The Process of Test Process Improvement

Jef Jacobs, Jan van Moll, and Tom Stokes

Software testing is still a pain-in-the-neck for many organisations. Because it is only marginally addressed in software process improvement models like CMM, a separate Testing Process Improvement model is needed. The current authors have implemented a structured testing process guided by the "Testing Maturity Model" (TMM). An outline of this model is presented, showing how with growing maturity, testing evolves from detecting defects in software code to testing as essential product quality control instrument. The biggest strengths of TMM are: it reflects 40 years of industry-wide best test practices and it is designed as a counterpart of the popular CMM model for software development improvement. Weaknesses include the under-representation of test people and organisation related issues, and missing maturity goals for the test infrastructure. Based on practical implementation guided by TMM, the process of test process improvement is addressed and experiences are presented.

Introduction

Software systems are becoming increasingly important in modern society and are rapidly growing in size and complexity. Forced by competition and the tendency to shorter commercial life cycles (especially in the consumer product market) the quality of the products must be higher and higher. Apart from modern software specification, design and implementation techniques, the introduction of a sound software testing process is vital to assure proper product quality.

Though software testing has existed as long as software development, it has been a neglected area for a long time. It is widely recognised that 1979 was the turning point: the publication of Glenford Myers' book "The Art of Software Testing" [1] raised the awareness that software testing is a discipline in its own right. Myers described various testing techniques, proposed a systematic test approach and he advocated a then revolutionary idea: the separation of testing from development. It took the software development community several years to digest Myers' ideas, but in the second half of the 1980's the effects became visible in the USA. It was not before the 1990's that the effects became visible in Europe.

Software testing is coming of age. A wide catalogue of excellent books on the subject exists, specialised journals are available, focused conferences, seminars and workshops are held, special interest groups are in place, news groups flourish, training services are offered and a certification program exists.

In spite of the vital role of testing in software development, existing software development maturity models, like CMM[2], have not adequately addressed testing issues nor has the nature of a mature testing process been well defined. For example, in the Capability Maturity Model, CMM [7]:

- The concept of testing maturity is not addressed
- There is no adequate inclusion of testing prac-

[2] CMM, registered service marks of Carnegie Mellon University.

tices as a process improvement mechanism

- Testing issues are not addressed in the key process areas
- Product-quality related issues are not satisfactorily addressed.

Many organisations struggle with the foundation of a sound software testing process. What is a sound testing process in the first place? How should you organise and implement test process improvement? How should you embed it into the organisation? What are the consequences of it? In short, guidance for the process of test process improvement is badly needed, as well as a method to measure the maturity level of testing, analogous to, let's say, SEI's Capability Maturity Model (CMM) for the software development process.

Some of the best known models for test process improvement are TIM (Test Improvement Model), TOM (Test Organisation Maturity model), TPI[3] (Test Process Improvement model), and the most recent addition, TMM[4] (Testing Maturity Model). Each of these models, of course, have there own characteristics, strengths, weaknesses and merits.

The authors have been involved in a Test Process Improvement programme in an industrial environment, guided by TMM. The remainder of this paper focuses on TMM and its usage.

The Testing Maturity Model (TMM)

TMM (Testing Maturity Model) was developed, in 1996, at the Illinois Institute of Technology [2,3]. It reflects the evolutionary pattern of testing process maturity growth documented over the last several decades. The basis for it was the historical model provided by Gelperin and Hetzel [5]. Their model describes phases and test goals for the periods of the 1950's through the late 1980's. Basically four periods can be distinguished: The "debugging-oriented" period, where testing was merely seen as an activity to help remove bugs, the "destruction-oriented" period focused on testing as an activity to detect implementation faults, the "evaluation-oriented" period in which testing became an activity that was integrated into the software life cycle with the purpose to detect requirements, design and implementation faults. Finally, the "prevention-oriented" stage where the scope of testing is broadly defined and includes review activities, with the primary goal to prevent requirement, design and implementation faults. The basic idea behind TMM is that every organisation goes through these historical phases, and that by providing the characteristics of these phases the test maturity can be determined. Thus in essence, TMM is an assessment model rather than an improvement model. But an assessment model can be used as a basis for an improvement programme as well.

TMM has two major components: the Maturity Model, in which five maturity levels are distinguished (like in CMM), and an Assessment Model. Each maturity level, with the exception of the initial level 1, has a structure consisting of: A set of maturity goals, identifying testing improvement goals that must be addressed to achieve maturity at that level (consider these as the Key Process Areas) Supporting subgoals, defining the scope, boundaries and needed accomplishments for a particular level necessary to achieve the goals associated with each level.

The model with its maturity levels and goals is depicted in Figure 1.

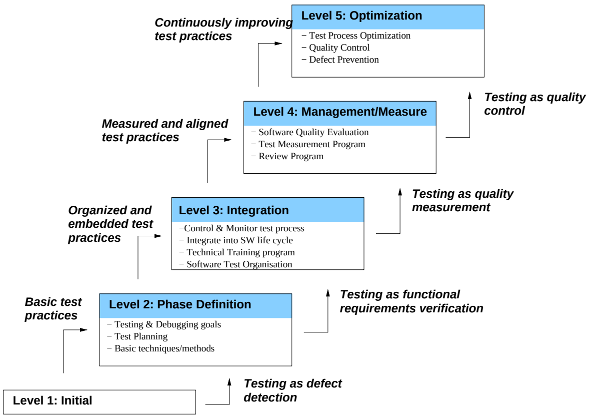
Continuously improving test practices
Level 5: Optimization
− Test Process Optimization
− Quality Control
− Defect Prevention
Testing as quality control
Measured and aligned test practices
Level 4: Management/Measure
− Software Quality Evaluation
− Test Measurement Program
− Review Program
Testing as quality measurement
Organized and embedded test practices
Level 3: Integration
−Control & Monitor test process
− Integrate into SW life cycle
− Technical Training program
− Software Test Organisation
Basic test practices
Level 2: Phase Definition
− Testing & Debugging goals
− Test Planning
− Basic techniques/methods
Testing as functional requirements verification
Level 1: Initial
Testing as defect detection

Figure 1: Maturity levels and goals of the Testing Maturity Model TMM.

Note that the layout of the model is very similar to CMM, and indeed, it was deliberately designed

[3] TPI, registered by Iquip Informatica B.V.

[4] TMM, registered service marks of Illinois Institute of Technology.

to be similar. The idea behind it is that growth in testing maturity should go hand-in-hand with growth in software capability maturity. Test maturity growth alone will become hindered by a lagging software development maturity, and shall eventually be blocked by it.

The TMM level 1, the Initial level, is characterised by a chaotic testing process. Tests are developed in an ad hoc way after coding is done. Testing and debugging are interleaved. The objective of testing is to show that the software works.

The TMM level 2, Phase Definition, is characterised by a separation of testing and debugging. Testing still follows coding but is a planned activity. The primary goal of testing at this maturity level is to show that the software meets its specifications. Post-code execution based testing is still considered the primary testing activity.

The TMM level 3, Integration, assumes that testing is no longer a phase after coding; it is integrated into the entire software life cycle. Test objectives are established with respect to the requirements based on user and client needs and are used for test case design and success criteria. There is a test organisation and testing is recognised as a professional activity, including an associated training programme.

TMM level 4, Management and Measurement, considers testing as a measured and quantified process. Reviews at all phases of the development process are now recognised as testing and quality control activities. Testware is conserved for reuse, defects are adequately logged and deficiencies in the test process are now often due to the lack of a defect prevention philosophy.

At TMM level 5, Optimization, Defect Prevention and Quality Control, the testing process is now said to be defined and managed, its costs and effectiveness can be monitored. There are mechanisms put in place to fine-tune and continuously improve testing. Defect prevention and quality control are practised. The testing process is driven by statistical sampling, measurements of confidence levels, trustworthiness and reliability.

Strengths and Weaknesses of TMM

Like any model, TMM has its strengths and weaknesses. However, the benefits of the strengths must be valued higher than the penalties of the weaknesses. Alternatively, the weaknesses of the model must be rectified in some way or another.

Definitely a strength of TMM is that it is founded on 40 years of industrial experience with software testing. It benefits from many past struggles to find a sound software testing process.

Also a very strong point of TMM is its design objective: being a counterpart of the popular software process improvement model CMM. Software process improvement programs can use TMM to complement CMM, as CMM does not adequately address software testing issues. On the other hand, it is also possible to improve the testing process independently, though one should be aware that maturity levels for testing and software development should remain close to each other.

TMM is a highly conceptual model. As such it fits every business environment. It leaves much room for business characteristics and its testing process. Though this is attractive, it has the downside that TMM is not a cook-book for a testing process. It needs the hands and brains of an experienced test process improvement leader to implement an effective, efficient and managed test process. However, the same can be said of any improvement model.

One of the biggest weaknesses of TMM is its rather poor description. Just compare the brief journal-like style of the TMM description with the extensive SEI's improvement model descriptions. TMM's cursory description causes a number of related weaknesses like lack of detail and insufficient explanation of terms.

Another weakness is the relative under-representation of goals or activities for people management and the test organisation. At TMM level 3 the goals "Establish a software test organisation" and "establish a technical training program" are indicated, but this is rather meagre. At other maturity levels, TMM addresses the people and organisation issues only casually. The development of a maturing test process implies the development of a maturing test organisation.

At every TMM maturity level, the people/organisational issues should be represented with goals and associated activities appropriate for that level. This could be rectified in several ways. Like TMM describes its relations with the CMM software improvement model, it could also refer to the People-CMM improvement model. A better idea might be to blend goals and activities from the People-CMM into the TMM model and tailor them towards a testing organisation. This is particularly a good idea, because many development-oriented organisations are just beginning with test process improvement and lack a historical frame of reference concerning test people management and the establishment of a testing competency.

Also missing in TMM is explicit attention for test infrastructure. Test infrastructure refers to test equipment, test systems, test beds, etcetera. Technical software environments often require special test systems or equipment which is quite often used by developers as well. A maturing testing process also requires a maturing management of the test infrastructure. TMM mentions test tools only in reference to tools like test coverage tools, capture & replay tools, test management tools and the like. Test infrastructure should be controlled, managed, updated, allocated and scheduled. The test system is paramount in testing and must therefore be addressed in any test process improvement model.

The Process of Test Process Improvement

Though an improvement model like TMM can help identify a good and adequate test process, it offers no guidance on how to achieve these goals. Here we give a few suggestions how one could approach it.

Initiation

The first step in the process of test process improvement is the initiation of it. The results of this first step are crucial for the remainder of the improvement programme. In essence these questions must be answered:

- How should the test process fit in the business goals?
- Does real management commitment exist?

To determine how the testing process should fit in the business goals, address the organisation's business goals, quality policy, structure, culture, style of management, available expertise, current practices, current software improvement programs and other related issues. Above all, determine what management expects from the test process, on the short term as well as on the long term. Does management view testing and product quality control to be as important as development itself, or as an unavoidable step in product development that is tolerated as long as it does not jeopardise the schedules? Is the organisation quality driven or time driven? Both are perfectly legal views, but they will result in different types of test processes. The challenge is to detect the real view, because every manager will say that he is driven by quality.

The next question to address, "Does real management commitment exist?", is even harder. Real management commitment is important because good testing:

- may announce bad news (many managers only want to hear the good news),
- has a profound effect on projects (which may lead to the bad news),
- forces managers to take difficult decisions (to release or not, based on product risks),
- may be as expensive as development itself,
- may expose a weak organisation
- may expose a weak development process.

Without real management commitment, testing is deemed to a marginal existence without much effect.

Once it is known how the test process should fit in the business goals, what is expected from it, and once belief in real management commitment exists, continue with the performance of a null-assessment. Use the assessment model that comes with the selected test process improvement model. Depending on budget and resources this can either be a full-blown or a quick-scan assessment. The purpose of this is to establish a baseline. Outcomes of later assessments can be compared to those of the null-assessment to measure the progress of the test process improvement. The null-assessment also identifies attention points and their priorities, which can be used for focusing the test process improvement.

Roadmap

Now the fun stuff follows: the real test process improvement. Figure 2 below shows the process flow.

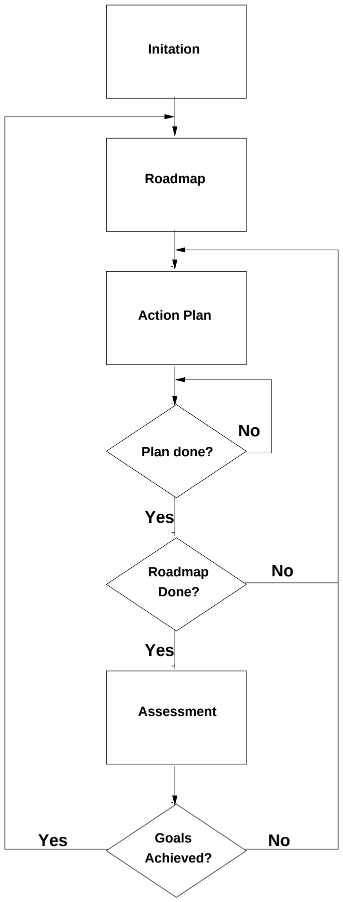

Figure 2: Process Flow of Test Process Improvement.

On the basis of the information gathered in the initiation phase, compile a roadmap. The roadmap is an overview of long term improvement actions. It identifies which goals/activities from the test improvement model are done in what order and when. It can be argued that the improvement activities must be addressed in the order given by the maturity levels of the model but, in practice, this may not be the best approach. Depending on the organisation, existing test structure, needs, etcetera, it may be better to attach higher priorities to actions where the organisation suffers the most intense pain.

The period that the roadmap covers is typically one to two years. A longer term is not really practical because ideas or priorities usually change over time. So it is better to revise the roadmap periodically.

Action Plans

The roadmap leads to action plans that cover the actions of a short term, say, a couple of months. Such an action plan is the “Project Management Plan” for specific improvement tasks. Conventional project tracking methods can be used for measuring progress. The action plan is considered to be finished when all activities described in it have been completed and the deliverables are available. This is a good moment to check whether the improvement path stipulated in the roadmap is still valid. Another action plan is then compiled, covering the next short-term period. And so on.

Assessment

Periodic assessment should be conducted to measure where you are with the test process, to see if you have achieved the goals set forth in the roadmap and to identify priorities and improvement focus points for the period to come. Assessment results should show which improvements have been accomplished and whether the roadmap should be updated.

Practical Experiences with TMM based Test Process Improvement

Using TMM as a guide and following the process as described above, the authors have implemented from 1998 on a Test Process Improvement in an industrial environment. The products concerned large and complex television, audio and data broadcast systems for satellite, cable and terrestrial operators. Before improvement, the organisation did devote attention to testing but the management judged it to be unsatisfactory. An external assessment of the test process was held which confirmed that testing was an unstructured activity that was neither effective nor efficient. It was decided to set up a Test Process Improvement programme. At the same time a CMM assessment was held to investigate if the software development capability CMM level 2 was achieved. This was not the case.

The improvement programme was initiated in 1998, according to the process outlined in the previous chapter. As an improvement model TMM was selected[5]. A null-assessment was held as a baseline for later TMM assessments. This TMM null-assessment indeed indicated that the test maturity was at the initial level 1.

A roadmap spanning the period 1999–2000 was prepared, indicating TMM level 3 as a goal by the end of 2000. Based on the roadmap action plans were developed, typically spanning 3–6 months. Action plans were periodically presented and reviewed with management.

An intermediate (self-)assessment, at the end of year 1999 indicated that TMM level 2 was achieved. A formal assessment is scheduled for the end of 2000. It is expected that all level 2 and level 3 KPA's will be satisfied except one (Control & Monitoring). The major reason for this is that the development capability still has not reached CMM level 2. In practice we experience that lagging CMM maturity becomes blocking for further test process improvement.

TMM has proven to be a valuable model for Test Process Improvement. The test process is considered to be adequate in terms of effectivity and efficiency, both by management and by developers. Test activities are performed in an orderly, structured and repeatable way, by a number of dedicated testers that consider testing to be a profession and an engineering discipline in its own rights. To obtain objectivity, the testing is organised separate and independent from software development itself. Initially, developers resisted the barrier of an independent test group. Gradually however, the developer's attitude changed from hostility to toleration to respect.

Testing has evolved in two years time from "trying software to see if it works (at first sight and to some degree)" to a true risk-based requirements verification. As anticipation to higher TMM levels test effort is also directed towards early defect detection e.g. reviews to prevent that defects in requirements documents propagate to later development stages.

Conclusions and Future Outlook

While many articles on Test Process Improvement dig immediately into technicalities of the testing process, we have tried to focus on the process of Test Process Improvement itself. This paper is an account of our experience with an improvement programme that was guided by the Testing Maturity Model (TMM) developed at the Illinois Institute of Technology. The sheer fact that we adopted TMM as an improvement model does not imply that this is the best model: it simply turned out to be the best fit for our initial situation. On the other hand, our use of TMM has exposed some weaknesses. Though we emphasise these weaknesses, we don not imply that TMM is an inadequate model. By and large, under guidance of TMM we have established a sound test process.

Altogether we were rather happy with TMM and the process we used to implement the improvements. Others have recognised the potential of TMM as well, also realising that it needs further extension. In the Netherlands, a consortium has been formed, consisting of industrial, service & consultancy and academic partners to enhance TMM, to improve the accompanying assessment model, provide it with a metrics basis and to give it wider support. The authors are currently participating in this consortium. The work of the consortium, carried out in a project subsidised by the Dutch government, eventually should lead to MB-TMM (Metrics-Based Testing Maturity Model).

References

[1] Myers, Glenford J., 1979, *The Art of Software Testing*, John Wiley & Sons Inc.

[5] As far as we are aware, this was the first time in the Netherlands that TMM was used for a Test Process Improvement Programme. Outside The Netherlands, (favourable) experiences with the application of TMM were reported at that time [6].

[2] Burnstein I., Suwanassart T., and Carlson C., 1996. *Developing a Testing Maturity Model. Part 1. Crosstalk, Journal of Defense Software Engineering*, 9, no. 8, 21–24, also available on http://www.stsc.hill.af.mil/-crosstalk/1996/aug/developi.asp

[3] Burnstein I., Suwanassart T., and Carlson C., 1996. *Developing a Testing Maturity Model. Part 2. Crosstalk, Journal of Defense Software Engineering*, 9, no. 9, 19–26, also available on http://www.stsc.hill.af.mil/-crosstalk/1996/sep/developi.asp

[4] Koomen T., Pol M., 1999, *Test Process Improvement: A Practical Step-by-Step Guide to Structured Testing*, Addison-Wesley Pub. Co.

[5] Gelperin D. and Hetzel B., 1988, *The Growth of Software Testing*, Communications of the ACM, 31, no. 6, 687–695

[6] Olsen, K. and Vinje, P.S., 1998, *Using the Testting Maturity Model in Practical Test-planning and Post-evaluation*, Conference Papers EuroStar '98, 6th European International Conference on Software Testing Analysis & Review, Munich, 345–359

[7] Paulk, M, Curtis B., Chrissis M.B. and Weber, C., 1993, *Capability Maturity Model for Software*, Software Engineering Institute, Carnegie Mellon University