TPI – a model for Test Process Improvement

Jari Andersin

Helsinki, 5th October 2004
Seminar on Quality Models for Software Engineering
Department of Computer Science
UNIVERSITY OF HELSINKI

TPI – a model for Test Process Improvement

Jari Andersin

Seminar on Quality Models for Software Engineering
Department of Computer Science
UNIVERSITY OF HELSINKI
Helsinki, 5th October 2004, 13 pages

Testing is often considered as an expensive and uncontrollable process. Testing takes too much time, costs a lot more than planned, and offers insufficient conception of the quality of the test process. Therefore, the quality of the information system and the risks for the business can be difficult to determine.

Many organisations realise that improving the test process can solve these problems. However, in practice it turns out to be hard to define what steps to take for improving and controlling the process, and in what order.

The Test Process Improvement (TPI) model has been developed based on the practical knowledge and experiences of test process development. TPI offers a viewpoint in the maturity of the test processes within the organisation. Based on this understanding the model helps to define gradual and controllable test process improvement steps.

In this paper the contents and the structure of the TPI model is introduced. Also some general aspects of the test process improvement and its challenges are discussed.

ACM Computing Classification System (CCS): D.2.4, D.2.5, K.6.3

Keywords: test process improvement, test process maturity, test maturity matrix

CONTENTS

1 Introduction

Testing is often considered as an expensive and uncontrollable process. Testing takes too much time, costs more than planned, and offers insufficient insight of the quality of the test process. Therefore, the quality of the information system and the risks for the business can be very challenging to determine.

Many organisations realise that improving the test process can solve these problems. However, in practice it turns out to be hard to define what steps to take for improving and controlling the process, and in what order.

The Test Process Improvement (TPI) model has been developed based on the practical knowledge and experiences of test process development. TPI offers a viewpoint in the maturity of the test processes within the organisation. Based on this understanding the model helps to define gradual and controllable test process improvement steps [Sog04].

The purpose of this seminar paper is to present and describe the contents and the structure of the TPI model. This seminar paper is structured as follows. Chapters 2 and 3 present an overview of testing and different aspects of test levels. In Chapter 4 some general problems concerning testing and testing processes are discussed. Chapter 5 describes different phases of test process improvement. Chapter 6 introduces the TPI model. Finally, Chapter 7 concludes the paper.

2 Purpose of testing

The testing activity in an information system development can be defined as follows [TMa04]:

Testing is a process of planning, preparing, executing and analyzing, aimed at establishing the characteristics of an information system, and demonstrating the difference between the actual status and the required status.

Test planning and preparation activities emphasize the fact that testing should not be regarded as a process that can be started when the object to be tested is delivered. A test process requires accurate planning and preparation phases before any measurement actions can be implemented [KoP99].

Testing reduces the level of uncertainty about the quality of a system. The level of testing effort depends on the risks involved in bringing this system in to operation, and on the decision of how much time and money is to be spent on reducing the level of uncertainty [KoP99].

3 Test levels

For organizing test efficiently, different test levels must be used. Each test level addresses a certain group of requirements, or functional or technical specifications. The content of this chapter is mainly based on [KoP99] and [ISE04].

3.1 Low level tests

Low-level tests involve testing the separate components of a system, e.g. programs, individually or in combination. From the beginning of system development, unit, program and module test are executed. The separation between above-mentioned concepts is depending on infrastructure and programming language used. These tests are in most cases executed by developers.

After the determination of the most elementary parts of the system to be fulfilled their technical specifications, larger parts of the system are tested in integration tests. The focus is on data throughput and the interface between programs at a system part level.

3.2 High level tests

High-level tests involve with testing whole, complete products [Kit95]. After the low-level tests have been executed and defects found corrected, a system test is to be executed to determine whether the system meets the functional and technical specifications.

After the system test is completed, a system is offered to customer for acceptance. The execution of acceptance test requires an environment that should be representative of the production environment.

The high-level tests especially should be regarded as individual processes. The past experience has shown that the importance of good test process design is greater with high-level tests than with low-level tests.

4 Problems concerning testing

This chapter points out some general problems concerning testing and the needs for test process improvement in general. The content of this chapter is mainly based on [KoP99].

4.1 Primitive testing

A primitive form of testing means an activity where testing is started shortly before a system goes into production phase and it is performed by someone who happens to be available. The activity usually stops when system goes into production or no new defects have been found recently. As a result the system is accepted with several defects still remaining which results in expensive and ongoing reworking and retesting.

4.2 Current state of affairs

Recognizing the importance of a well-managed test process is more common in many organisations nowadays. Test are planned and prepared before the execution takes place and they are based on specifications. There is some insight within the organisation into what has and what has not been tested. However, testing is still short of time, people, resources and expertise. Testing is involved late in the development cycle and often leads to expensive cycle of rework and retest. When testing stops, is still uncertain what the quality level of the system is.

4.3 New developments

To be able to face the competition in the current market, organisations must continue to shorten the time-to-market for new products. Although the development processes are going faster, there is no evidence of decreasing the number of errors made in a certain period of time. Lack of experience and increased technical complexity justify this proposition. Even if the test process seems to be reasonably satisfying for the current situation, it is obvious that this will not be the case in the future.

5 Improving the test process

5.1 The need for test process improvement

The cause of the problems mentioned in the chapter above can be traced to an uncontrolled or deficiently arranged test process. There is a need for test process improvement. According to Koomen and Pol [KoP99] the concept of test process improvement can be defined as follows:

Optimizing the quality, costs, and lead time of the test process, in relation to the total information services.

Quality means here the degree of insight given by the test process concerning the quality of the tested object. Quality of the system or program is not part of this definition. The natural result of a qualitatively improved test process is not better quality of the tested system. Testing itself does not add quality to the system. It can in fact determine the available quality and enable to others to improve the quality with this information given.

In relation to the total information services stands for that the test process is not on its own. Cheaper and more efficient testing should not be a goal in itself. It should contribute to better performance of the total information services.

The aim of an improved test process should be to detect defects as close as possible to their source to minimize correction costs, and to give information about the system quality as early as possible. All evaluation and test levels should be carefully adjusted to each other for achieving an optimized total strategy for detecting the most important defects as early as possible [KoP99].

Testing should become more professional task which requires special testing skills, with functions like test managers, method specialists, and testing engineers. The progress and quality of the whole test process should be measured, and the results should be used as input for further test process improvement [KoP99].

5.2 Test process improvement steps

Improving the test process can be compared to the improvement of any other process. In test process improvement, generally following steps are used [KoP99]:

1. **Determining target and area of consideration.** Quality characteristics of testing are determined: is the target to make testing faster, cheaper or with higher coverage? Which test processes are most in need of improvement, how long can the improvement process last, and with what effort?

2. **Determining current situation**. Strong and weak points of current situation are determined.

3. **Determining required situation.** Based on the analysis of the current situation and the improvement targets, required situation and the actions needed are determined.

4. **Implementing changes.** The suggested improvement actions are implemented according to a plan and situation checks are carried out to verify that the targets have been met.

By comparing the test process to a frame of reference the strong and weak aspects of the test process become more visible. A frame of reference can be a test methodology or a model for improving the test process.

According to Koomen and Pol [KoP99] the general software process improvement models (e.g. SPICE and CMM) offer an insufficient frame of reference for stepwise improvement of the test process. Due to a high level of abstraction, improvement of the test process is often handled as a single step.

Also, certain models specially designed for test process improvement, such as Testability Maturity Model, Test Improvement Model (TIM) and the Testing Maturity Model (TMM) do not contain sufficiently practical improvement steps, details and instructions.

6 The TPI model

In this chapter the TPI model is described. The content of this chapter is based on [KoP99], [Sog04] and [TMa04].

6.1 General description of the TPI model

A test process improvement model must observe a test process from different points of view, for example the use of test tools, test specification techniques, and reporting. In TPI model these are called *Key areas*. Every key area can be classified into *Levels of maturity.* All key areas are not equally important for the performance of the whole test process and some dependencies exist between the different key areas and levels. Therefore a *Test Maturity Matrix* is used.

To verify that the classification in to levels is done objectively, one or more *Checkpoints* are assigned to each level. A checkpoint is a requirement. If a test process passes all the checkpoints of a certain level, then the process is classified at that level.

In addition to mapping the current situation of the test process, the key areas and levels can also be used to define the required situation and intermediate steps on the way to this situation. As and extra aid the *Improvement suggestions* have been added to the model giving instructions and suggestions for reaching a certain maturity level.

The main concepts of the TPI model are visualized in Figure 1.

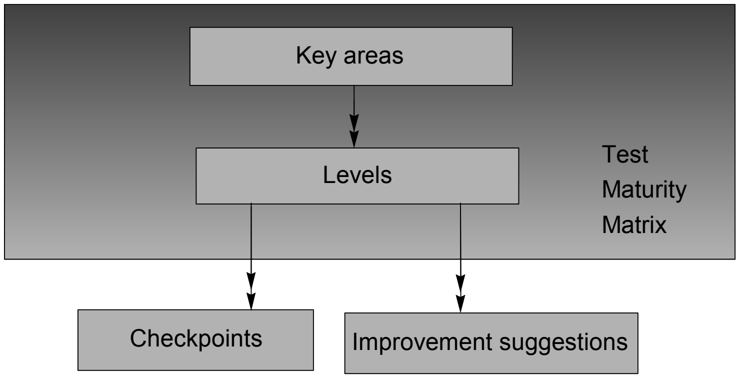

Figure 1 The TPI model [Sog04]

The TPI model has grown out of experience and it provides a frame of reference for determining the strong and weak points of the current test process and formulating specific and realistic improvement actions for this test process.

6.2 The TMap model

TMap [TMa04], the methodology for structured testing as well as several aspects from the models mentioned in chapter 5.2., are used as a basis for the TPI model. The TMap methodology has four cornerstones. These cornerstones are a *Life cycle (L)* of test activities related to the development cycle, good *Organisation (O),* the right *Infrastructure* and *tools (I),* and usable *Techniques (T)* for performing the activities.

The cornerstones are universal, and within each test process some degree of attention must be given to each cornerstone. For a balanced test process, the development of the cornerstones should be in balance.

6.3. Key areas and levels

By looking at the different aspects of each cornerstone under a structured test process, a total of 20 key areas can be recognized for the TPI model. These key areas cover the total test process.

The way key areas are organised within a test process determines the maturity of the process. However, each key area will not be addressed equally thoroughly - each test process has its strengths and weaknesses.

In order to enable insight in the state of the key areas, the model supplies them with ascending levels (generally from A to D). On the average, there are three to four levels for each key area. Each higher level is better than its prior level in terms of time (faster), money (cheaper) and/or quality (better).

Each level consists of certain requirements for the key area. The requirements (i.e. checkpoints) of a certain level also comprise the requirements of lower levels: a test process at level B fulfils the requirements of both level A and B. If a test process does not satisfy the requirements for level A, it is considered to be at the lowest and undefined level for that particular key area.

A description of the different levels of the key areas can be found in table 1.

Table 1 Different levels of key areas including cornerstones [KoP99]

Level **Key area** **Cornerstone**	A	B	C	D
Test strategy L	Strategy for single high-level test	Combined strategy for high-level tests	Combined strategy for high-level tests and low-level tests or evaluation	Combined strategy for all test and evaluation levels
Life-cycle model L	Planning, Specification, Execution	Planning, Preparation, Specification, Execution, Completion		
Moment of involvement L	Completion of test basis	Start of test basis	Start of requirements definition	Project initiation
Estimating and planning T	Substantiated estimating and planning	Statistically substantiated estimating and planning		
Test specification techniques T	Informal techniques	Formal techniques		
Static test techniques T	Inspection of test basis	Checklists		
Metrics T	Project metrics (product)	Project metrics (process)	System metrics	Organisation metrics (>1 system)
Test tools I	Planning and control tools	Execution and analysis tools	Extensive automation of the test process	
Test environment I	Managed and controlled environment	Testing in most suitable environment	Environment on call	
Office environment I	Adequate and timely office environment			
Commitment and motivation O	Assignment of budget and time	Testing integrated in project organisation	Test-engineering	

Test functions and training O	Test manager and testers	(Formal) Methodical, technical and functional support, management	Formal internal Quality Assurance	
Scope of methodology O	Project specific	Organisation generic	Organisation optimising (R&D)	
Communication O	Internal communication	Project communication (defects, change control)	Communication within the organisation about the quality of the test processes	
Reporting O	Defects	Progress (status of tests and products), activities (costs and time, milestones), defects with priorities	Risks and recommendations, substantiated with metrics	Recommendations have a Software Process Improvement character
Defect management O	Internal defect management	Extended defect management with flexible reporting facilities	Project defect management	
Testware management O	Internal testware management	External management of test basis and test object	Reusable testware	Traceability system requirements to test cases
Test process management O	Planning and execution	Planning, execution, monitoring, and adjusting	Monitoring and adjusting within organisation	
Evaluation (all)	Evaluation techniques	Evaluation strategy		
Low-level testing (all)	Low-level test lifecycle: planning, specification and execution	White-box techniques	Low-level test strategy	

For example, the test process reports weekly and contains an overview of the defects found and the hours spent. Because the defects have no indication of priority and test progress is not mentioned in the reports, the process is classified for the key area Reporting on level A.

6.4 Checkpoints

In order to determine the requirements of certain levels, the checkpoints are used. The requirements are defined in the form of questions that need to be answered positively in order to reach certain level.

Based on the checkpoints a test process can be assessed, and for each key area the proper level can be established. Every next level of a key area correspond an improvement. The checkpoints are also cumulative: in order to classify for level B the test process needs to answer positively to the checkpoints both of level B and of level A.

Example: Checkpoints for Test tools key area.

Planning and control tools (level A)
Checkpoints:

- Automated tools (other than standard word processing) are used for the defect administration and for at least two other activities of planning and control.

Execution and analysis tools (level B)
Checkpoints:

- At least two sorts of automated tools are used for test execution, such as Capture and Playback tools, test coverage tools, etc.
- The test team has a general insight into the cost/profit ratio of these tools.

Extensive automation of the test process (level C)
Checkpoints:

- Automated tools (other than standard word processing) are used for the planning phase (for the activities estimating, planning, progress monitoring, configuration management, and defect administration), preparation, specification, and execution (in total at least five sorts of tools should be used).
- The test team has an insight into the cost/profit ratio of these tools.

6.5 Test Maturity Matrix

After determining the levels for each key area, attention should be directed as to which improvement steps to take, because not all key areas and levels are equally important. For example, a good test strategy (level A of key area Test strategy) is more important than a description of the test methodology used (level A of key area Scope of methodology).

In addition to these priorities there are dependencies between the levels of different key areas. For example, before statistics can be gathered for defects found (level A of key area Metrics) the test process has to classify for level B of key area Defect management. Corresponding dependencies can be found between many levels and key areas.

Therefore, levels and key areas are related to each other in a Test Maturity Matrix. The vertical axis of the matrix indicates key areas, the horizontal axis shows scales of maturity. In the matrix each level is related to a certain scale of test maturity. This results in 13 scales of test maturity. The open cells between different levels have no meaning in themselves, but indicate that achieving a higher maturity for a key area is related to the maturity of other key areas. There is no gradation between different levels. As long as a test process is not entirely classified at level B, it remains at level A.

The structure of the Test Maturity Matrix is described in table 2.

Table 2 Test Maturity Matrix [Sog04]

Scale **Key area**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Test strategy		A					B				C		D	
Life-cycle model		A			B									
Moment of involvement			A				B				C		D	
Estimating and planning				A							B			
Test specification techniques		A		B										
Static test techniques					A		B							
Metrics						A			B			C		D
Test automation				A				B			C			
Test environment				A				B						C
Office environment				A										
Commitment and motivation		A				B						C		
Test functions and training				A			B			C				
Scope of methodology					A						B			C
Communication			A		B							C		
Reporting		A			B		C					D		
Defect management		A				B		C						
Testware management			A			B				C				D
Test process management		A		B								C		
Evaluation							A			B				
Low-level testing					A		B		C					

The scales of test maturity can generally be divided into three categories:

Controlled. Scales 1 to 5 are mainly for the control of the test process. The test process is carried out in phases according to a strategy defined in advance. Test specification techniques are used for testing, and defects are recorded and reported. The testware and test environment are well controlled and the test staff are sufficiently trained.

Efficient. The levels in scales 6 to 10 aim more on the efficiency of the test process. The efficiency can be achieved e.g. by automating the test process, by better integration between the mutual test process and with the other parties within the system development.

Optimizing. An efficient test process in the current situation may not be an efficient one in the future. The levels in scales 11 to 13 are for increasing optimization of the test process and they focus more on ensuring that continuous improvement of the test process will be part of the working methods of the organisation.

The main purpose of the matrix is to show the strong and weak points of the current test process and to support prioritising proposals and actions for improvement. The current situation of the test process can be viewed clearly. The matrix works from left to right, so low mature key areas are to be improved first.

In the example in table 3, the test process does not classify for the lowest level of the key area Test strategy (level is below A), the organisation is working according to a life-cycle model (level A) and the testers are involved at the moment when the specifications are completed (level A).

Table 3 Current situation [Sog04]

Scale **Key area**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Test strategy		A					B				C		D	
Life-cycle model		A			B									
Moment of involvement			A				B				C		D	
etc.														

Based on these levels, the improvements can be discussed. For example, a choice is made for a combined test strategy for high-level tests (to level B) and for a full life-cycle model (also to level B). Earlier involvement is not considered to be necessary at the moment. The required situation is represented in table 4.

Table 4 Required situation [Sog04]

Scale **Key area**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Test strategy		A					B				C		D	
Life-cycle model		A			B									
Moment of involvement			A				B				C		D	
etc.														

6.6 Improvement Suggestions

Improvement actions can be defined in terms of desired higher levels. For reaching a higher level the checkpoints provide much assistance. Additionally, the TPI model includes improvement suggestions for the test process improvement. These are different kinds of hints and ideas that will help to achieve a certain level of test maturity.

Example: Improvement suggestions.

Key area Test strategy, level A, Test strategy for single high level test.
Improvement suggestions:

- Involve the various interested parties such as end user, systems manager, and project manager in determining the test strategy
- Create awareness by indicating the risks of the current working method, or indicate how testing can be done cheaper and/or faster.

Unlike the use of checkpoints, the use of improvement suggestions is not obligatory. However, each level is supplied with several improvement suggestions.

7 Conclusions

Currently the software development is proceeding at a very high speed. The productivity of software development processes is rising continuously and ever higher quality is demanded by the customers. There is a strong possibility that a test process that seems to be satisfactory at the moment may need to be improved in the future.

The TPI model offers objective procedures for classifying the current situation of the test process. Additionally, the model offers assistance for test process improvement in the form of key areas, levels and improvement suggestions. Improvement is carried out using controlled improvement steps which are based on priorities.

The TPI model is based in practice and follows a structured test methodology. TPI is considered to be an objective one. By means of checkpoints it is possible to determine the levels of key areas that a test process is on. The different maturity levels and key areas and their dependencies are presented in the Test Maturity Matrix. Also, the improvement suggestions can be used for improvement actions.

However, attention should be paid on the fact that the use of the TPI model does not automatically lead to good analysis of the current and required situation and to improved test process. The model should be seen as a tool for structuring the improvement of the test process and also for better communication in the organisation. Regardless of the model used, improvement of the test process demands a high degree of knowledge and expertise of the people involved.

References

ISE04 ISEB Foundation Certificate in Software Testing course material, Grove Consultants, Kista, Sweden, 2004

Kit95 Kit, E., Software Testing in the Real World: Improving the Process. ACM Press, Essex, England, 1995

KoP99 Koomen, T. and Pol M., Test Process Improvement: A practical step-by-step guide to structured testing. ACM Press, London, England, 1999

Sog04 TPI home pages, Sogeti Nederland B.V., 2004. `http://www.sogeti.nl/index.html?/iospagina.cfm?uNr=150` [27.9.2004]

TMa04 TMap home pages, TMap - Sogeti Nederland B.V., 2004, `http://www.tmap.net` [4.10.2004]