27

September 2014

testing experience

The Magazine for Professional Testers

NEW: New Columns & Book Corner

ISSN 1866-5705 www.testingexperience.com free digital version print version 8, 00 € printed in Germany

Column

by Erik van Veenendaal

Test Process Improvement and Agile: Friends or Foes?

Currently I am leading a project to describe the implementation of TMMi in an Agile environment. There is much debate as to whether software and test process improvement still have added value when using Agile methodologies. Many Agile purists state that there is absolutely no added value and we should complete ignore all process improvement methods. Coming more from a practical background and approaching this with an open mind, I strongly beg to differ. Many organizations struggle when they are in transition from a sequential life cycle to an Agile iterative life cycle. It is interesting when you discuss testing with representatives from these organizations to find out they (still) have many problems and are looking for concrete answers. Using Agile makes a strong contribution to being more flexible (e.g., in terms of requirements to be implemented) and providing business value. However, it is not a silver bullet that will solve all our quality problems and make testing obsolete. There is little to no proof that introducing Agile will "automatically" also improve product quality (see Figure 1). In this column I will briefly discuss some of the aspects that need to be taken into account when performing test process improvement in an Agile context.

Of course, using the Agile life cycle model has a decisive influence on the way in which test process improvement is approached. The improvement culture here is closely aligned to the iterations and can be characterized as follows:

- Improvement is considered at frequent intervals (e.g., at the end of a sprint when using SCRUM).
- The scope of the improvement is often limited to the cycle (e.g., a sprint) that has just taken place, the aim being to improve little and often.
- Improvements are closely coupled to the problem, and waiting times for improvements to be implemented are minimized.

The principal aspects to be considered when applying an Agile life cycle model in the improvement context are:

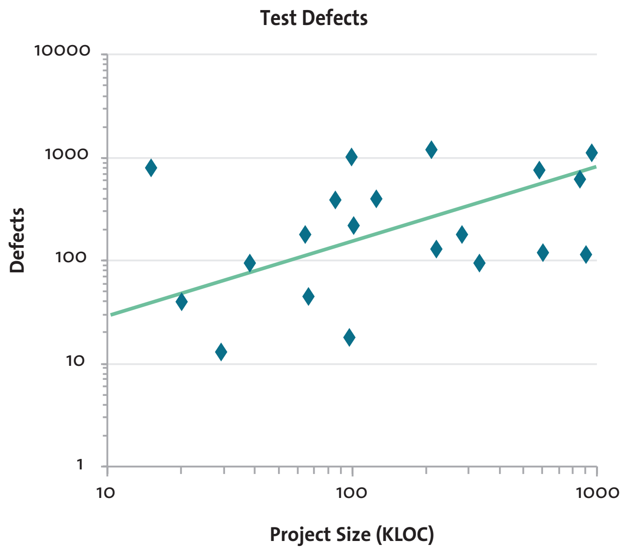
- Improvement cycle frequency
- Organizational aspects
- Scope of improvement
- Source of improvements
- Level of (test) documentation
- Improvement methods
- Support from test process improvement models

Within projects that use Agile life cycle models, improvements generally take place in frequent feedback loops that enable test process improvements to be considered frequently, e.g., when applying SCRUM, at the end of a sprint, or even as part of a daily stand-up meeting. Retrospectives are a standard and important tool that will drive (test) improvements. A team-based improvement focus is already embedded in Agile. As a test improver, the challenge is to make use of this improvement cycle, take the improvements to another level (e.g., facilitate cross project learning), and institutionalize them where necessary.

Because the scope is often limited to the previous sprint, small but frequent improvements are made that focus mainly on solving specific project problems. The focus of these improvements is often not on cross-project learning and institutionalization of improvements. Looking at the organization of test improvement, we find that there is likely to be less focus on test process improvement at an organizational level and more emphasis on the self-management of teams within the project. These teams generally have the mandate to change the testing process within the project according to their needs, resulting in highly tailored processes. However, some organizations also use weekly test stand-up meetings to take things to a higher and cross-project level.

Since there is a more project-specific focus on (test) process improvement, less emphasis is likely to be placed on broader issues affecting testing across the organization. This could mean, for example, that fundamental testing problems may not be fully addressed because they are beyond this project-centric context. A typical example here is the approach taken to testing certain quality attributes, such as performance and reliabity. These issues may be deferred from sprint to iteration because they often require more skills and resources than the project team has available. In these areas, it is hard to make a substantial next step without major investment. Solving problems only on a project level could also easily lead to suboptimization and losing touch with the bigger picture.

In Agile contexts, the range and number of alternative improvement ideas to be considered may be considerably more than with sequential life cycle models. Since most members have a part-time testing role within the project, these ideas can come from any project member. This places a stronger emphasis on evaluating and prioritizing improvement suggestions, which may be more of a team effort than a task assigned to a test process improver. Since this may require the specific testing knowlegde of a test process improver, they can also act as a consultant to the team if required to do so.



Agile defects are "randomly" scattered around the industry average. No pattern or conclusion for higher product quality (less defects) in Agile projects can be derived.

Figure 1. Agile defects vs. industry average

In projects using the Agile methodology and practicing Agile testing techniques, such as extreme programming (XP), do not expect to find the level of test documentation you would expect from projects using a sequential life cycle. There may be a single combined "test document" covering the essential elements of a test policy, test strategy, and even a high-level test plan. Test process improvers should avoid making "improvement" suggestions that call for more rigorous and thorough test documentation. Like it or not, this is not part of the life cycle approach. One of the main Agile principles is that documentation is created only when there is a clear and unambigious need for it.

Improvement methods and models

The methods used to propose test process improvements when using an Agile life cycle will tend to be analytical methods for evaluating the root causes of problems, such as cause-effect diagrams. These are particularly useful methods for the problem-solving mindset that prevails at the end of a sprint. Note, however, that the life cycle used does not dictate the improvement method used.

Analytical approaches often go hand-in-hand with model-based approaches to test process improvement, and this is also true for projects that use an Agile life cycle. However, more tailoring of the models is required. When using a process improvement model such as TPI NEXT or TMMi, more help is available to make the necessary adjustments for Agile and iterative life cycles.

The official TPI NEXT book includes chapters that show how to use the model in Agile and iterative projects. This includes, for example, a list of the principal key areas to be considered and how their checkpoints should be best tailored and interpreted. In addition, the TMap NEXT content-based methodology (which forms the methodological foundation for the TPI NEXT model) is tailored for SCRUM projects in "TMap NEXT in Scrum", so that TMap can also be applied in Agile and SCRUM contexts.

The TMMi website provides case studies and other material on using TMMi in Agile projects. I have personally provided consulting services to a small financial institution while achieving TMMi level 2 and to a medium-sized embedded software company while achieving TMMi level 3, both employing the Agile (SCRUM) life cycle using the "standard" TMMi model. Note that within TMMi, only the goals are mandatory, not the practices.

As stated with TMMi, a special project has been launched to develop a special derivate that focuses on TMMi in Agile environments. The main underlying principle is that TMMi is a generic model applicable to various life cycle models and various environments. Most (specific) goals and (specific) practices as defined by the TMMi have been shown to be also applicable in Agile environments. Remember, testing still needs to be done in a professional way. However, many of the sub-practices and examples and their interpretation are (very) different. As a result, the TMMi Foundation is not developing a new maturity model but will document the way TMMi can be applied in an Agile environment. It will determine whether each “standard” TMMi goal and practice is also applicable for testing in an Agile life cycle. Some goals (or practices) may just not be. For each goal and practice that is applicable, typical lightweight Agile sub-practices and examples will be defined. Watch the TMMi website (and my tweets) for the latest updates and results of this project.

Focus on business value

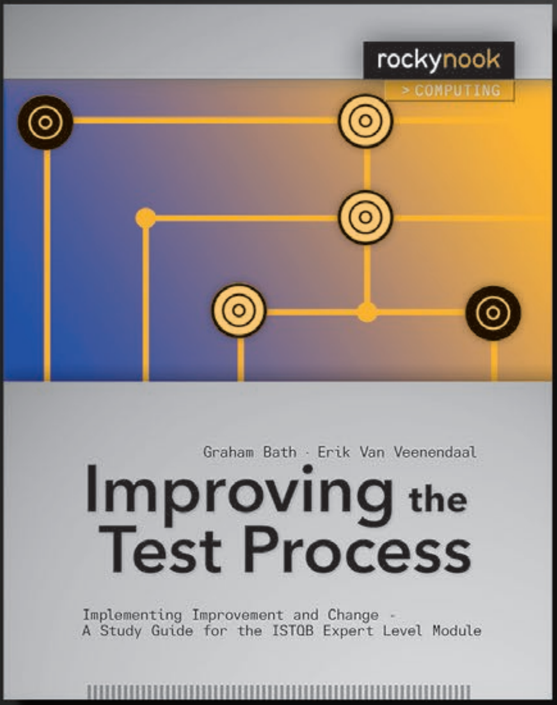
As always, anything you like to improve through testing needs to have added value. Never improve for the sake of following a model. This sounds obvious on paper, but in practice I have seen so many organizations making this mistake. Whatever you do, make sure you know why you are doing it and what it means in the Agile context. If you cannot identify the added business value, do not do it! Process improvement must be constantly reviewed against the business drivers. Following this essential principle will help you to be successful, including in Agile environments.

> about the author

Erik van Veenendaal is a leading international consultant and trainer, and a widely recognized expert in the area of software testing and quality management. He is the founder of Improve Quality Services BV (*www.improveqs.nl*). He holds the EuroSTAR record, winning the best tutorial award three times! In 2007 he received the European Testing Excellence Award for his contribution to the testing profession over the years. He has been working as a test manager and consultant in various domains for more than 20 years. He has written numerous papers and a number of books, including “Practical Risk-Based Testing: The PRISMA Approach” and “ISTQB Foundations of Software Testing”. He is one of the core developers of the TMap testing methodology and a participant in working parties of the International Requirements Engineering Board (IREB). Erik is also a former part-time senior lecturer at the Eindhoven University of Technology, vice-president of the International Software Testing Qualifications Board (2005–2009) and currently board member of the TMMi Foundation.

Twitter: *@ErikvVeenendaal*

Website: *www.erikvanveenendaal.nl*


rockynook
> COMPUTING
Graham Bath · Erik Van Veenendaal
Improving the Test Process
Implementing Improvement and Change -
A Study Guide for the ISTQB Expert Level Module