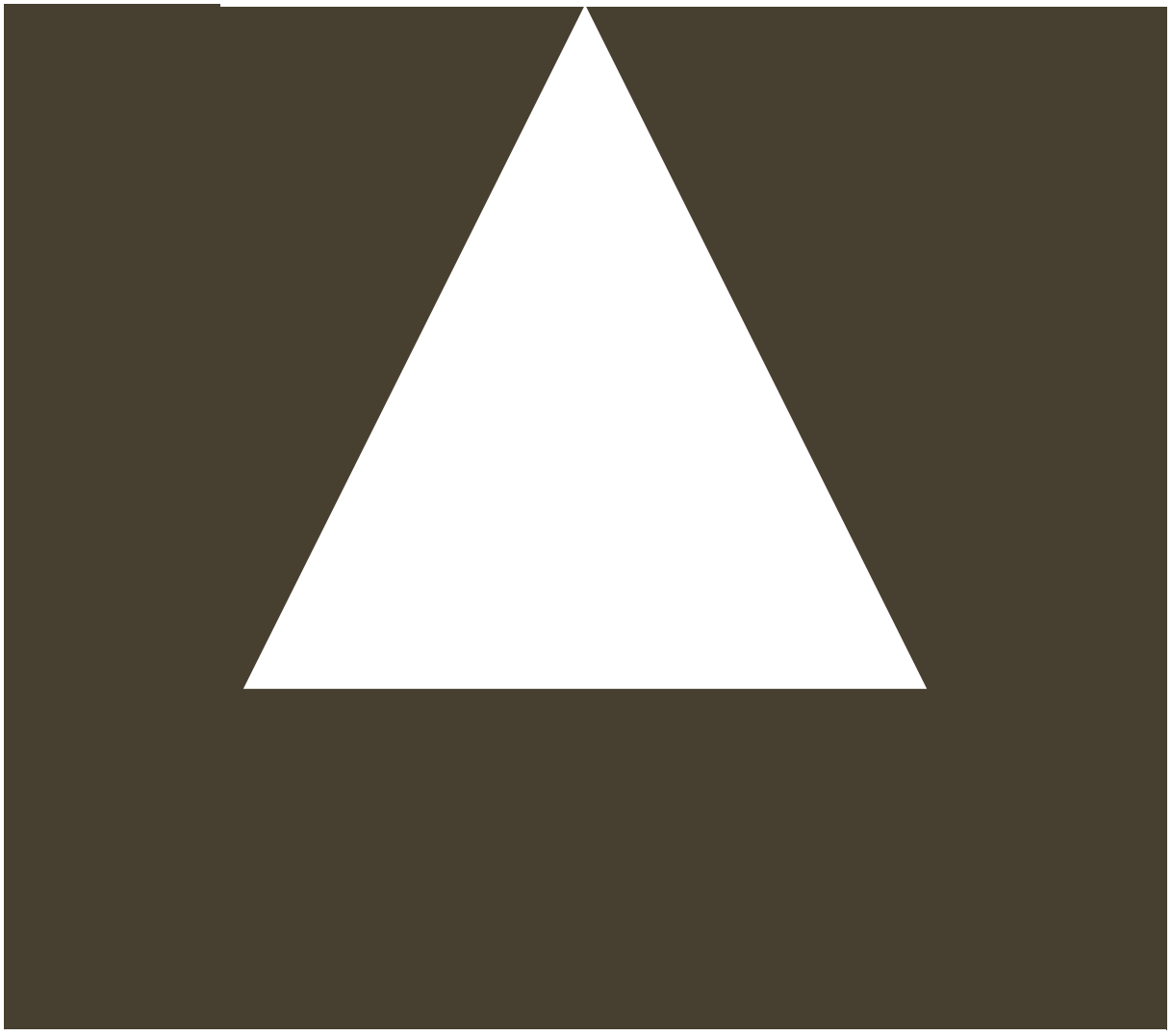
SOGETI

Agile Test Practice Improvement
AGILE TPI

Version 2.4 March 2017



Version information

Version	Date issued	Remarks	Author
0.1	September 5, 2014	Initial setup and concept	Gerrit de Vries
0.2	September 24, 2014	Added contributions from: • Yvonne van Zaanen (Triodos) • Jos Punter (Sogeti NL) • Bert Linker (Sogeti NL) • Ben Visser (Sogeti NL)	Gerrit de Vries
0.3	September 25	Added contributions from: James Redpath (Sogeti USA)	Gerrit de Vries
1.0	December 2014	Final version	Gerrit de Vries
1.1	Februari 2015	Minor adjustments	Gerrit de Vries
1.2	April 2015	Added contributions from Arne Bergs (Sogeti DE)	Gerrit de Vries
1.3	July 2015	Synchronized with the ATIM tool	Gerrit de Vries
1.4	August 2016	ATIM changed to Agile TPI	Gerrit de Vries
2.0	August 2016	Synchronized with Agile TPI Tool Minor changes Input from Eric Belt and Ben Visser	Gerrit de Vries
2.4	March 2017	Some checkpoints adapted Synchronized with Tool 2.4 Input from Eric Belt and Ben Visser	Gerrit de Vries

Summary

Agile, Scrum and TPI

Agile methodologies have emphasized many excellent practices from a testing perspective such as test-driven development and exploratory testing. Nevertheless, practice also shows that the overall approach and process of testing is often less than satisfactory in agile projects.
Like in waterfall situations there is a need for improvement of the test activities or test processes. This aspect has been elaborated in the TPI NEXT® book Business Driven Test Process Improvement[1], where Chapter 7.2 explains how the model can be used in agile situations.
However, we learned that organizations struggle with the implementation and use of agile methods. In many situations we see that it's Agile 'In Name Only', or specific elements of Scrum are adopted like the (daily) stand ups. We also learned that the professional tester is able to contribute to the agile teams with his skills and above all experience. Through his work as a tester he becomes the 'center of operations', closely working together with analysts, developers and users. From this position the tester is also capable of convincing others of the benefits of testing, thus creating a quality driven mindset with other team members and stakeholders.

Based on our test experience and our test activities in agile environments we think that this document will help organizations, project teams and individuals apply the Agile Manifesto that states:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions	**over**	processes and tools
Working software	**over**	comprehensive documentation
Customer collaboration	**over**	contract negotiation
Responding to change	**over**	following a plan

While there is value in the items on the right, we value the items on the left more!

Although 'Agile' has many different flavors we will use the Scrum terminology for this document, as Scrum is a commonly used methodology. Furthermore we refer to "TMap NEXT® in Scrum"[2], where the authors use the Scrum methodology as a basic starting point.

Developer or tester

In the ideal situation the Scrum team will consist of developers, with no specific roles or expertise: *Scrum recognizes no sub-teams in the Development Team, regardless of particular domains that need to be addressed like testing or business analysis; there are no exceptions to this rule*[3].
In the real situation however we see that the developers do have a specific role, especially the tester. With this document we do not want to challenge the Scrum Guide but, building from the practical situation, support and improve the test activities within the teams. Furthermore with this document we aim at a situation where each team member is capable of performing the test activities in the right way; where we address the 'tester' we imply each team member that performs test activities. Testing in a Scrum team is a team activity: as a team you develop test scenarios, as a team you decide what will be tested, where and how (manual or automated). Tasks coming from that activity, and shown on the Scrum board, can be picked by anyone in the team.

How to read

The next chapter gives an introduction into the World of test practice improvement and the way we approach the improvement of Agile testing. Chapter 3 focuses on the skills of the tester (or the one performing test activities), both soft and technical skills.

[1] See: Driven TPI NEXT® - Business Driven Test Process Improvement; Gerrit de Vries et al. – UTN Publishers 2009.
[2] See: TMap NEXT® in Scrum; Leo van de Aalst en Cecile Davis, Sogeti Nederland B.V. 2012.
[3] See: *The Scrum Guide, July 2013*

In Chapter 4 we describe, based on the Checkpoints of the TPI NEXT® model, which practices can be applied to the test activities. This is addressed on three operational views:

- P: Test practices for the individual developer (Professional)
- T: Practices for the Team as whole
- O: Practices on the level of Organizations, programs or projects.

The document also contains an appendix. This is the text from the book (Business Driven test Process Improvement), Chapter 4. The appendix can be used as a reference to obtain more information on the Key areas or their related Checkpoints.

Tooling

This paper comes with a AGILE TPI-tool that can be used to assess and analyze practices and organizations. For the Agile situation we have adapted the TPI NEXT®-tool, so it can easily be used to find the bottlenecks in testing and obtain indications for improvement. The tool (and its user manual) can be downloaded from www.tpinext.com.
The advised user manual in the Appendix will help you use the tool properly.

Table of contents

1 Introduction

1.1 Need for improvement?

Test activities play an important role in each software development program, whether that be in the so-called waterfall environments or where more agile and iterative methods are being used. Organizations run the risk of defects and flaws in their software programs, systems or complete end-to-end IT solutions. That's one of the main reasons for testing. Another important driver for test activities is to gain confidence and trust in the IT products from a user and business perspective. Hence, 'testing' needs to produce insight in the quality of products and solutions.
Apart from that more and more organizations require *efficient* testing: the results of testing activities are produced with a minimum of effort, spending as little time, money and other resources on the required level of quality. In TMap® terms this means: Risk Based and Human Driven testing[4], using the right instruments in the right way at the right time.

Ever since 1998 organizations have used the TPI NEXT(R) model to assess their test maturity and improve where possible. In 2009 we delivered the TPI NEXT® model to enhance the approach in a more business driven manner and focus on environments like Agile. The TPI NEXT® book provides a section where the use of the model in these environments is described. Although the recommendations in this section are still valid we feel that there is room for improvement as well. Not in the least because some organizations are still struggling to gain the benefits of excellent testing activities in an Agile environment.

1.2 Test activities distinguished…. but not separated

Providing an article on the improvement of Agile test activities sounds like a *contradictio in terminis*;

- *"Scrum[5] recognizes no titles for Development Team members other than Developer, regardless of the work being performed by the person; there are no exceptions to this rule;*
- *Scrum recognizes no sub-teams in the Development Team, regardless of particular domains that need to be addressed like testing or business analysis; there are no exceptions to this rule;*
- *Individual Development Team members may have specialized skills and areas of focus, but accountability belongs to the Development Team as a whole".*

So, one could say that a 'testing process' does not exist, let alone the role of tester, test engineer or test manager. Still, test *practices* do exist in Agile so test activities can be distinguished from other development activities and therefore can be assessed, evaluated and improved where necessary. Furthermore the referred *specialized skills* are worth looking at. From years of experience as testers we believe that the tasks and activities of the tester are not only crucial for the quality of final products; the tester also is the 'center of operations' when it comes to the quality of requirements, designs, semi products, user support and acceptance support. See also: The tester as coach.
This position gives the tester (or rather: the developer who performs test activities) a unique opportunity to coach the fellow developers. From his experience he is able to improve not only his own activities but the activities of the whole team. For this reason we have distinguished the test activities in Agile environments; not to separate them or set them apart, but to make them and the other development activities as strong as possible.

1.3 Different views and levels

Improvement of test activities in Agile environments can be initiated on different levels and different starting points. Furthermore many organizations are in a transition from some method to Agile. What we will use as a focus is the operational view of the test activities:

- Test activities performed by one or more **individual** team members.
- Test activities performed by one or more **teams**.
- Test activities from an **organizational** perspective.

In the following paragraphs we will use these distinctions to apply the elements of the TPI NEXT® model for improvements and guidelines.

---

[4] Read more about TMap Human Driven testing on www.tmap.net.
[5] From: *The Scrum Guide, July 2013*

2 The AGILE TPI Model

The original TPI NEXT® model has been used as a starting point to set up recommendations to improve Agile test activities. The main difference with the original model are the maturity levels. We have taken the formerly mentioned operational views (individual/team/organization) where a certain level of maturity can both be measured and improved. Furthermore the Clustering method is used to set out stepwise improvement.

2.1 Key areas

As in the original model there are 16 Key areas[6] to group the different test activities and aspects. These Key areas are applicable to all three operational views.

		Individual	Team	Organization
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			

2.2 Categories

In the model and specifically the Maturity matrix the operational views and categories replace the original maturity levels. The original matrix showed four levels of maturity: Initial, Controlled, Efficient and Optimized. For the Agile TPI model the initial level has been removed, the other three levels have been replaced (see illustration below) by:

- Individual level
- Team level
- Organizational or project level

In the Agile TPI model maturity may be indicated and grow per operational view (and of course per Key area). Depending on the achieved and fulfilled checkpoints the levels are indicated with a percentage (for example: 55% score on the Individual level).

[6] See Appendix A for further descriptions.

		Individual	Team	Organization
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			

2.3 Checkpoints[7]

In order to value the level of maturity of a certain view and to indicate improvement suggestions the adapted TPI NEXT checkpoints are used. In the illustration below the checkpoints are indicated by their Key area (first position), the operational view (second position) and the sequence number (third position). Example: 1.P.1 stands for: the first checkpoint for the Professional level belonging to the first Key area (Stakeholder commitment).

		Individual				Team				Organization			
1	Stakeholder commitment	1	2	3		1	2			1			
2	Degree of involvement	1	2	3	4	1				1	2		
3	Test strategy	1	2	3	4	A	C	C		A	C		
4	Test organization	1	2	3	4	D	F			D	G		
5	Communication	1		1		1	2			1			
6	Reporting	1	2	3		1	2	3		1			
7	Test process management	1	2	3		1	2	3		1	2		
8	Estimating and planning	1		2		1	2			1			
9	Metrics	1				1				1	2		
10	Defect management	1	2	3		1	2	3	4	1	2		
11	Testware management	1		2		1	2			1	2		
12	Methodology practice	1				1				1	2	3	4
13	Tester professionalism	1	2	3	4	1				1	2		
14	Test case design	1	2	3		1	2	3		1			
15	Test tools	1		2		1				1	2	3	
16	Test environment	1	2	3		1				1	2	3	

2.4 Setting priorities

Priorities may differ between all the checkpoints or recommendations; one checkpoint will be more important than the other. One of the reasons to set priorities to the checkpoints is the 'business driven'[8] aspect of the model: in the Agile situation the Key areas have the following significance as they are more or less contributing to strengthen Agile:

[7] In the Test maturity matrix of the AGILE TPI-tool the identification of the checkpoints is replaced by their Cluster letter (A, B, C etc.).

[8] See also: *TPI NEXT Business Driven Test Process Improvement, Appendix B3.1. Categorized Key areas.*

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

	High
	Medium
	Low

The other reason to set priorities for checkpoints is their interdependency: some checkpoints need to be fulfilled to enable another checkpoint.
This prioritization is formalized in the Clusters.

2.5 Clusters

Clusters are a combination of certain Key areas and a limited number of Checkpoints. Coming from an initial level and growing to a higher level of maturity the improvement steps are indicated by A (the first step), B (second step) and so on. This is indicated in the illustration below, where the first cluster (A) of all three levels is marked green.

		Individual				Team				Organization			
1	Stakeholder commitment	C	D	D		C	E			C			
2	Degree of involvement	A	A	B	C	A				A	B		
3	Test strategy	A	B	C	D	A	C	C		A	C		
4	Test organization	D	E	F	G	D	F			D	G		
5	Communication	A		C		B	D			C			
6	Reporting	D	F	F		D	E	F		E			
7	Test process management	F	F	G		B	B	E		E	F		
8	Estimating and planning	B	B			A	B			F			
9	Metrics	B				F				G	G		
10	Defect management	C	D	E		A	C	C	D	D	E		
11	Testware management	C	C			E	F			D	E		
12	Methodology practice	B				C				A	B	B	C
13	Tester professionalism	A	B	B	E	A				A	A		
14	Test case design	A	B	B		A	B	C		C			
15	Test tools	A	A			A				A	C	B	
16	Test environment	A	C	C		A				A	A	A	

Setting priorities

Note that each category has its own set of clusters and that each cluster has a unique set of checkpoints: Cluster A on the Team level is different from Cluster A on the Individual level. The clusters give an indication

of priority: what should be done first, what comes later. In practice improvements will be implemented by making combinations of checkpoints from different operational views.

3 Improvement strategy

3.1 Determine where you are

It is always advised to determine the starting position: how is testing done to-date, what are the known or hidden bottlenecks, which activities are performed quite well, what are possible quick wins. This starting position can be measured quite well by using the AGILE TPI assessment tool, illustrating the test maturity with the help of all the checkpoints. Using the assessment will not only provide insight in the test maturity. It also indicates the dependencies with other capabilities and it enables prioritization for short and long term improvements: what should be done first, what comes next (see also Clusters).

3.2 Starting from scratch

When testing activities need to be organized and set up in Agile teams, where both the teams, the projects and the organization are new in the Agile area and where Agile is the approach for all projects, it is advised to use the AGILE TPI model with a broad focus. This strategy combines the checkpoints and improvement steps of all the three categories, thus taking care of a generic approach. Start with assessing the situation by using all the checkpoints from the first (A) cluster[9] in all categories (P, T and O). Then determine the necessary steps for improvement, also based on the clusters in all categories. Make an improvement plan, covering the improvement steps, worked out into estimated and planned activities and actions, tasks and responsibilities.

3.3 The hybrid situation

In the situation where both waterfall and Agile are used we advise to use the TPI NEXT® model for the waterfall situation and the AGILE TPI model for the Agile situation. Comparing the assessment results of both models makes it possible to take adequate decisions on what necessary improvement steps need to be. Furthermore implement improvement activities in both environments, following a detailed action plan based on realistic estimates and planning. Combining and comparing the two makes it possible to learn from each other, use each others' best practices. Make sure management is committed to the necessary changes and improvements indicating and setting priorities for both or for one of the methods.

3.4 Stepwise improvement with Clusters

Improving the test activities in Agile environments must be done in a controlled way and stepwise. That is where the Clusters can be used. Coming from an initial level and growing to a higher level of maturity the improvement steps are indicated by A (the first step), B (second step) and so on.
The first improvement step consists of all the checkpoints from the different Key areas in Cluster A. An improvement plan and strategy needs to define to feasibility of the improvement steps; sometimes it is necessary to take smaller steps, include less checkpoints to improve. The next paragraphs show the stepwise improvement using Clusters.

3.4.1 First step: Cluster A

KEY AREA	NR.	CHECKPOINT
Degree of involvement	2.1.1	Test activities like product risk analysis, planning, preparation, test case design are performed prior to test execution, with the goal of keeping the test activities off the sprints' critical path.
Degree of involvement	2.1.2	Testers/developers participate in project and sprint planning in order to align the test activities with other activities.
Degree of involvement	2.2.1	The developer/tester contributes to the optimization of the test basis (more than a testability review), in which the object under test is described.

Degree of involvement	2.3.1	Lessons learned from the evaluated test activities are used for (the set up of) future projects.
Test strategy	3.1.1	Product risks are defined and managed during the project e.g. by risk poker for stories.
Test strategy	3.2.1	The strategy is monitored and where necessary adapted during the project.
Test strategy	3.3.1	The process of product risk analysis and strategy is evaluated during the retrospective or during the refinement of user stories.
Communication	5.1.1	Each individual developer/tester (pro)actively participates in the daily stand-ups to provide and receive relevant information.
Tester professionalism	13.1.1	Developers with a test role have followed a test training, specifically on test engineering and test case design.
Tester professionalism	13.2.1	Required expertise – either industry, business and/or technical – for and within the team is at hand when needed.
Test tools	15.1.1	A test tool is used to automatically perform regression testing
Test tools	15.1.2	The tools are available and can be used at any required time if not permanently.
Test tools	15.2.1	Training for the proper use of (implemented) test tools is provided and followed.
Test tools	15.3.1	Maintenance, knowledge about and support for the test tools in use is available.
Test tools	16.1.1	The (automated) test cases can be run on all relevant environments.
Test tools	16.1.2	The required test environment configuration and test data are clearly defined at the start of the iteration.

3.4.2 Next step: Cluster B

KEY AREA	NR.	CHECKPOINT
Degree of involvement	2.1.3	Quality and test aspects are well considered in the analysis and mitigation of overall project and sprint risks by team and Product owner.
Degree of involvement	2.3.2	Training and coaching facilities are provided before sprint activities start.
Test strategy	3.1.2	The test activities and strategy (including retest and regression testing) are based on the product risks.
Communication	5.2.1	Relevant information (like product quality and risks) is defined and effectively communicated with team and stakeholders.
Tester professionalism	13.1.2	The essentials of a test method like TMap or ISTQB are well known to and applied by the developers.
Tester professionalism	13.1.3	Individual knowledge and skills are shared with the team e.g. by letting others test.
Tester professionalism	13.3.1	Training and certification facilities in test methods, like TMap NEXT® or ISTQB® are provided by the organization.
Test tools	15.3.2	The use of tools is evaluated against aspects of testing faster, cheaper, better or making the test process more manageable
Test tools	16.2.1	Working agreements with supplying parties regarding availability of environments are well defined and aligned.

Test tools	16.3.1	A logical or functional design of the test environment is provided and maintained by the organization. It addresses applications, systems and their connections, and the use of stubs and drivers (mock-ups).

3.4.3 Cluster C

KEY AREA	NR.	CHECKPOINT
Metrics	9.3.1	Means and procedures to collect, store, provide and manage metrics that support the Agile teams are documented.
Defect management	10.1.1	The defect life-cycle is as short as possible; defects are solved during the sprint where they were injected or, even better, during Refinement and Planning or even Creating the user stories.
Defect management	10.2.1	Handling of the defects is assigned to a specific role (not only developers but also Product Owner and Scrum master).
Tester professionalism	13.3.2	Facilities to participate in Special Interest Groups, attend test seminars or read and apply test literature are available.
Test case design	14.1.1	A well-considered choice, based on product risks and strategy, is made to document test cases on a logical level using a template or a tool
Test case design	14.2.1	The coverage level of the test basis is clearly indicated (for example in the requirements document).
Test tools	15.3.3	A tool policy is defined for all agile activities
Test tools	16.1.3	A predefined checklist is used to check the (availability and functionality) of the environment.
Test tools	16.3.2	A technical design of the test environment is available and formally accepted by the test manager or the test environment specialist.
Degree of involvement	2.1.4	Quality and test aspects are well considered in the impact (risks) of changes (in priority, user stories, functional or technical solutions) in the refinement session.

3.4.4 Cluster D

KEY AREA	NR.	CHECKPOINT
Stakeholder commitment	1.1.2	The Product Owner's (pro) active tasks and responsibilities are carried out well.
Stakeholder commitment	1.1.3	All relevant stakeholders are defined and known to the whole team.
Stakeholder commitment	1.3.2	Budget for resources (including testing) is granted by and negotiable with the Product Owner.
Test strategy	3.1.4	When more than one test levels are executed (unit testing, functional testing, acceptance testing) the tests are differentiated between the levels (test types, test coverage and test depth), depending on the analyzed risks.
Test organization	4.1.1	The quality and test tasks and responsibilities are clear and aligned with the Scrum master and product owner.
Test organization	4.2.1	With more than one tester in the team and in case of more teams: test activities are reconciled and aligned.

Test organization	4.3.1	The tasks, role, responsibilities and expectations for the tester-role are clearly defined timely before the sprint activities start.
Communication	5.2.2	Different means of communication are considered for effective messages inside and outside the team.
Reporting	6.1.1	The reporting in the standup contains aspects of time and/or costs, results and (above all) product risks.
Reporting	6.2.1	The reporting from retrospectives contains trends and recommendations concerning test time and cost.
Reporting	6.2.2	The reporting from retrospectives contains trends and recommendations concerning product quality and -risks.
Defect management	10.1.2	When defects are transferred to a following phase or sprint, the following data is stored: the related test case, person reporting the defect, date, severity category, description (the actions to reproduce the defect, expected and observed result) and defect status.
Defect management	10.3.1	All teams and projects use the same defect management system (tool) ór a seamless connection between all systems is provided.
Methodology practice	12.1.1	Test activities are performed following a described guideline, the test products to be delivered and additional requirements on the way of working.
Methodology practice	12.2.1	The practical use of the guidelines or test method is evaluated by/in the team.
Methodology practice	12.3.1	The test method describes for all activities the goal, the responsible role, any tools and techniques to be used and the preconditions.
Tester professionalism	13.1.4	Testing skills are discussed and evaluated against expectations on a regular basis with team and management.
Test case design	14.1.2	Each documented test case contains the following elements: • A (description of the) initial situation • The change process = test actions to be performed • The predicted result (!) • The required test data.
Test case design	14.2.2	A checklist or other means such as reviews are applied for quality aspects (for example usability) that cannot be covered by test cases.
Test tools	16.3.3	The ownership of test environments and test data is assigned to a distinct department.

3.4.5 Cluster E

KEY AREA	NR.	CHECKPOINT
Stakeholder commitment	1.2.2	Information on the quality of both the test activities and the test object is provided to stakeholders.
Test organization	4.1.2	The quality and test tasks and responsibilities are clear for the whole team.
Reporting	6.3.1	Information from the teams is used for improvements of all quality and test activities and the software development lifecycle.

Test process management	7.1.1	The backlog and the definition of done include test activities to be performed and the required quality.
Test process management	7.1.2	The sprint planning and/or backlog refinement are aligned with the Product Owner.
Test process management	7.2.1	The definition of done is discussed with the Product Owner and other relevant stakeholders.
Test process management	7.3.1	The (re)allocation of test resources is delegated to the team(s).
Metrics	9.3.2	The efforts to provide metrics are measured against their benefit for Agile activities, e.g. the way metrics contribute to the need for information from stakeholders.
Defect management	10.1.3	The developer/tester contributes, from a testing perspective, to the risk analysis of defects.
Defect management	10.2.2	The whole team has access to the defect administration (preferably a tool).
Defect management	10.3.2	Defects are analyzed on a project/program level and test practice improvements are initiated to prevent future defects.
Methodology practice	12.3.2	A complete and comprehensive set of templates is part of the test method.
Test case design	14.1.3	In the test case or test script a reference is made to the test basis (like use cases and user stories), describing a specific system behavior.

3.4.6 Cluster F

KEY AREA	NR.	CHECKPOINT
Test organization	4.1.3	The quality and test work in progress can be processed by any team member.
Test organization	4.2.2	The team feels accountable for success and failure of product and process quality.
Reporting	6.1.2	The reporting (frequency and content) to the Product Owner matches his/her basic demands for the decision making process.
Reporting	6.1.3	Reporting is done in writing as and when required by the stakeholder(s).
Reporting	6.2.3	Reporting demands from stakeholders is balanced against the effort for providing them.
Test process management	7.1.3	Each test activity is visible for the team and monitored and where necessary adjusted.
Test process management	7.2.2	Anticipated deviations of the backlog are discussed with the Product Owner in the backlog refinement session.
Test process management	7.2.3	Adjustments of the backlog are documented as and if required by the stakeholders.
Test process management	7.3.2	The coordination and management of test activities is evaluated regularly, internally by the team and with stakeholders.
Estimating and planning	8.1.2	For the estimation the activities taken into account are: test planning, defining test cases, gathering of test data, executing test cases, conservation of test cases.
Estimating and planning	8.2.2	The estimation techniques are evaluated during the retrospective.
Estimating and planning	8.3.1	Metrics like burndown/burnup or velocity are defined and maintained to support the estimating and planning

		activities of the teams.
Defect management	10.2.3	Defects are analyzed in the retrospective(s) to indicate the quality of the test object.
Testware management	11.1.1	The test basis, the test object and all testware (scripts, data, plans) are identified by name and version.
Testware management	11.2.2	There is a procedure and a logical structure to manage test data, testware, the test basis and the test object.
Testware management	11.3.1	Guidelines are provided for conserving testware for further use across projects.
Methodology practice	12.3.3	Mandatory, conditional and optional elements of the test method are well indicated.
Test case design	14.2.3	The accuracy and effectiveness of test cases are evaluated and improved where necessary.
Test case design	14.3.1	Test design techniques and checklists are evaluated, adjusted and maintained for further re-use.

3.4.7 Cluster G

KEY AREA	NR.	CHECKPOINT
Test organization	4.1.4	The assigned roles, products and services are clear to stakeholders outside the team.
Test organization	4.3.2	The quality and testing products and services are evaluated regularly on an organizational level.
Estimating and planning	8.1.1	For (test) effort estimation, simple techniques are used such as planning poker.
Estimating and planning	8.2.1	The need for a plotted planning, indicating the dependencies between test varieties and teams, is discussed with the stakeholders.
Defect management	10.2.4	Trends in defects are analyzed. For this, more information is recorded about a defect: the subsystem, priority, program and version, test basis and version, root cause, all status transitions and problem solver.
Testware management	11.1.2	Each test case is easily traceable to the test basis in a transparent way.
Testware management	11.2.1	Items under testware management are accessible for each team member.
Testware management	11.3.2	A procedure is defined for the handover of testware to upcoming projects.
Methodology practice	12.3.4	The test practice is periodically evaluated and adjusments are implemented where necessary.

4 Improving test capabilities

4.1 Focus on testing skills

4.1.1 Introduction

Like the Scrum-guide says, individual team members may have specialized skills and competences. In this chapter we will focus on these testing skills.

4.1.2 Technical skills

The tester was and will be a specialist in testing. The specialist in the process of the preparation, design, execution and reporting on the results of test cases. To this respect there is no difference between Agile and other environments. So, the following skills will always apply to the one who is responsible for testing:

- Have an understanding of performing testing and applying test design techniques
- Know how to set up test cases and scripts in a logical order.
- Have a clear judgment on the results of tests.
- Have an understanding of and experience with test coordination / management tasks such as risk analysis and stakeholder management.
- Know how to estimate and plan test activities
- Have a vision on product quality.
- Have the ability to explain the how, what and why of testing.
- Show affinity with tooling and engineering.
- Clarify the need for (technical) improvements.

In short, the technical skills required for testing activities in the Agile situation are not so much different from the technical skills we already know[10].

4.1.3 Soft skills

Without a doubt the Agile situation requires communication, on every level, between all participants, team members and stakeholders. Communication also requires a pro-active attitude: do not wait until someone will ask for information. Anything that will help you and your team needs to be shared.
Pro-active communication also means having an interest in information. So you are very curious to hear about ideas, opinions and suggestions: anything that can help you and your team.
Furthermore the next aspects will help to become the best player of the team.

Risk based testing

More than in the waterfall situation a guiding principle in Agile development is: 'good is good enough'. That implies that there will be risks of not reaching the required or desired scope or functionalities or the risk of defects. As a tester you must be willing to take chances, take risks or better: think and act risk driven. Analyze the possibilities of making mistakes, consciously considering possible impact. Being risk driven is not only related to the product under test or the project. It relates to your whole attitude, realizing that you have to set priorities to the things you do or don't do, people you work with, decisions you make.
Testing practice:

- Consider skipping some tests you planned to do and deliberate possible risks with your team mates. It may speed up the process and create testing debt but it should never jeopardize the stated quality (as mentioned in the definition of done) and the principle of working software. Always emphasize that quality is a whole team approach and responsibility.
- In the planning poker session estimate 'risky': challenge yourself and your team to take chances in the estimates.
- When you find defects also consider the risks of non-repair (possibility of workaround) or no retest after repair.

Take responsibility

[10] see also *Tester professionalism* in the TPI NEXT book, Chapter 4.13

While in the waterfall and comparable projects and situations you, as a tester, could sometimes hide in your expertise-skin and be a perfect tester at the same time. In the Agile situation you need more. Being part of the Agile team means: you are as an individual responsible for the results of your team. At the same time your team is responsible for the results of their individual members. That requires more than the rather technical skills we discussed in the former chapter. You need to come forward, communicate clearly, be assertive and most of all: be a team player.
Testing practice:

- Expand your (test) focus to other disciplines in your team, discuss your mutual responsibilities and tasks.
- See how you can help your team mates to get to the best possible results
- Be self critical and discuss possible improvable skills with your team mates to find out how they can help you.
- Show others, also Scrum master and product owner, how you tested, what you have done or did not do and why e.g. the coverage of the use cases or requirements.

Challenge your competences and skills

The results of your team will be greatly appreciated if you succeed in performing beyond expectation. In a positive way, that is. That requires an eager attitude: you want to learn from mistakes, you are willing to take chances, willing to step outside your comfort zone. At the same time you have an open mind towards your team mates and stakeholders outside the team. You share your knowledge with the team and challenge them to share their knowledge with you.
Do not be afraid to fail; the best way to learn is to make mistakes and learn from them.
Testing practice:

- Use a test design technique you have not used before
- Initiate and conduct the Product Risk Analysis and practice with your team

Be organized

In the Agile situation a test plan is seldom made. Activities and processes are short cycled and controllable. Nevertheless it is wise to plan, prepare and structure your work. It helps to be organized; a chaotic desk does not really help to find the things you need quickly. Furthermore, being responsible for your own actions also means proper management of these activities: structure, align, be in control, be visible.
Testing practice:

- Prepare your tests as soon as possible, for instance before programming starts or during pairing with the developer. Where possible have your tests integrated in unit tests. This is repeated for every user story..
- Set up your test cases in such a way that they are easy to understand for your teammates.
- Explain to your team how you work and ask for feedback.

Trust and respect your team mates

As a tester you are used to being critical, eager to find mistakes that other people made. That's good and it's why you were asked to be a tester. But in doing so always show respect to those who made that mistake, who caused that defect in the application. And trust them to learn from those mistakes and improve their deliverables. Moreover compliment them on a job well done. They will in turn respect you for the help you provide to offer good quality.
Testing practice:

- Trust the other team mates to be able to execute your test cases or encourage them to test themselves.
- If you find a defect, report it in a constructive way e.g. by providing possible solutions or workarounds.
- Discuss your test cases with the other developers and ask for feedback.

4.1.4 The tester as coach

Through the years and evolving from their daily test practice testers have become the center of operations within the application lifecycle. On the one hand driven by the necessity of input from the other disciplines: architects, designers, developers, business (management and users), operations, maintenance and

suppliers. Example: the tester needs a reference for his test cases: what is the expected outcome or result of this specific function, which action is required to perform this function. On the other hand the other disciplines and parties depend on the results of the tests and are eager to know the outcome: (when) can we go live, what is the quality of (intermediate) products.
In this position and based on his broad knowledge and experience the tester may act as coach for other disciplines. Especially the agile teams and the stakeholders outside can benefit from this knowledge, coming from testers in their role of test engineers, test managers, test team leads, etc. in different phases of the application lifecycle. The following paragraphs show examples of the knowledge and experience that the 'tester' may contribute to Agile teams.

Skills to Initiate a project:

- How to set up and manage a project.
- Knowledge of budgeting or estimations.
- Developing and using templates.
- Analyze and mitigate Product risks:
    - Who is involved in a risk assessment.
    - How to determine what risks are.
    - How to determine the seriousness of the risks (classification).
    - Mitigation activities.
- Analyze and mitigate project risks.

Preparation skills:

- Knowledge of required functionality through the essential part of a test case: desired outcome or the expected result of a function, process or activity.
- Make the design (both functional and technical design) itself more readable and analyzable for reviewers.
- Translate the wishes from users and avoid surprises at the User Acceptance Test.

Skills to support Development/implementation:

- Manage defects and their repairs:
    - Provide root cause analysis.
    - Possible and feasible solutions.
    - Work-arounds.
    - Risk assessment of defects.
    - Plan and schedule: fix now or later.
- Is a retest required yes/no!
- And also:
    - Skilled in bringing the bad news.
    - Diplomacy.

Skills to support Acceptance:

- Testing is not just a search for errors and defects. Testing provides confidence in the product.
- User support:
    - Sharing of experiences / results.
    - What is good, not so good.
- Support the design of test cases.
- Provide information on the coverage and mitigation of (business) risks.

Skills to support Operations:

- Support users.
- Write user manual.
- Test used as a training/training used as a test.
- Explain the logic of programs, functions, systems.
- Testing 'in production':
    - Support bug fixes.
    - Quick repairs.
    - Smart re-test.

5 Appendix A: Key areas and their significance

This appendix describes the Key areas as they are used and explained in detail in the book Business Driven Test Process Improvement TPI NEXT®. Note that in Agile situations the essence of the Key area may be the same but terminology may be different.

#	Key Area	Group	Significance
K01	Stakeholder commitment	SR	**Committed and proactive stakeholders and create good conditions for efficient communication and cooperation:** The commitment of the persons involved in testing is an important condition for a smooth running test process. Stakeholders can be project management or line management, but also (end) users, internal accountancy and marketing. The principal stakeholder provides the necessary time, money and resources (quantitatively and qualitatively) to perform a good test.
K02	Degree of involvement	SR	**Tight involvement of testing in the project helps to improve the product quality from the beginning, and helps to keep test activities off the project's critical path:** Timely preparation and coordination between different tests can be done and the time that the test project is on the critical path of the project can be kept as short as possible. Early involvement of the test team in the software development lifecycle helps to find defects as soon and easy as possible and perhaps even to prevent errors. At an early stage the test team supports the analysis of project risks, the review of requirements and designs.
K03	Test strategy	SR	**The test strategy guides the test process towards an optimal allocation of effort and resources:** The test strategy defines the distribution of the test effort and the coverage over the parts to be tested or aspects of the test object. The test strategy aims at detecting the most important defects as early and cheaply as possible. The qualification 'most important' is driven by the test goals such as business processes, critical success factors or perceived product risks.
K04	Test organization	SR	**A test organization meets the needs of projects for test resources, test products and test services:** Testing is the work of human resources. Each participant will have their own skills, tasks, responsibilities, authorities, competences and interests and this needs to be clear to everyone involved. The test organization deals with all these aspects and relationships in order to have smooth running communication and processes.
K05	Communication	SR	**Distinct communication ensures common understanding and alignment of expectations between all parties involved:** In a test process, clear and effective communication between people involved is needed for creating the right conditions, coordinating the test strategy, tuning activities, negotiate priorities, etc. Communication takes place in different directions, from, towards and within the test team. Communication at its best encourages interactivity and feedback.
K06	Reporting	SR	**Reporting provides stakeholders with insight to support the decision making process and test project accounting:** Reporting, as a one-way communication, aims at providing substantiated

			advice and insight in the quality of both the test object and the test process. Reporting provides test project accounting and supports the decision making process. This makes reports one of the most important test products.
K07	Test process management	TM	**Managing the test process maximizes the execution of the test assignment within the required time, costs, and results:** Test process management is about keeping the resources, stakeholders, the infrastructure, the techniques and the cycle of planning, preparation, actions, checks in control and related to each other. Proper test management ensures an effective and efficient test process.
K08	Estimating and planning	TM	**The proper use of appropriate estimating and planning techniques makes the test process planning and estimating predictable and reliable:** Test estimating and -planning indicate which activities have to be carried out, what the necessary resources (people) are and when they are needed. Proper use of estimating and planning techniques makes the test process predictable and reliable.
K09	Metrics	TM	**Metrics provide objectivity by quantifying observations:** Metrics are quantified and objective observations of the characteristics of a product or process. They are used to control the test process, to substantiate the test advices and to compare systems and/or processes. Specifically for improving the test process metrics are important to evaluate consequences of certain improvement actions, by comparing data before and after performing the improvement action.
K10	Defect management	TM	**Defect management treats defects both at an individual level and at a group level where root causes are analyzed and guidelines are available:** Although administrating and managing defects is in fact a matter of the entire software development project and not specifically for the test team, the testers are the most involved people. A good administration is able to monitor the life-cycle of a defect and provide various (statistical) overviews and reports such as the quality advice.
K11	Testware management	TM	**Testware management ensures coherence between test artifacts, and between test artifacts and their related design documents:** The input products (such as design, test object) as well as the output products of testing (such as test cases, scripts, test plans, reports) must be maintainable and reusable, so they must be managed. Furthermore, testware management ensures coherence between testing artifacts on the one hand and between testing artifacts and their related design documents on the other hand.
K12	Methodology practice	TP	**A described method for testing directs and supports the test projects:** For each test activity or test process a certain working method is used, comprising actions, procedures, regulations, techniques etc. This test method needs to be sufficiently generic to be applicable in every situation, but contains enough detail to direct and support daily test practice.
K13	Tester professionalism	TP	**Tester professionalism includes the right mix of the various skills, competences, disciplines, functions and knowledge that are necessary to perform test activities to the levels expected:** Testing requires a large set of skills: familiarity and understanding of the business and the organization, knowledge of the test object, general IT know-how and technical skills. In addition, testers need to have specific test skills,

			from expertise in methods through to the ability to communicate with stakeholders effectively (social competence). Tester professionalism refers to the right mix of these skills and competences that are necessary to perform test activities to the levels expected.
K14	Test case design	TP	**Test case design directs test execution to search for defects according to the test strategy:** Test cases are derived from source information (the test basis) in a standardized way. This approach provides insight into the coverage and depth of tests and makes the (re)design and the (re) execution of the test cases easier.
K15	Test tools	TP	**Test tools enable or accelerate specific test activities:** Test tools are automated or unattended aids for the test process. Test tools excel in systematically doing repetitive work, independent of complexity and volume, in situations where manual activities would take too much time, would be infeasible or error prone. Test tools enable or accelerate specific test activities, resulting in shorter lead time, more test depth, better motivation of testers etc.
K16	Test environment	TP	**The test environment is explicitly designed, implemented and maintained with the goals of the test level in mind:** The test environment mainly comprises the following components: • Client environment • Network • Storage • (Enterprise) Server • Middleware • Test data The test environment should be composed and set up in such a way that it reflects the test goals. The test environment has a large influence on the quality, lead time, and cost of the test process. Important aspects of the test environment are responsibilities, management, on-time and sufficient availability, representativeness, and flexibility.

6 Appendix B: Guidelines for the AGILE TPI Tool

Quick Start

For all macro's to do their work (like automatically produce the maturity matrix) please click on the {Options} box behind the security warning and choose the use of macro's.
Click on the first worksheet [Information] to enter specific data about the assessment, analysis or improvement project at hand. Do not forget to save the file with a new name.
Depending on the view that you choose (individual, team or organizational level) use one of the worksheets or use the worksheet [All checkpoints]. All these worksheets are automatically synchronized. The checkpoints can be valued by yes [y,Y] or no [n,N]. Filling in the Notes-column is advised for later refinement or when you want to use the tool for following assessments.
After all checkpoints have been valued the tool produces an overview of the results in the worksheet [Test maturity matrix].

For improvement purposes the not-fulfilled checkpoints can be used as a reference for improvement activities. Defining the proper improvements is also supported by the Clusters: start with checkpoints in Cluster A, followed by B etc.

The following guidelines will detail the different steps, activities and the worksheets of the tool."

Detailed reference

GENERAL

When opening the file unlock and activate the macro's. This will ensure that the results are automatically processed into the Maturity matrix sheet.
After enabling the macro's save the file under an appropriate name (see also Quick start).

WORKSHEET INFORMATION

Column	Guideline
C	Click on the cells in the rows under C to fill the proper information. The information in cell B4 [Project} and B8 {Company name} will be added to the Maturity matrix Clicking on cells C15 and C16 provides a box to choose the proper dates.

WORKSHEET ALL CHECKPOINTS

Column	Guideline
A*	**Seq.Nr.** This column refers to the numbers of the checkpoints, indicated by [Seq.Nr.]. The sequence number combines: Position 1: Reference to the Key area Position 2: Indicates the operative angle: P = Professional or individual level T = Team level O = Organizational or project level Position 3: the sequence number of the checkpoint Example: 2.P.1 stands for the first checkpoint of the Key area Degree of involvement (Key area 2) for the Professional level.
B*	**Cluster** The letters shown in this column refer to one of the five Clusters (A, B, C, D, E of F). Example: The E in row 12 means that the checkpoint is part of Cluster E.
C*	**Checkpoint** In this column the checkpoints are defined. Note that in some checkpoints more than one variable may exist (e.g. the word "and" is used). All variables are applicable for the analysis and valuation, except where "Example" is mentioned.
D	**Fullfilled?**

This is the column to enter the results of an assessment or analyses. Clicking the cell provides a checkbox for possible values. The default value is 'NA' resulting in a non-colored cell in the [Test maturity matrix]. Options are:
- 'y' or 'Y' when the checkpoint is verified and fulfilled. This results in a green-colored cell in the [Test maturity matrix].
- 'n' or 'N' when the checkpoint was verified but not fulfilled. This results in a red-colored cell in the [Test maturity matrix]

E **Notes**
This area is necessary for the foundation of your analysis results. Here you can explain why a certain checkpoint is, or is not fulfilled or not applicable. Possible explanations can be that no proof has been found in documents and/or in interviews.
It is advised to add information and notes to all the checkpoints, except for the ones that were not checked. This will help to support your reports and presentation for management. It also helps to make the document transferable to other analysts and/or for future assessments.

* Note: Columns A, B and C are fixed. They cannot be changed.

WORKSHEET TEST MATURITY MATRIX

Column	Guideline
general	This sheet shows the results of the analysis and the values of the checkpoints as they were entered in worksheet [All checkpoints] column D. It contains the Maturity matrix, the results per category and in total (absolute figures), and the Spider graph for the results per Key area (percentages).
A -AK	**Row 8-23** These cells contain all checkpoints and their values. The checkpoints are marked with a letter corresponding to their Cluster. The colors have the following significance: grey: the checkpoint was not analysed and/or valued; as a result of the 'NA' value. red: the checkpoint was analysed but was not fulfilled; as a result of the 'n'or 'N'value. green: the checkpoints was analysed and has been fulfilled; as a result of the 'y' or 'Y' value. Click on one of the cells (checkpoints) to see their text (in Row 4) and/or their Comment (in Row 5) as a result of the notes in the [All checkpoints] worksheet. **Row 4 and 5** Row 4 shows the text of a checkpoint when this checkpoint is selected and clicked on in the maturity matrix. Row 5 show the Notes that were added to that checkpoint in worksheet [All checkpoints]
AN-AU	**Row 1-9** The graphic shows the results of the values of all checkpoints in absolute numbers. All fulfilled checkpoints will be shown as a blue color, the non-fulfilled or not analysed checkpoints will remain grey. Bars from left to right: 1st bar: score of all (109) checkpoints 2nd bar: score of all checkpoints in the category Professional 3rd bar: score of all checkpoints in the category Team 4th bar: score of all checkpoints in the category Organization **Row 11-23** The spider graph shows the results of the values of the checkpoints in [All checkpoints] per Key area, as percentages of the total. The results will be presented as a green line in the web.