What We Know about Software Test Maturity and Test Process Improvement

Vahid Garousi, Hacettepe University

Michael Felderer, University of Innsbruck

Tuna Hacaloğlu, Atilim University

// A review of the scientific literature and practitioners' gray literature identified 58 test maturity models and many sources with varying degrees of empirical evidence. Using this knowledge, researchers and practitioners should be able to assess and improve test process maturity. //

THE EFFICIENCY AND effectiveness of software testing practices vary among companies and software teams. Some companies conduct efficient, effective software development and testing that produce high-quality software. Unfortunately, however, many companies' testing practices are far from mature and are usually conducted in an ad hoc fashion.[1–3] Such immature practices lead to negative outcomes—for example, testing that doesn't detect all the defects or that incurs cost and schedule overruns.

To determine the efficiency, effectiveness, and quality of testing practices, companies and software teams often perform test maturity assessment (TMA).[4] As a follow-up, test engineers and managers often perform test process improvement (TPI). To conduct TMA and TPI systematically, researchers and practitioners have proposed various approaches and frameworks, such as the approaches described in the recent book *Improving the Test Process: Implementing Improvement and Change*.[4] This book forms the basis for the International Software Testing Qualifications Board (ISTQB) expert-level certification on TPI.

In collaborations with practitioner testers and in the context of several TPI projects in which we've been involved, we've come to realize that testers or managers interested in conducting TMA and TPI face several challenges. These challenges include

- raising the need for TMA and TPI among team members and in the company,
- proper planning of TMA and TPI activities before actually starting them,
- identifying the challenges beforehand and being ready to address them,
- systematically measuring the benefits of TMA and TPI, and
- assessing the success of TMA and TPI activities.

0740-7459/18/$33.00 © 2018 IEEE

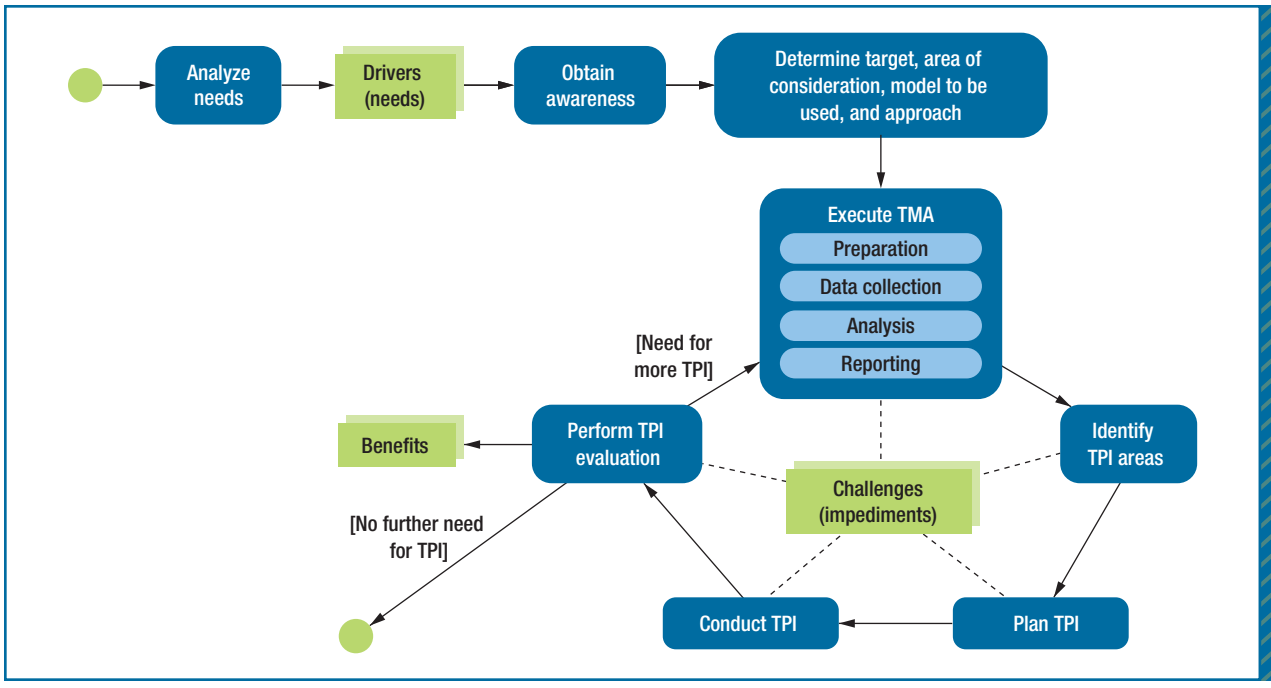

FIGURE 1. A general process for test maturity assessment (TMA) and test process improvement (TPI). This process was inspired by a simpler version in *Test Process Improvement: A Practical Step-by-Step Guide to Structured Testing*.[12]

To help software engineers meet these challenges, to identify the state of the art and the practice in this area, and to find out what the software engineering community knows about TMA and TPI, we conducted a *multivocal literature review* (MLR). An MLR is a type of systematic literature review (SLR) that includes data from multiple types of sources—for example, scientific literature and practitioners' gray literature (such as blog posts, white papers, and presentation videos).[5,6] MLRs related to software engineering have recently started appearing; one example is an MLR on technical debt.[7] They're especially suitable for investigating TMA and TPI, which are equally driven by and relevant to both industry and academia. Our review identified many test maturity models and many sources with varying degrees of empirical evidence.

Papers like this article have been published on other topics—for example, agile development[8] and developer motivation[9]—and have provided concise overviews of a given area. Some review papers on TMA and TPI exist,[10,11] but none of them considered both the academic literature and practitioners' gray literature. Also, none of them studied the topic in as much depth as our review.

A General Process for TMA and TPI

Figure 1 depicts the general process for TMA and TPI as a UML activity diagram. This process was inspired by a simpler version in *Test Process Improvement: A Practical Step-by-Step Guide to Structured Testing*;[12] we extended that process using our review's findings.

Usually, a TMA-and-TPI initiative starts with a needs analysis; that is, a test engineer or testing team determines whether such an initiative is necessary. The next step is to promote awareness among other stakeholders and management. Then, the engineer or team determines the areas of consideration, the TMA model and TPI model to use, and the suitable approach. After that, the actual TMA starts, which identifies the TPI areas.

Next, the engineer or team plans and conducts TPI and evaluates its outcomes and benefits. If more TPI is necessary, the process repeats; if not, it finishes. As you can see, choosing the right models and assessing the drivers, challenges, and benefits play a major role in this context.

The Review Procedure

Our MLR followed the standard process for SLRs in software engineering. We aimed to address these questions:

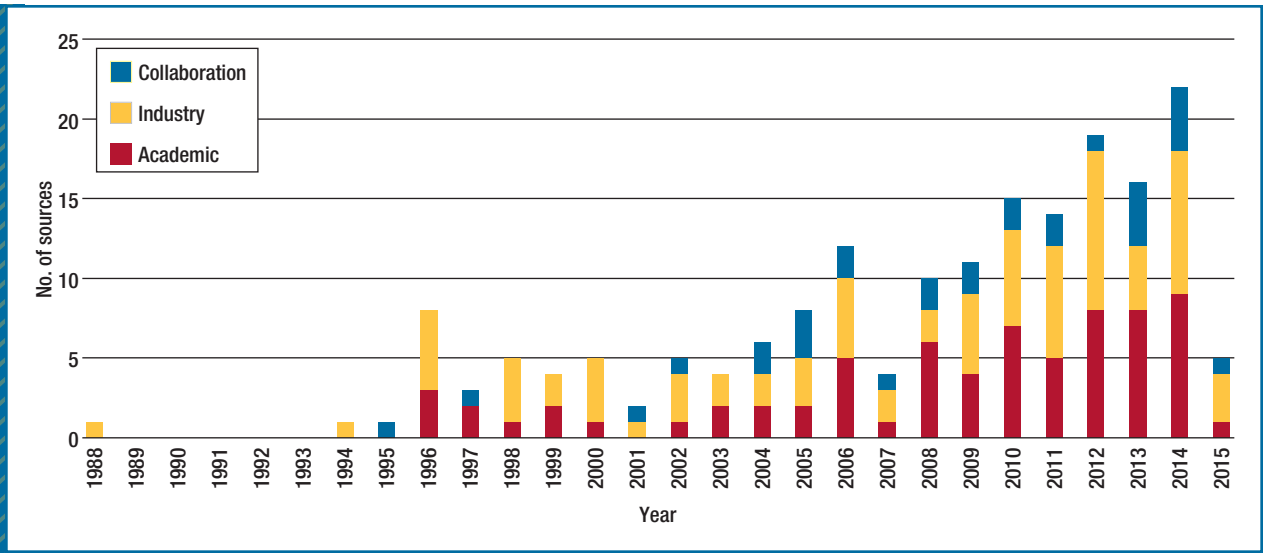

FIGURE 2. Growth of the TMA and TPI field and types of authors. Attention to this topic from both researchers and practitioners has steadily risen since the early 1990s. The pool of sources for 2015 is partial because we selected them in June 2015.

- What test maturity models have researchers and practitioners proposed?
- What are the drivers for TMA and TPI?
- What are the challenges of TMA and TPI?
- What are the benefits of TMA and TPI?

We performed the searches in the Google and Google Scholar databases. Our search strings were

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

To synthesize the opinions and empirical evidence in the primary sources regarding the drivers, challenges, and benefits, we used qualitative coding (also called grounded theory). A more detailed description of our MLR process and qualitative coding is at goo.gl/pNCKpn. In that document, we also discuss how we identified and addressed the potential threats to our review's validity.

Researcher and Practitioner Involvement

After we voted and applied inclusion or exclusion criteria, we were left with 181 sources, of which 130 were formally published and 51 were gray literature. The final pool of sources and a mapping repository are at goo.gl/lG4LqF. Throughout the rest of this article, we indicate those sources using "S" and a number—for example, S49.

Figure 2 plots the number of studies published by academic researchers only, by practitioners only, or as collaborations between the two. As you can see, attention to this topic from both researchers and practitioners has steadily risen since the early 1990s. The pool of sources for 2015 is partial (only five sources) because we selected them in June 2015.

The Test Maturity Models

Of the 181 sources, 58 presented new test maturity models and 117 used existing models. The remaining six presented other types of research contributions; for example, S49 proposed test metrics for test strategy evaluation. We were surprised to see so many test maturity models. We don't have enough space here to list them all. Table 1 presents nine representative examples; three are generic, three are for specific software development types, and three are for specific purposes.

In our review, the most popular models were Test Maturity Model Integration (TMMi) [S127] and its earlier version, the Testing Maturity Model (TMM) [S44], and TPI [S74] and its successor, TPI Next [S80]. Fifty-seven sources used TMMi and TMM for assessments or base models, whereas 18 used TPI and TPI Next. Twenty-eight sources used other models—for example, TestSPICE [S93] and the Test Management Approach (TMap) [S157].

Table 1. Nine test maturity models.

Category	Model*	Staged or continuous?
Generic	Test Maturity Model Integration (TMMi) [S127]	Staged: Level 1. Initial Level 2. Managed Level 3. Defined Level 4. Measured Level 5. Optimization
	Test process improvement (TPI) [S74]	Continuous. Includes 20 Key Performance Areas (KPAs), each with levels A through D: 1. Test strategy 2. Lifecycle model 3. Moment of involvement 4. Estimating and planning … 18. Test process management 19. Evaluation 20. Low-level testing
	TestSPICE [S93]	Continuous. Comprises a set of KPAs based on ISO/IEC 15504, the *S*oftware *P*rocess *I*mprovement and *C*apability D*e*termination (SPICE) standard.
Targeted for specific development types or domains	Agile Quality Assurance Model (AQAM) [S3]	Staged: Level 1. Initial Level 2. Performed Level 3. Managed Level 4. Optimized
	Agile Testing Maturity Model (ATMM) [S35]	Staged: Level 0. Waterfall Level 1. Forming Level 2. Agile Bonding Level 3. Performing Level 4. Scaling
	TPI for Embedded Software and Industrial Characteristics (TPI-EI) [S24]	Continuous. An adaptation of TPI for embedded software.
Targeted for specific test activities	Unit Test Maturity Model (UTMM) [S156]	Staged: Level 0. Ignorance Level 1. Few Simple Tests Level 2. Mocks and Stubs Level 3. Design for Testability Level 4. Test-Driven Development Level 5. Code Coverage Level 6. Unit Tests in the Build Level 7. Code Coverage Feedback Loop Level 8. Automated Builds and Tasks
	Automated Software Testing Maturity Model (ASTMM) [S5]	Staged: Level 1. Accidental Automation Level 2. Beginning Automation Level 3. Intentional Automation Level 4. Advanced Automation
	Personal Test Maturity Matrix (PTMM) [S151]	Continuous. Comprises a set of KPAs such as test execution, automated test support, and reviewing.

** An "S" with a number in brackets indicates one of the sources in our review pool, which is available at goo.gl/IG4LqF.*

We observed the development of models such as TPI for Embedded Software and Industrial Characteristics (TPI-EI) [S24], the Unit Test Maturity Model (UTMM) [S156], or the Personal Test Maturity Matrix (PTMM) [S151], which is used to gauge test engineers' (knowledge or skill) maturity and capability development. After reviewing several models' technical details, we determined that many aspects in various models clearly overlap.

Like the Capability Maturity Model Integration (CMMI) program,[13] the testing maturity models, broadly speaking, fall into two types: staged or continuous (see the last column of Table 1). In staged models, such as TMMi, the Agile Quality Assurance Model (AQAM) [S3], and the Automated Software Testing Maturity Model (ASTMM) [S5], testing maturity is assigned a level on the basis of a set of specific goals and practices. In continuous models, such as TPI, TestSPICE, and PTMM, a set of individual Key Process Areas (KPAs) are assessed according to a set of defined criteria.

What's evident from the set of 58 models is that no one model fits all test improvement needs. One possible reason for the creation of a subset of the models originating from academia is that the original models weren't based on industrial needs but sometimes on hypothetically argued motivations. Also, it seems that some researchers often didn't fully review the state of the art or the practice to minimize overlap and to take into account best practices from research and industry.

Figure 3 shows a chronological evolution graph of TMA and TPI models and their relationships—that is, how models have been based on earlier models. This figure was inspired by a similar evolution model prepared for UML.[14] As you can see, new (original or extended) TMA and TPI models have been proposed regularly since 1985. Many of the new models are based on older models; for example, the Metrics Based Testing Maturity Model (MB-TMM) [S48], proposed in 2001, is based on TMM.

With such a large collection of models and the overlap among them, choosing the most suitable model or models to apply isn't easy.[15,16] Further complicating things, many practitioners and researchers have reported challenges when using even established models such as TMMi[15]—for example, not being able to objectively assess each maturity area or item using the existing model guidelines.

Drivers

After careful data extraction, logging of drivers phrased in different forms and terminologies, and qualitative coding of the drivers as reported in the sources, we synthesized and classified drivers into the following categories.

Process and operational drivers (mentioned by 46 sources) included

- a lack of focus in test activities and people-dependent performance [S23];
- low test efficiency [S56];
- testing practices not meeting expectations or commitments [S58];
- internal stakeholder dissatisfaction with existing testing practices [S58];
- missing exit criteria for testing [S70];
- the need to improve the productivity of testing [S73]; and
- the need to determine a baseline for test capabilities, develop a credible testing roadmap, and raise the profile of testing [S159].

Software quality drivers (25 sources) included

- a high number of faults due to low testing quality (effectiveness) [S4];
- a direct relationship between test process quality and the developed product's final quality [S29]; and
- a lack of planning and resources for testing, which impacted software quality [S40].

Cost-related drivers (23 sources) included

- the argument that most current testing processes were often technique-centered rather than organized to maximize business value [S78],
- excessive testing costs [S177], and
- testing that wasn't cost-effective [S181].

Schedule-related drivers (12 sources) included

- production delays due to ineffective testing [S4],
- the need to accelerate time to market by effective testing [S25], and
- a test team that spent a lot of time on manual testing [S28].

Finally, 15 sources cited various other drivers.

Many sources reported that a main step in starting (and successfully performing) TMA and TPI was to get (and keep) stakeholders' commitment. To establish

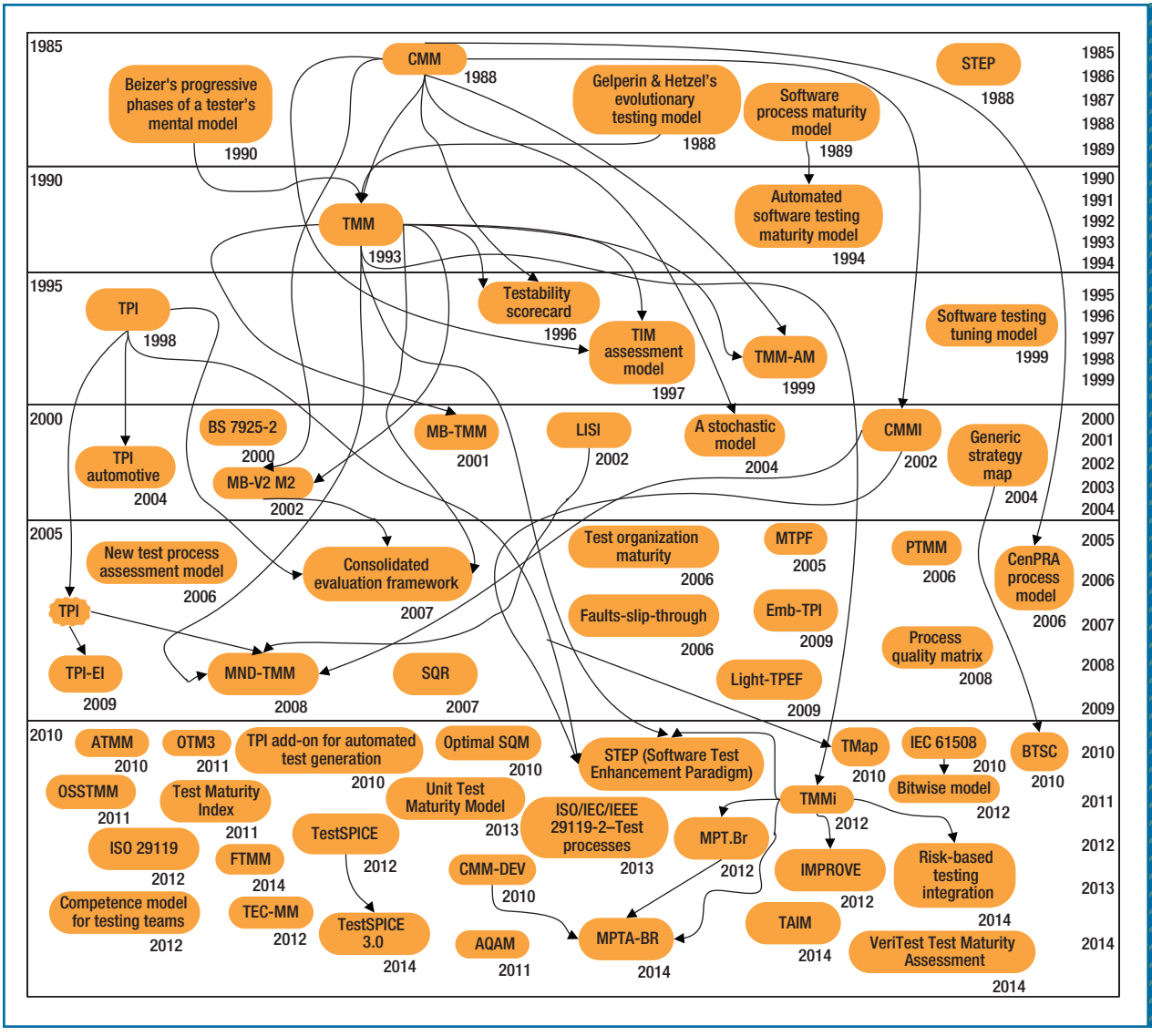

FIGURE 3. The evolution of TMA and TPI models and their relationships. New (original or extended) models have been proposed regularly since 1985.

commitment, cost–benefit analysis (quantitative and qualitative) of TMA and TPI activities is important. In this context, costs are related to the effort spent on the activities, and the benefits are related to satisfying the drivers. Only if the expected benefits outweigh the costs will TMA and TPI get the green light to be conducted.

Challenges

Any improvement activity presents challenges. We classified the challenges into the following categories.

Seventeen sources mentioned a lack of (required) resources. For example, S23 reported that the lack of a process improvement infrastructure was a major barrier to TPI at Union Switch and Signal, a supplier of railway signaling equipment in the US.

Twelve sources mentioned resistance to change. For example, S155 recommended tailoring TMA activities to meet an organization's cultural norms, to avoid resistance. S89 focused on the personal psychology of testers and reported that minimizing the fear factor in applying

TPI put testers through fewer emotional swings.

Nine sources mentioned that improvement felt like an additional effort. For example, S123 reported this situation in activities dedicated to diagnosing the current testing practices in relation to TPI activities.

Seven sources mentioned a lack of competencies. For example, S123 considered the lack of available human and economic resources an important challenge for small and medium organizations to conduct TMA and TPI. To deal with the lack of competencies, S113 recommended training testers to conduct TPI.

Seven sources mentioned an unclear scope and focus. For example, S62 reported that a major challenge was to prioritize the areas to improve. Without such decision support, organizations often didn't implement improvements because they had difficulty prioritizing them.

Five sources mentioned that there was no owner for the improvement.

In four sources, the study subjects saw no clear benefits from such improvement activities. For example, in S102, a team of Brazilian practitioners and researchers reported that small companies aiming to implement TPI models sometimes aborted this undertaking. This might have been because the models didn't show benefits or the company wasn't ready for the maturity improvement. S123 stated that estimating TPI activities' expected return on investment was often difficult. Moreover, such estimations usually had low reliability.

Finally, 23 sources mentioned various other challenges.

Benefits

The successful implementation of TMA and TPI depends heavily on the expected or actual benefits. We classified the following categories of benefits.

Operational benefits (mentioned by 48 sources) included

- shorter development time [S121];
- lower development costs [S121];
- better planning of testing costs [S71];
- alignment of internal testing processes with external value objectives [S78];
- better adherence to release dates [S79];
- reduced failure administration [S59];
- minimized test cycle time [S96];
- improved risk identification and management [S146];
- development of adequate training for test personnel [S43]; and
- process control based on metrics, resulting in more accurate estimations and predictions [S170].

Technical benefits (37 sources) included

- fewer field defects, resulting in better software quality [S79];
- fewer high-severity defects [S180];
- increased traceability to support release decisions [S88];
- improved test automation [S117]; and
- improved test-case design by adopting new techniques [S133].

Business benefits (27 sources) included

- increased profit [S78],
- increased customer satisfaction [S78],
- a positive return on investment [S55 and S92],
- reduced cost of test tasks [S66],
- reduced defect costs [S112],
- better internal and external reputation [S146],
- increased business opportunities [S146], and
- reduced support costs [S180].

An Industrial Case Study

We studied a Turkish software firm (one of Vahid Garousi's industry partners) that was interested in increasing its test practices' maturity. Our MLR helped us plan and conduct the TMA and TPI project more rigorously and systematically. We performed a TMMi assessment, using *Test Maturity Model Integration (TMMi), Release 1.0*[17] and *TMMi Assessment Method Application Requirements (TAMAR), Release 1.0*.[18]

TMMi has five maturity levels (see Table 1). Each level has several process areas (PAs). Each PA has several specific goals (SGs) and specific practices (SPs). For example, level 2 (Managed) has five PAs—for instance, PA 2.1 (test policy and strategy). This PA has three SGs: SG 1 (establish a test policy), SG 2 (establish a test strategy), and SG 3 (establish test performance indicators). SG 1, in turn, has three SPs: SP 1.1 (define test goals), SP 1.2 (define a test policy), and SP 1.3 (distribute the test policy to stakeholders). There are 50 SGs and 188 SPs.

To rate each SP, we used the following scale, suggested by the TAMAR document:

- fully implemented (FI),
- largely implemented (LI),
- partially implemented (PI),
- not implemented (NI), and
- not applicable (N/A),

This scale is similar to the five levels of the Standard CMMI Appraisal

Method for Process Improvement (SCAMPI):

- fully implemented (FI),
- largely implemented (LI),
- partially implemented (PI),
- not implemented (NI), and
- not yet (NY).

After conducting the TMMi assessment, to systematically evaluate test maturity and compile the areas for improvement, we reviewed and collected the SPs that ranked lower than FI. Here are several suggestions that were made in relation to those SPs to the test managers in the software firm under study:

- Separation of debugging from testing could be clearer in test policy documents.
- It would be a good idea to document generic product risks in potential-risks documents and to conduct systematic risk-based testing.
- Test policy and test performance indicators and metrics could be updated.

Thanks to the MLR, we knew the potential drivers, challenges, and benefits before the project started and could observe or tackle several of them throughout our activities. Many of the team members in the company felt that the MLR helped them to be prepared for the project. As a result, the company has planned and is conducting TPI activities.

The review results we presented here should help both researchers and practitioners assess and improve the maturity of test processes. The issue of choosing the right maturity models has been explored in other areas (for example, business process maturity assessment[19]) but needs further investigation for the set of test maturity models. Practitioners new to this area who intend to conduct TMA and TPI should know the differences and similarities of the models and their success or failure. The need exists for domain analysis of the models and in-depth examination of the extent to which they are similar and tend to become unified.

Also, another important issue is assessing the models' "fit for purpose." That is, to what extent do they really help test teams assess and improve their test processes? This issue has also been investigated in the domain of business process maturity assessment.[20]

Although there has been considerable interest and progress in TMA and TPI, the need exists for more empirical studies providing evidence for TMA and TPI in specific contexts—for example, by taking into account the domains of the systems under test. We also need more evidence-based approaches.

References

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

ABOUT THE AUTHORS

VAHID GAROUSI is an associate professor of software engineering at Wageningen University. He previously was at Hacettepe University. His research interests include software test engineering, software quality, empirical software engineering, and improving industry–academia collaboration in software engineering. Garousi received a PhD in software engineering from Carleton University. He was a Distinguished Visitor in the IEEE Computer Society's Distinguished Visitors Program from 2012 to 2015. Contact him at vahid.garousi@wur.nl.

MICHAEL FELDERER is an associate professor in the University of Innsbruck's Department of Computer Science and a guest researcher in the Blekinge Institute of Technology's Department of Software Engineering. His research interests include software and security testing, software processes, requirements engineering, empirical software engineering, and improving industry–academia collaboration. Felderer received a habilitation in computer science from the University of Innsbruck. Contact him at michael.felderer@uibk.ac.at.

TUNA HACALOĞLU is a PhD student in the Department of Information Systems at Middle East Technical University's Informatics Institute. She's also an instructor in Atilim University's Department of Information Systems Engineering. Her research interests include information systems, software engineering, software engineering education, and software testing. Hacaloğlu received a master's in information systems from Middle East Technical University. Contact her at tuna.hacaloglu@metu.edu.tr.

Methods and Notation," 2012; en.wikipedia.org/wiki/Unified_Modeling_Language#/media/File:OO_Modeling_languages_history.jpg.

15. K. Rungi and R. Matulevičius, "Empirical Analysis of the Test Maturity Model Integration (TMMi)," *Information and Software Technologies*, T. Skersys, R. Butleris, and R. Butkiene, eds., Springer, 2013, pp. 376–391.
16. M. Felderer and R. Ramler, "Integrating Risk-Based Testing in Industrial Test Processes," *Software Quality J.*, vol. 22, no. 3, 2014, pp. 543–575.
17. *Test Maturity Model Integration (TMMi), Release 1.0*, TMMi Foundation, 2012; www.tmmi.org/wp-content/uploads/2016/09/TMMi.Framework.pdf.
18. *TMMi Assessment Method Application Requirements (TAMAR), Release 1.0*, TMMi Foundation, 2014; www.tmmi.org/wp-content/uploads/2016/09/TMMi.TAMAR_.pdf.
19. A. Van Looy et al., "Choosing the Right Business Process Maturity Model," *Information & Management*, vol. 50, no. 7, 2013, pp. 466–488.
20. A. Van Looy, "Looking for a Fit for Purpose: Business Process Maturity Models from a User's Perspective," *Proc. IFIP Working Conf. Enterprise Information Systems of the Future*, 2013, pp. 182–189.