

Sensemaking in the Agile Forest

Agile and testing¹

Henny Portman

Introduction

Of course, an agile team will test too. This often involves testing techniques such as test-driven development (TDD), Acceptance Test Driven Development (ATDD) and A/B testing. TDD is a developer's tool and tests the correctness of a feature. ATDD ensures that requirements are well defined and understood by both development team and customer. A/B testing focuses on the end user and tests the usability of a feature.

Principles of agile testing²

Comparing agile with a waterfall approach, agile testing is a continuous process and not a sequential process. Agile testing is based on various principles to increase productivity.

Principle	Explanation
Constant response	Agile testing delivers a response or feedback on an ongoing basis. Therefore, the product can meet the business needs. In other words, the product and business requirements are understood throughout the constant response.
Less documentation	The execution of agile testing requires less documentation as the Agile teams or all the test engineers use a reusable specification or a checklist.
Continuous testing	The agile test engineers execute the testing endlessly as this is the only technique to make sure that the constant improvement of the product.
Customer satisfaction	In any project delivery, customer satisfaction is important as the customers are exposed to their product throughout the development process. As the development phase progresses, the customer can easily modify and update requirements. And the tests can also be changed as per the updated requirements.

¹ How to cite this article: Portman, H. (2023). Agile and testing, Sensemaking in the Agile Forest, series article, PM World Journal, Vol. XII, Issue III, March.

² <https://www.javatpoint.com/agile-testing>

Easy and clean code	Bugs or defects occurred by the agile team or the testing team are fixed in a similar iteration, which leads to easy and clean code.
Involvement of the entire team	The entire agile team with the business analysts and the developers to test the application or the software.
Test-Driven	Testing is not the final step in the development cycle but a continuous process. See the TDD and ATDD.
Quick feedback	In each iteration of agile testing, the business (team) is involved. Therefore, received continuous feedback helps to reduce the time of feedback response on development work.

Test-driven development (TDD)

TDD, developed by Kent Beck, is a tool for developers to develop well-written modules that perform a series of operations correctly. Whereas in familiar module and system testing, the software is developed first, in TDD the requirements are converted into test cases before the software is developed. The software is then modified until there is good working software. TDD requires an environment in which testing can be automated. TDD is also known as test first development (TFD) as described within Xtreme Programming (XP).

TDD development cycle

Kent Beck's book *Test-Driven Development by Example* describes the TDD development cycle in detail. It is briefly summarized in the following table.

Step	Name	Description
1	Add a test	Adding a new function starts by writing a test that passes if the specifications of the function are met.
2	Run all tests	The new test must fail for expected reasons. This shows that new code is needed for the desired function. It validates that the tests are working correctly. It rules out that the new test is flawed and will always pass.
3	Writing the simplest code	Inelegant or hard code is acceptable, as long as it passes the test. Regardless, the code will be tightened up in step 5. No code should be added beyond the tested functionality.
4	All tests must now pass	If any tests fail, the new code must be revised until it does pass. This ensures that the new code meets the test requirements and does not break any existing functionality.

5	Refactor if necessary	Code is refactored for readability and maintainability. Refactor includes tests after each refactor to ensure functionality is retained.
6	Repeat	The above cycle is repeated for each new piece of functionality. Tests should be small and incremental.

TDD and acceptance test driven development (ATDD)

TDD is related but different from acceptance test driven development (ATDD). ATDD is a means of communication between customer, developer, and tester to ensure that requirements are well defined. Tests used in TDD can often be derived from ATDD tests because the modules developed implement part of a requirement. ATDD tests need to be readable by the customer. TDD tests do not have to be.

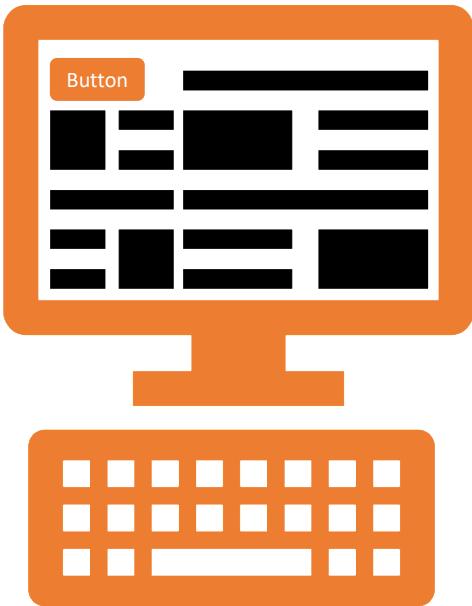
A/B testing

A/B testing is a research method for user experience. A/B testing consists of a randomized experiment that usually involves two variants (A and B). It involves the application of statistical hypothesis testing or "two-sample hypothesis testing" as used in statistics. A/B testing is a way of comparing multiple versions of a single variable, for example by testing a customer's response to variant A against variant B and determining which of the variants is more effective. A negative aspect of A/B testing is the extra programming to measure and report the conversion rate.

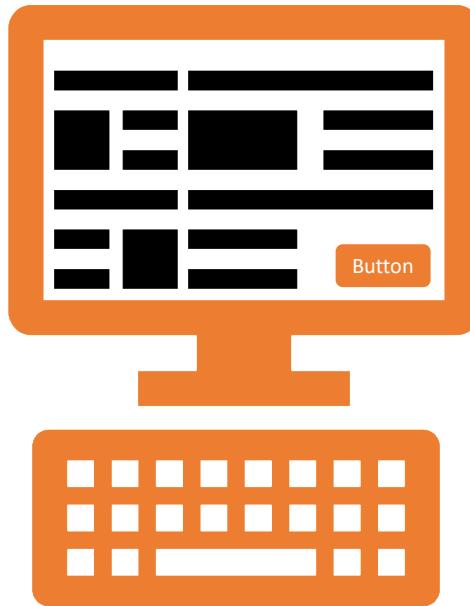
An example of A/B testing on a website

By randomly presenting visitors with two versions of a website that differ only in the design of a single button element, the relative effectiveness of the two designs can be measured. For example, visitors presented with situation A press the button in 60% of visits. Visitors presented with situation B press the button in only 35% of visits. The chosen solution represented by situation A is more successful. This is also known as the conversion rate.

Situation A



Situation B



Conversion rate 60%



Conversion rate 35%

Conclusion

Besides the already well-known tests such as unit, system and integration testing, new useful tests such as TDD, ATDD and A/B testing have emerged through agile development and are widely used.

Sensemaking in the Agile Forest series

This article is part of a series of articles called *Sensemaking in the Agile Forest*. This series³ consists of the following parts:

- [Portman, H. \(2022\). What is Agile? Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue I, January.](#)

³ This series is based on a number of short blogs I made for Forsa Advies, a project management training organization in the Netherlands (<https://www.forsa-advies.nl>).

- [Portman, H. \(2022\). What is Scrum? Sensemaking in the Agile Forest, series article 2, PM World Journal, Vol. XI, Issue II, February](#)
- [Portman, H. \(2022\). Is agile always better? Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue III, March](#)
- [Portman, H. \(2022\). The ideal Product Owner, Sensemaking in the Agile Forest series, PM World Journal, Vol. IX, Issue IV, April](#)
- [Portman, H. \(2022\). The Ideal Scrum Master, Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue V, May](#)
- [Portman, H. \(2022\). Is an agile team always autonomous? Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue VI, June](#)
- [Portman, H. \(2022\). What do iterative and incremental mean in Agile? Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue VII, July](#)
- [Portman, H. \(2022\). The Minimum Viable Product \(MVP\) unraveled; Sensemaking in the Agile Forest, series article, PM World Journal, Vol. XI, Issue VIII, August](#)
- [Portman, H. \(2022\). Prioritizing in an agile team, Sensemaking in the Agile Forest, series article, PM World Journal, Vol. XI, Issue IX, September](#)
- [Portman, H. \(2022\). Multitasking, task-switching or monotasking; Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue X, October](#)
- [Portman, H. \(2022\). Being predictable as an agile team; Sensemaking in the Agile Forest series, PM World Journal, Vol. XI, Issue XI, November](#)
- [Portman, H. \(2022\). Self-managing or self-organizing agile teams, Sensemaking in the Agile Forest series article, PM World Journal, Vol. XI, Issue XII, December](#)
- [Portman, H. \(2023\). Slicing user stories, Sensemaking in the Agile Forest series, PM World Journal, Vol. VII, Issue I, January](#)
- [Portman, H. \(2023\). Agile management products, Sensemaking in the Agile Forest series article, PM World Journal, Vol. VII, Issue II, February](#)
- Agile testing
- What is Kanban?
- Culture makes or breaks your agile transformation
- Why agility?
- Towards a more agile organization
- Getting started as an agile team (a pilot)
- Agile team of teams structures
- Agile centers of excellence (CoE)
- Knowledge sharing within and between agile teams
- The evolution of agile frameworks
- ?

Please let me know if you would like to add specific agile topics to this series.

About the Author



Henny Portman

The Netherlands



Henny Portman, owner of Portman PM[O] Consultancy and was partner of HWP Consulting, has 40 years of experience in the project management domain. He was the project management office (PMO) thought leader within NN Group and responsible for the introduction and application of the PMO methodologies (portfolio, program, and project management) across Europe and Asia. He trains, coaches, and directs (senior) programme, project and portfolio managers and project sponsors at all levels, and has built several professional (PM(O)) communities.

Henny Portman is/was accredited in a variety of qualifications, including P3O, PRINCE2, MSP, MoP, PRINCE2 Agile, AgilePM, AgilePgM and AgileSHIFT trainer and an SPC4 SAFe consultant and trainer. He is a P3M3 trainer and assessor and PMO Value Ring Certified Consultant (PMO Global Alliance). On behalf of IPMA, he assesses mega and large projects for the IPMA Project Excellence Award. In addition to this, he is an international speaker, author of many articles and books in the PM(O) field, and an active blogger (hennypotman.wordpress.com/).

Henny can be contacted at henny.portman@gmail.com.