Integrate Test Activities in Agile Projects

Leo van der Aalst
& Rik Marselis, Sogeti

EuroSTAR
Software Testing
Community

In traditional testing a tester distinguishes various phases. To be able to effectively test in Agile projects the test activities should not be phased but be properly integrated in an Agile approach. This eBook starts with a vision on Agile testing and, based on that, describes how to effectively integrate test activities in the Scrum approach.

1. Introduction

People not working in Agile sometimes have the impression that Agile projects are chaotic, unorganized and uncontrolled. And indeed if discipline and structure are really lacking, an Agile project will deliver poor quality software or will not deliver at all!

A proven approach on how to measure quality is by means of testing the product. However both the Agile Manifesto [AMF, 2001] and the Definitive Guide to Scrum of Schwaber and Sutherland [DGS, 2011] don't give you many clues on how to do that.

Therefore in this eBook we describe how you can effectively test in Agile projects.

2. The Agile Manifesto has a Lack of Footholds for testing

The basis for all Agile projects is the Agile Manifesto [AMF, 2001]. The four values and twelve principles of the Agile Manifesto say nothing concrete about a specific approach to systems development or testing. An Agile approach, such as Scrum for example, scarcely provides a foothold to reach a method for quality assurance and testing. Also the Definitive Guide to Scrum [DGS, 2011] does not add much knowledge on quality and testing. In addition, real-life practice shows that the meanings of and the relations between Lean, Agile and Scrum are not equally clear to everyone. It will be no surprise that the integration of a test and Scrum approach does not always run smoothly. Actually, in their Definitive Guide to Scrum Sutherland and Schwaber state that "Scrum is (…) extremely difficult to master".

In the implementation of an Agile approach, organizations find it difficult to gain a good overview of which parts of their current test approach can be reused, and which cannot. The only thing the Definitive Scrum Guide says about testing is that there are no sub-teams dedicated to particular domains like testing (…). For this reason, it happens all too often that the organization decides not to adopt any existing testing practices at all. However, this means that the benefits of structured testing are not realized (see box below). In our opinion to make sure that product quality can be measured we must integrate a structured testing approach with the Scrum events.

Benefits of a structured approach to testing in Agile projects:

- It gives insight into and advice about any risks that may arise in relation to the quality of the system under test.
- Defects can be found and tracked at an early stage, providing a better insight into the root cause of the defects.
- Defects may be prevented from arising.
- While test execution tasks still remain on the critical path, delays should have fewer consequences.
- Test products, such as test cases for example, can be reused.
- The test process is transparent and controllable.

- The automation of testing tasks (e.g. test specification and/or test execution) can be pulled earlier into the project.

3. Vision on Testing in Agile

The four values and twelve principles of the Agile Manifesto have a neutral relationship with a test approach. And Scrum, which is an approach based on the Manifesto, also scarcely provides any guidance for integrating testing in an Agile approach.

Firstly we provide a brief overview of our vision on testing in Agile environments. You could regard the statements in this vision as add-on testing values for the Agile Manifesto. For a detailed understanding of this vision, we refer to the point-of-view paper entitled 'Testing in Agile Software Development Environments with TMap NEXT' [TAE, 2010].

The four statements of the vision, followed by a short elucidation:

1. **Use the Agile Manifesto as the starting point.**
   The four values and the twelve principles of the Agile Manifesto form the starting point of each test activity in an Agile approach.

2. **The test process must be integrated in the Agile approach.**
   a. The test activities must be integrated in the development process.
   This means that testing is not a separate phase but is rather a continual activity of the Agile team. This also entails a different implementation of test levels.
   b. All team members must be prepared to perform test activities.
   Although the team should contain a professional tester, this does not mean that all test activities can be carried out by this tester.
   c. Testing is the driving force behind the quality of the project.
   Testing must not be seen as the last safety net before the software is implemented. The tester collaborates with all team members to provide continual information on the product's quality and its satisfaction of the business requirements.
   d. The use of automation is becoming increasingly important and indispensable in the realization of a successful Agile project. Model Based Test Design (automated generation of test cases) and Model Based Test Execution (automated execution of these test cases) are valuable practices here.
   e. Testing must be incorporated into the Definition of Done.
   The Definitive Guide to Scrum state that "each increment is additive to all prior increments and thoroughly tested, ensuring that all increments work together". Thus, it is important to include the test aspects in the Definition of Done.

3. **Find the right balance by making conscious choices.**
   For example: working software is more important than comprehensive documentation, which means that the 'lack' of documented information must be compensated by other forms of communication.

4. **Use the strength of the four TMap NEXT essentials.**
   a. Be adaptive.
   b. Use techniques and tools suitable for Agile environments.
   c. Apply a business-driven test management approach that is geared to the Agile environment to implement a risk-based approach.
   d. Use the TMap NEXT lifecycle and TMap

NEXT activities in an Agile way.

4. The Test Method as a Basis for Integrating Activities

The test method we have integrated with the Scrum events is TMap NEXT® [TMapNEXT, 2006]. This is a test method that consists of four essentials, seven phases and, as the basis, fifty-three activities.

An important essential of TMap is the adaptiveness. This enables TMap to be applied in a flexible way in any situation. In order to be able to apply TMap well in a certain situation, it may be necessary to alter, remove or supplement some activities and techniques. We understand that it is sometimes troublesome to keep a good overall view of the situation. That is why this eBook demonstrates how TMap can be integrated with the Scrum approach to support a true Agile approach in which testing is fully integrated and not a separate activity.

4.1 Testing must be fully integrated

Experience has taught us that testing is not only an extremely important activity in a Scrum approach, but has also led us to believe that testing must be fully integrated with this approach in order to be as Agile as possible. Of the many Scrum variants with different concepts, we have chosen the variant of Schwaber and Sutherland [DGS, 2011] as a basis, because this is the one most frequently used and it provides a good basis to elaborate on, taking account of new insights.

In this eBook, we explain our vision and translate it to concrete applications. Thus, we regard testing as an integral component of the scrum approach, NOT as a process running parallel to it (see figure 1).

Please keep in mind that both Scrum and TMap are merely approaches that attempt to support the project team delivering a product at the right

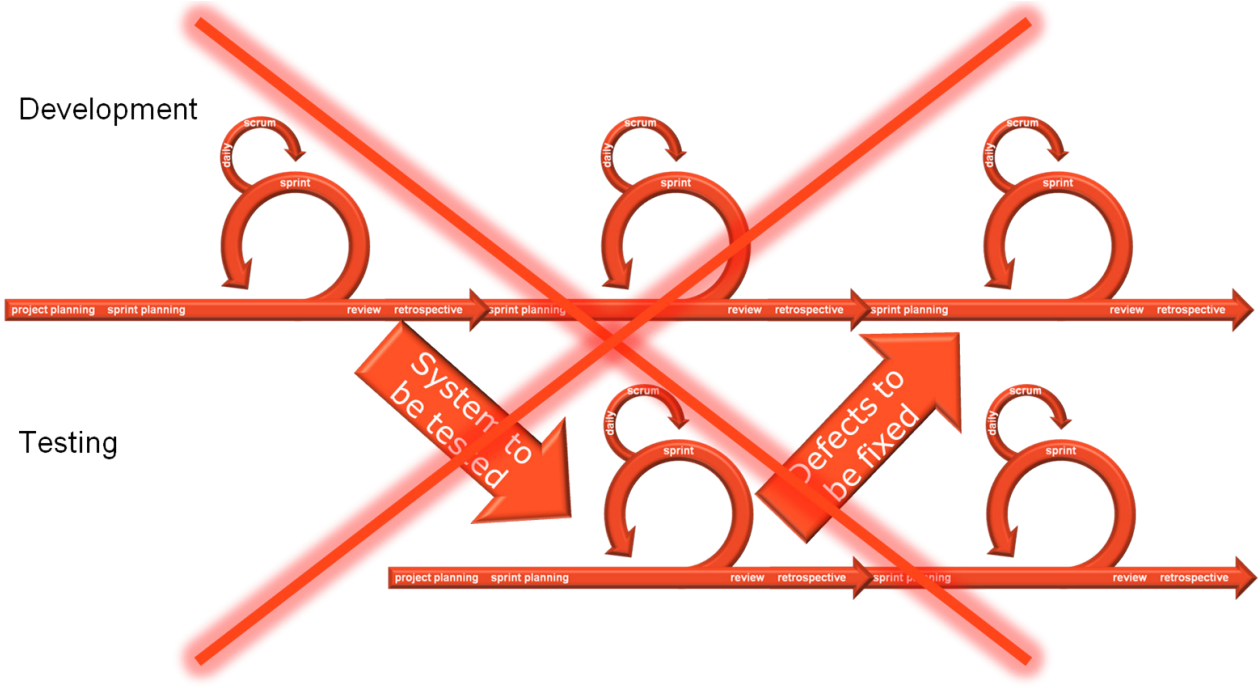

Figure 1: Testing must not be implemented as a separate process, nor as separate sprints

time-to-market, with a fit-for-purpose quality at an affordable cost and with an acceptable risk level. An approach can be adopted without any adjustment into a one-to-one structure, or can be taken as a guideline that is regularly re-aligned. In the first case (implementing as out-of-the-box), chances are that it will not work perfectly on all fronts, as all projects are different, and the approach will become a tight straitjacket that will eventually irritate.

The second application (being adaptive) requires more initial effort to learn and implement, but it provides a tailored flexible framework, geared to an ever-changing world.

Our suggestions will have to be implemented in an adaptive manner to comply with your own specific situation and project requirements. However, we assume that, with this eBook and the detailed explanation of our vision of testing in Scrum in the book 'TMap NEXT in Scrum' [TNIS, 2013], we can help you along the path to measure product quality and increasing your business success [PointZERO, 2012].

4.2 Mapping TMap NEXT phases on Scrum

Based on this vision we have mapped the TMap lifecycle phases on the Scrum model. The result of this mapping is shown in figure 2:

The phases originally contain many activities. Some activities are relevant for the Scrum approach, others are not. So we have determined which are the relevant activities and matched them with the Scrum events.

4.3 Test activities in one overview

In this eBook we will not explain how the individual test activities within the phases are integrated with the scrum events but show you the end result right away. If you are

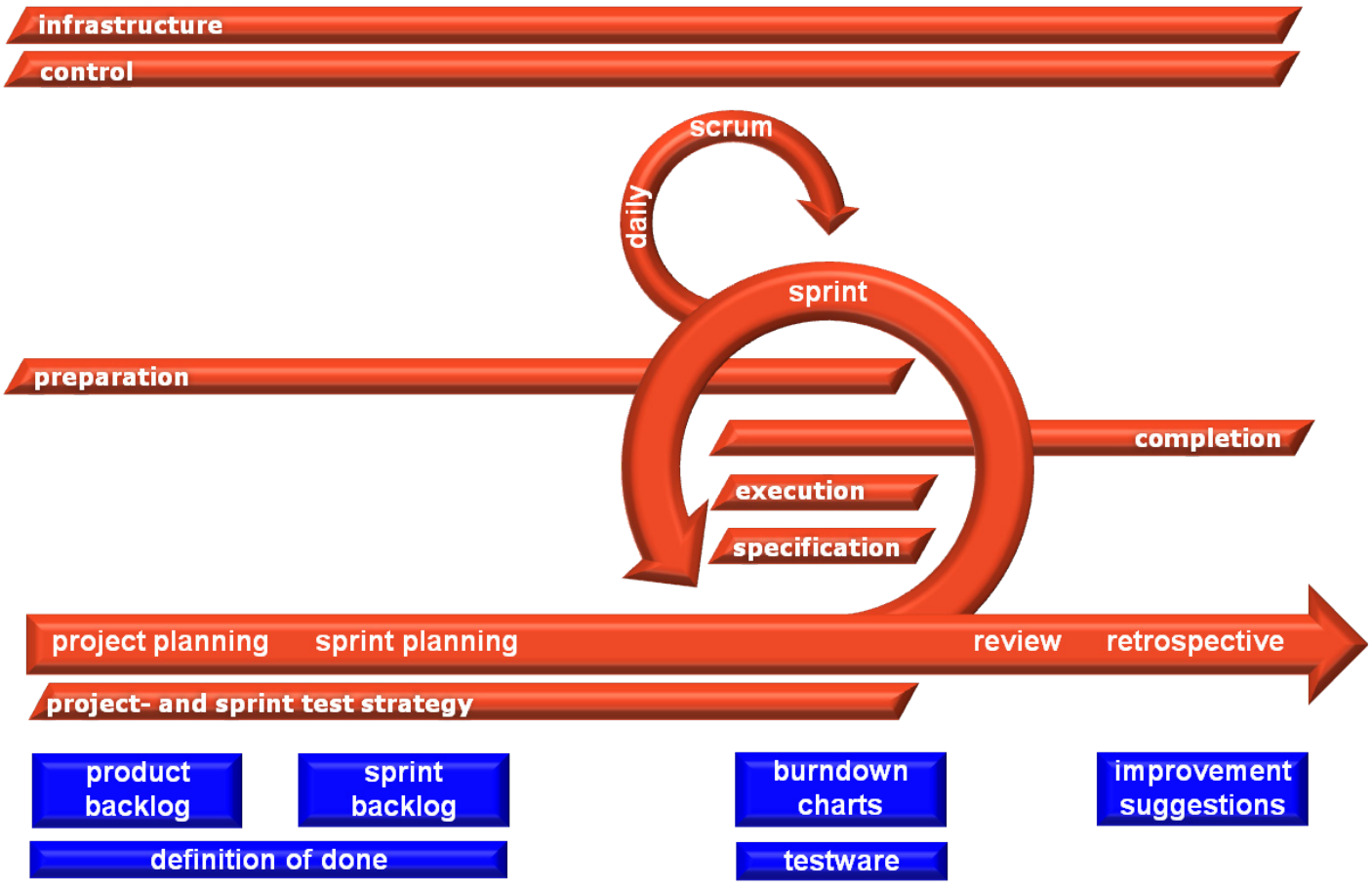

Figure 2: TMap lifecycle phases mapped on the Scrum model

interested in a detailed explanation please refer to the TMap NEXT in Scrum book [TNIS, 2013]. Figure 3 demonstrates how the test activities are integrated with the Scrum events. These activities can be carried out by all team members. Please note that, in the total overview, you can act in an adaptive way. Various activities are probably not relevant for your project or they may be executed at some other time, or you may add your specific activities. The model is thus easily adapted to the situation within your own organization or project.

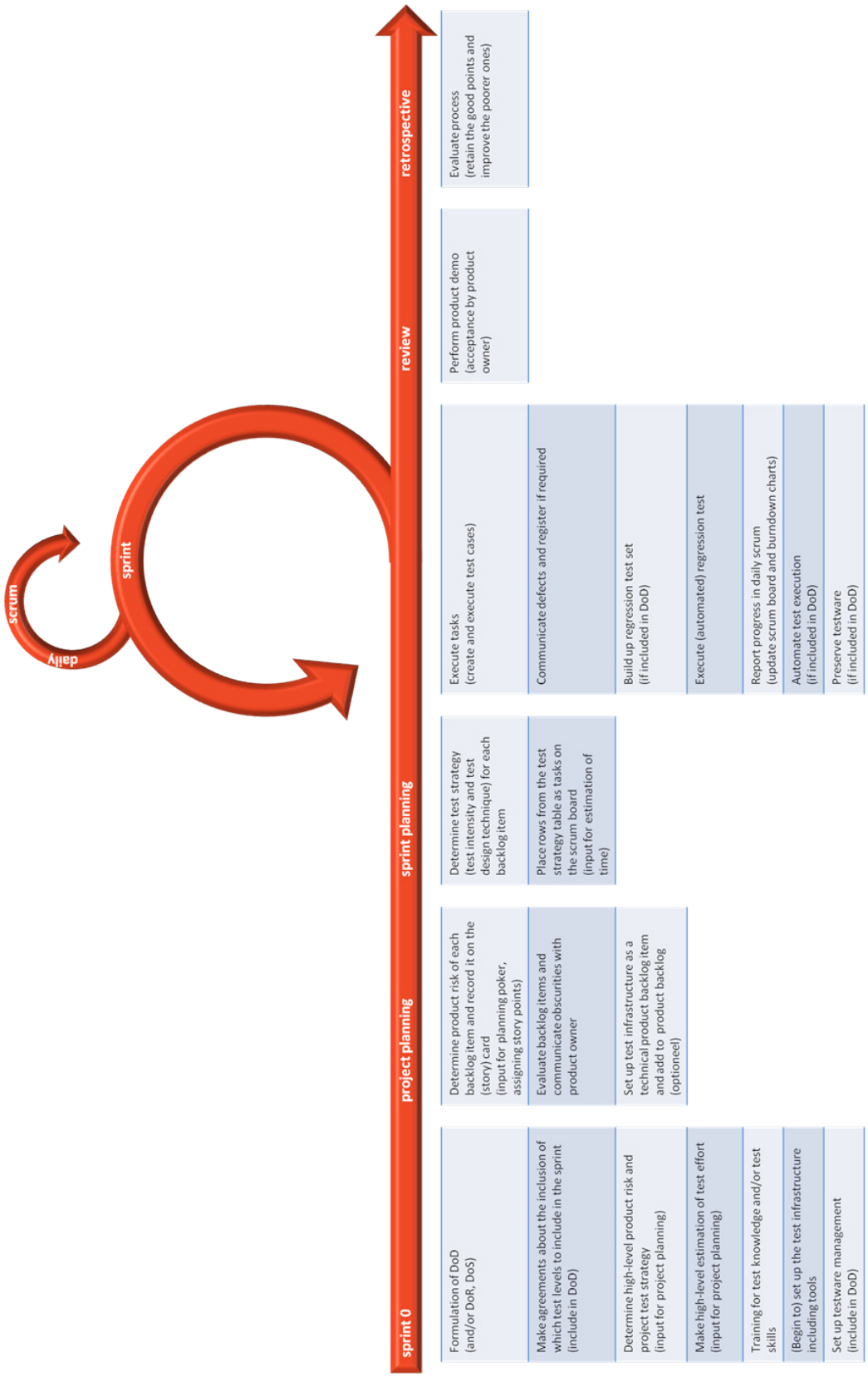

Figure 3: Total overview of test activities in the Scrum events

5. In More detail

There is a lot more to say about testing in a Scrum environment. We can not address all testing aspects in this eBook. Therefore we have identified some aspects which are always subject to discussion. These aspects are:

- What is the role of the Product Owner and the Scrum Master?
- Which team members perform the test activities?
- How to deal with the test strategy?
- Which test levels (e.g. unit-, system-, acceptance test) should be executed?
- Is test automation necessary?
- What aspects should a Definition of Done contain?

5.1 What is the role of the Product Owner and the Scrum Master?

The Product Owner and Scrum Master have their roles as described in the Definitive Scrum Guide [DGS, 2011]. With respect to testing the main focus is on the criteria they set for the level of quality that is needed to reach a fit for purpose business solution, and the level of risk that is acceptable for the organization. Also the Product Owner will be involved in accepting the product which may need him to take part in acceptance testing activities.

5.2 Which team members perform the test activities?

In Agile teams all team members simply have the title of 'developer'. Still many different capabilities are needed within the team which asks for a variety of people on the team.

Next to various experts it is advisable to have a professional tester in the Agile team to guarantee the expertise on quality and testing of the team. This tester should possess knowledge of how to perform a risk analysis, how to review artifacts, and how to formulate and execute test cases using the relevant test design techniques.

But this does not mean that all test activities must be carried out by this tester. All team members must be prepared to perform test activities. To be able to perform all activities necessary in each sprint, team members will need to take part in the specification and execution of the test cases, for example. In this setting, the tester can act as a coach. Of course, this will work reciprocally, the tester may be asked to support other team members, and this requires a broadening of the knowledge and skills of the tester.

A few examples of what a tester in an Agile environment should be capable of: the tester is a skilled communicator, is flexible, possesses knowledge of the domain, is creative but also practical, is solution-oriented, customer-oriented, a team player, acts as a spider in a web, supports the product owner in the formulation of product backlog items and acceptance criteria, must be able to support a business analyst, must be able to evaluate unit tests, and be prepared to pair with a developer (both on design and programming). However, the most important characteristic is that the tester must be proactive and open-minded, and, as a part of the team, feels responsible for the delivery of a good result.

5.3 How to deal with test strategy?

In the Scrum model we distinguish two events, the Project Planning and the Sprint Planning, in which we could define a test strategy (see figure 4).

5.3.1 Project test strategy, defined in the Project Planning

The planning schedule for the total test process
is formulated at the beginning of a project. However, Scrum is not primarily concerned with the actual planning of tasks in terms of fixed points in time. These are placed on a scrum board and become 'active' when the time is right. At this point in time of the Scrum project, the focus lies on defining a global test strategy. We refer to this as the 'project test strategy' — in a traditional developmental environment this would be a component of the master test plan. It is good to be aware that the formulation of the project test strategy takes place during the 'planning' scrum event, and is incorporated into the project planning (product backlog) schedule.

The main objective of creating the project test strategy is to establish what testing activities are suited to cover the risks that are identified for the product that is created by the team. This includes deciding the test-depth that is required for various parts of the product and the level of regression testing that has to be performed.

5.3.2 Sprint test strategy defined in the Sprint Planning

The sprint test strategy is defined during the sprint planning event and included in the sprint backlog. For example, during the sprint planning stage, the development team estimates, with the aid of planning poker, the amount of time required for each task of a sprint backlog. The amount of time required is influenced by factors such as whether the test has to be thorough or light, which is related to the product risk. Therefore we advise to include the risk classification of a backlog item — particularly in the case of user stories — in addition to the priority specified by the product owner, before the planning poker is initiated. To make this practical just add the risk level to the product card on the Scrum board. During the planning poker the risk then is one of the factors that determines the effort that is agreed.

5.4 Which test levels (e.g. unit-, system-, acceptance test) should be executed?

There is always a discussion about which test levels to include in a Sprint. And also about the inclusion of an end-to-end test in a Spint. In this section we will give our vision on this topic.

5.4.1 All test levels in a Sprint

According to both the Agile and the Scrum theory, a working product should be delivered at the end of the sprint. In concrete terms, this means that this is a tested product that is accepted by the product owner. Therefore, the planning, analysis, design, development and all tests (unit test, system test, acceptance test) take place in a sprint. In this situation it is not necessary to talk about test levels. However, in real-life practice, deviation from the Scrum process does occur regularly. We see variants that we cannot recommend: in one sprint, software is developed, including the unit tests, whereas in another a system test and an acceptance test are subsequently executed. This is not actually the substantiation of a sprint as we advise and in which testing is integrated with scrum. Some organizations approach this differently, and specify that the sprint within which the software is developed, including the unit tests, should be followed by one or more — traditional — test phases after the delivery of the project. This does cause a so-called 'technical debt'. Defects are then included in and solved during a subsequent sprint. This may ensure that the quality of the software considerably declines during long-term projects with many sprints. And the effectiveness of scrum with an integrated test process is lost.

5.4.2 End-to-end testing as separate phase

Theoretically it should be possible to execute an end-to-end test in a sprint. But, in practice, this turns out to be difficult due to the participation and mutual attunement of several parties and consequently lead times that are often longer than the length of a sprint. For this reason, we have the opinion that the end-to-end is a test level for which doing this as a separate phase after the delivery of the project, is a valid deviation from the Agile and Scrum principles.

5.5 Is test automation necessary?

In this section we will discuss two possibilities for test automation. The first one is straightforward, the second is Model Based Testing.

5.5.1 Automate the test execution

An ever-present project risk in Agile projects is the availability of capacity to execute regression tests. In each sprint the product grows and during each sprint regression testing is needed to make sure the product still meets the expectations (both on functional and non-functional aspects). In practice we have seen that performing a regression test every 4 or even 2 weeks is not feasible without tooling. During the creation of new features for every feature the regression test is created and added to the automated regression test pack. A team member that is involved in programming can use some time at the start of a sprint to create an automated test script for the features that were finished during the previous sprint. That way the time of all team members can be used in an optimal way.

Of course automated testing does not have to be limited to regression tests. Other tests can also benefit from automation, as a component of the continuous build and of integration strategies, for example. As is the case with many aspects, the actual plan to develop and execute an automated test ought to be included in the Definition of Done.

5.5.2 Model Based Testing

In practice, various solutions are chosen for the automation itself, sometimes during the sprint by the team members themselves, sometimes parallel to the sprint and by others. Another approach to test automation is to use Model Based Testing [PointZERO, 2012]. We distinguish three possibilities to use models in testing activities. Since based on the Agile Manifesto we strive not to create superfluous documentation we suggest to use models to show what the system should do and how processes should work (instead of lenghty and fuzzy texts). Examples of models are process flows and data structures. Using Model Based Reviewing these models can easily be evaluated, risks can be found and the quality level is established while the team members discuss the model. Next the model can be used as input for Model Based Test Design, a tool generates the test cases so the effort of the team members is significantly reduced. Finally these test cases can be provided to a test execution tool and Model Based Test Execution takes place. The main advantage of the Model Based Testing approach is that when the system changes, only the models have to be maintained. The test set is automatically generated each time the model has been changed. Thus the regression test set is always up to date without the risk of technical debt.

5.6 What test aspects should a Definition of Done contain?

The Definitive Guide to Scrum describes the Definition of Done. An extension we often see is that in the sprint 0 of large projects, the Scrum team formulates three kinds of definitions: Definition of Ready (DoR), Definition of Done (DoD) and Definition of Shippable (DoS). The formulation and fulfillment of the DoR aims at ensuring that all criteria for successfully

launching a sprint have been met. The formulation of a DoD is always necessary to determine whether or not activities and products meet the specified criteria and are consequently 'done'. When the sprint has been completed, the DoS determines whether or not the product can be released into production and used. The DoR and DoS are optional, the DoD is mandatory.

From a test perspective, a DoD contains:

- the criteria that have to be met in a sprint with regard to the defects procedure
- a specification of the test intensity that is to be used while creating the test cases, based on the established product risk
- the agreements made concerning the test process
- the agreements made concerning the test results
- the test levels that have been included in the sprint.

In general, one can say: do not allow anything that is not completely ready into the sprint, and do not allow anything that is not quite done to escape. A sprint can only be classified as done if the testing has also been done.

Sources

[AMF, 2001]
Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. (2001), Agile Manifesto, Ohio, http://Agilemanifesto.org/

[DGS, 2011]
Schwaber, K., Sutherland, J. (2011), The Definitive Guide to Scrum: The Rules of the Game, http://www.Scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum_Guide.pdf

[TAE, 2010]
Davis, C., Aalst, L. van der (2010), Testing in Agile Software Development Environments with TMap NEXT, Vianen, http://www.tmap.net/tmap/downloads/testing-Agile-software-development- environments-tmap-next

[TMapNEXT, 2006]
Aalst, L. van der, Broekman, B., Koomen, T., Vroon, M. (2006), TMap Next for result-driven testing, 's-Hertogenbosch: UTN Publishers, ISBN 90–72194–80–2 (available on www.ict-books.com)

[TMIS, 2013]
Aalst, L. van der, Davis, Cecile (2013), TMap NEXT in Scrum – Effective testing in Agile projects, Sogeti Nederland B.V., Vianen, ISBN 978 90 75414 64 6 (book), 978 90 75414 65 3 (ePub), www.ict-books.com

[PointZERO, 2012],
Marselis, R., Roodenrijs, E., (2012), The PointZERO vision, ISBN 978-90-75414-55-4 (book), 978-90-75414-56-1 (ePub) (available on www.ict-books.com)

Biography

Leo van der Aalst has more than 25 years of testing experience. He is experienced in international projects and consultancy trajectories and developed amongst others services for Agile testing.

Leo is lector "Software Quality and Testing" at Fontys University of Applied Sciences (Eindhoven, The Netherlands) and he is co-author of the "TMap NEXT® in Scrum" guide, the "TMap NEXT® for result-driven testing" and "TMap NEXT® Business Driven Test Management" books. He is also a member of the Dutch Innovation Board Testing, the Dutch Country Board of ISO/NC 381007 "Software and System Engineering" and member of the Research & Development unit of Sogeti Netherlands.

Besides all this, Leo is a much sought-after teacher of international test training courses, a regular speaker at national and international conferences, and he is the author of several articles (a.o. testing in Agile software development environments).

Rik Marselis is one of Sogeti's most senior management consultants in the field of quality and testing. He played an important role in organizing quality and testing in various organizations in different branches, by making practical implementations of well-known methods and approaches. Rik started his career over 30 years ago in software development and gradually shifted to testing and quality management. He was author of many articles and contributed to 14 books on software testing and quality management, for example "the PointZERO vision", "Quality Supervision", "TMap NEXT® in Scrum", "TPI NEXT®" and "TMap NEXT® BDTM". As a trainer and coach Rik has worked together with numerous IT-professionals to further improve their skills. Also Rik is a much appreciated presenter at conferences throughout Europe.

Join the EuroSTAR Community...

Access the latest testing news! Our Community strives to provide test professionals with resources that prove beneficial to their day-to-day roles. These resources include catalogues of free on-demand webinars, ebooks, videos, presentations from past conferences and much more...

Follow us on Twitter @esconfs
Remember to use our hash tag #esconfs when tweeting about EuroSTAR 2013!

Become a fan of EuroSTAR on Facebook

Join our LinkedIn Group

Add us to your circles

Contribute to the Blog

Check out our free Webinar Archive

Download our latest eBook

www.eurostarconferences.com