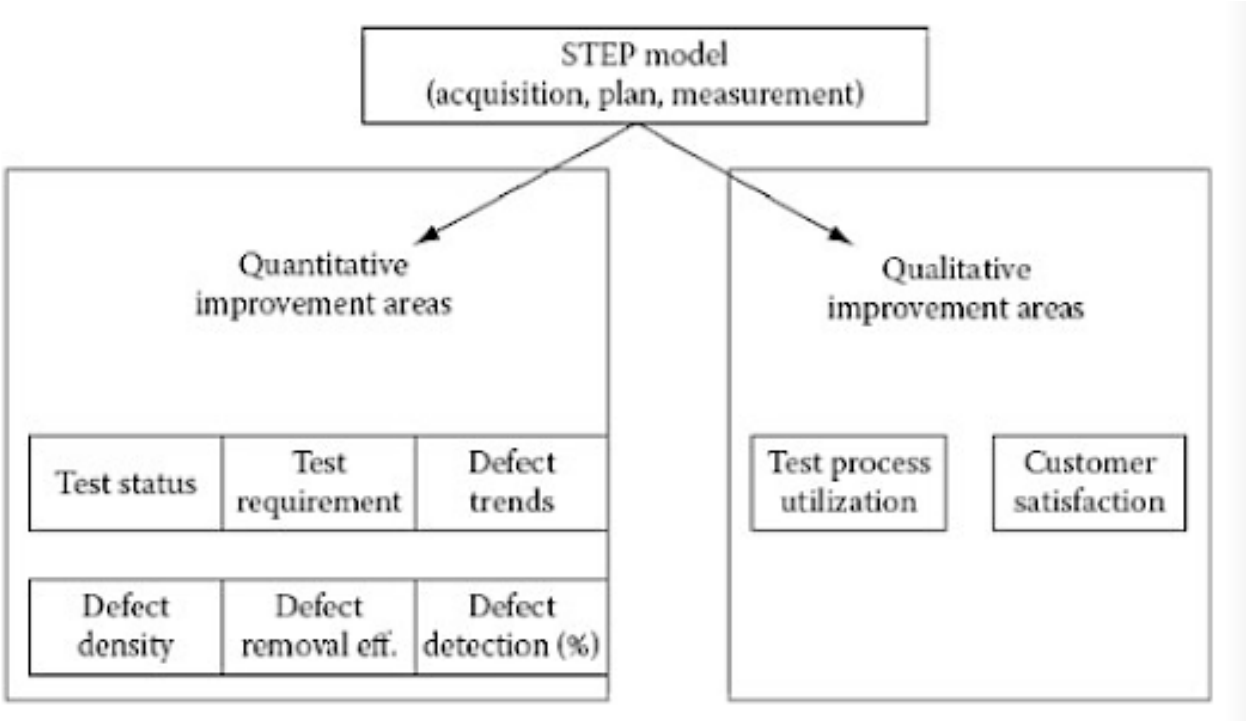

Figure 5

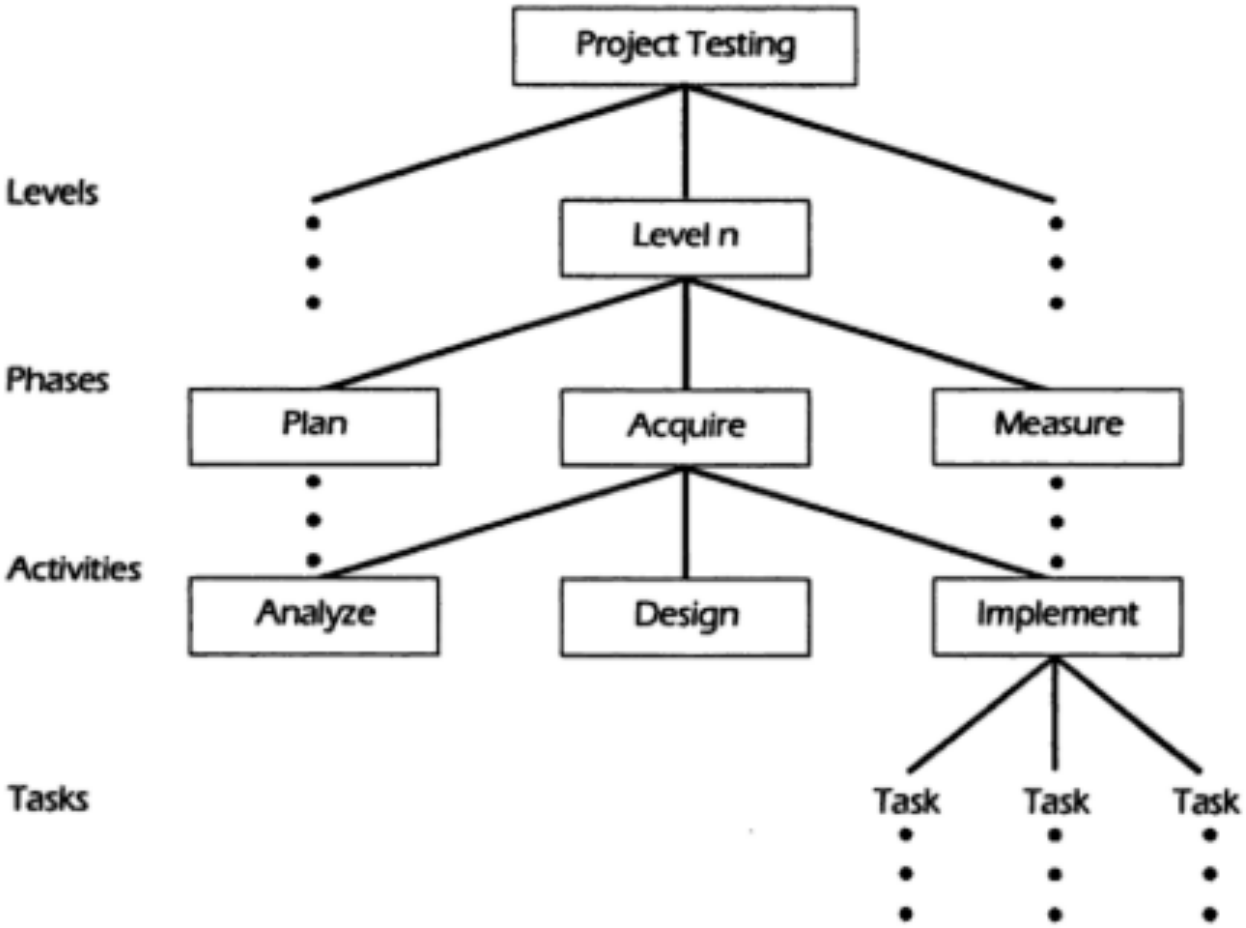
Project Testing
Levels
Level n
Phases
Plan
Acquire
Measure
Activities
Analyze
Design
Implement
Tasks
Task
Task
Task

Figure 6

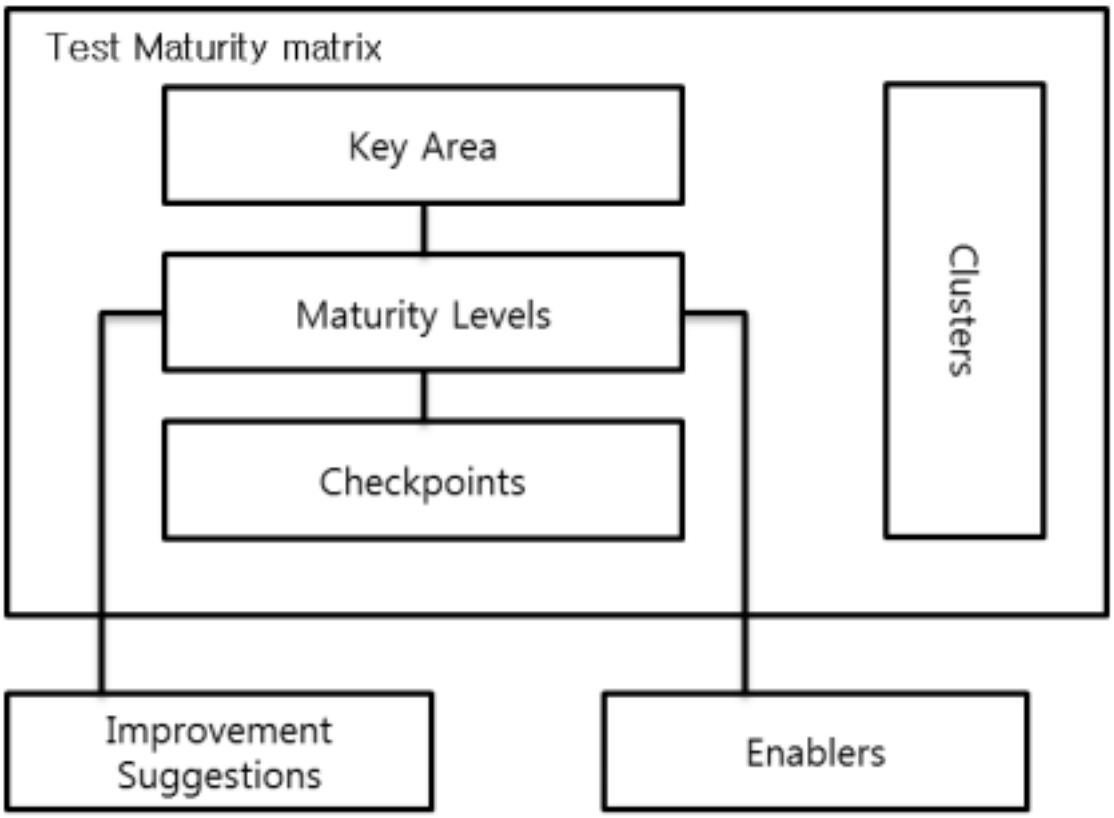
Test Maturity matrix
Key Area
Maturity Levels
Checkpoints
Clusters
Improvement Suggestions
Enablers

Figure 7

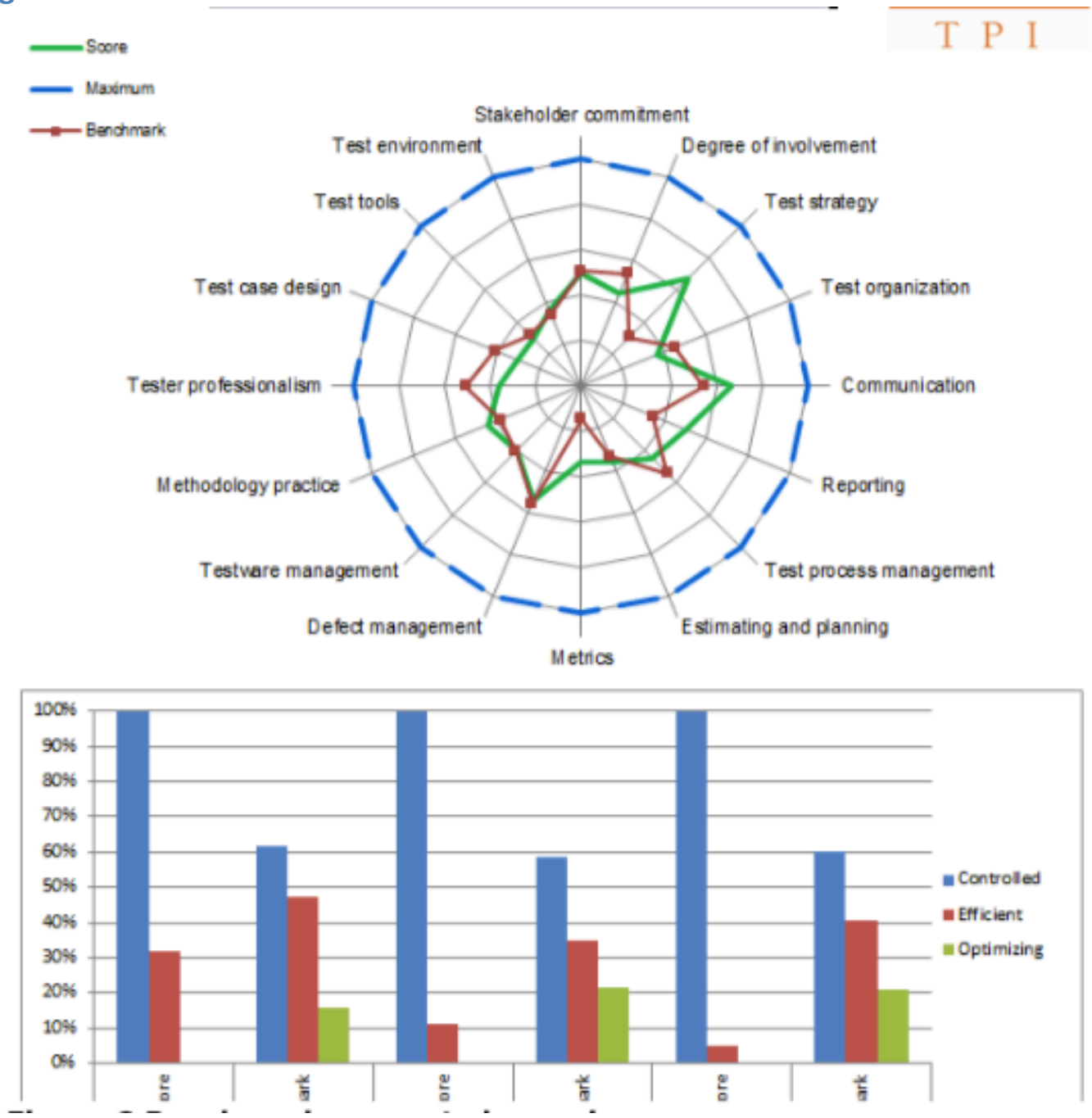
Score
Maximum
Benchmark
Stakeholder commitment
Degree of involvement
Test strategy
Test organization
Communication
Reporting
Test process management
Estimating and planning
Metrics
Defect management
Testware management
Methodology practice
Tester professionalism
Test case design
Test tools
Test environment
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Controlled
Efficient
Optimizing

Figure 8

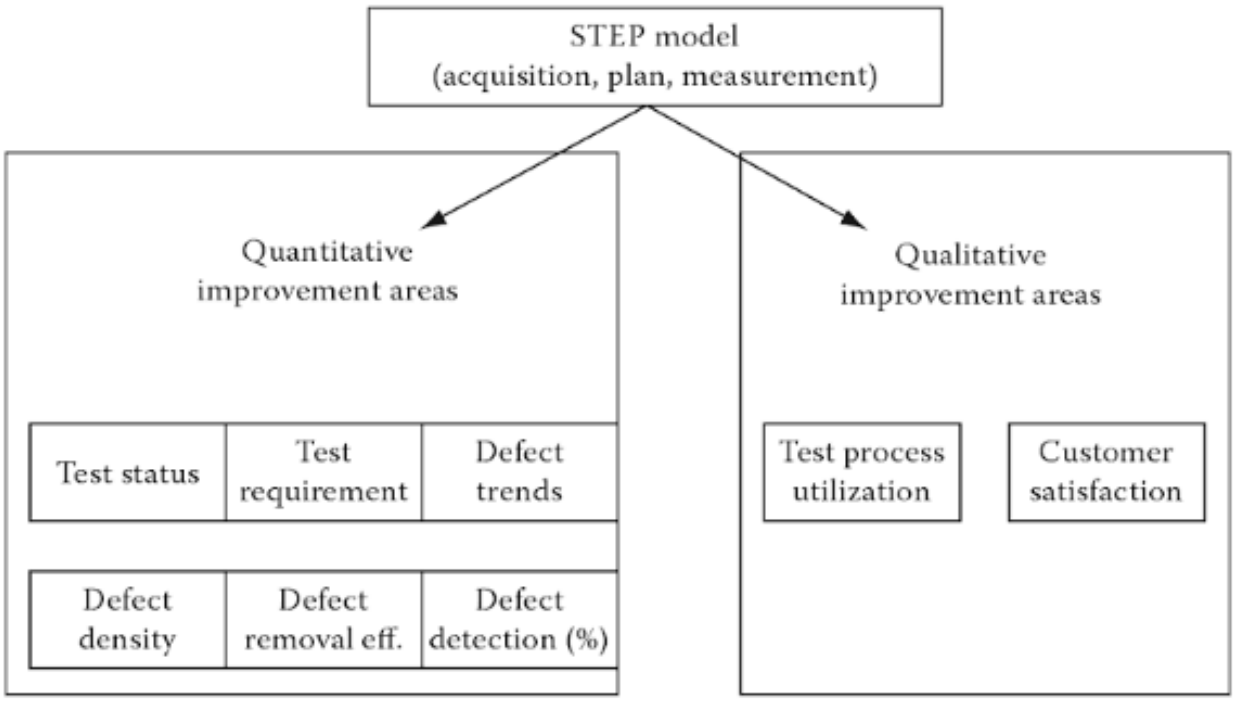

Figure 9