Zephyr Enterprise

The Test Management Platform for Agile Teams

WHITEPAPER

ZEPHYR
Real-Time Test Management

Table of contents:

During a recent Agile Methodology conference, a speaker courageously admitted that their company had utterly failed in continually implementing Agile. This company had previously been practicing Agile for 4-5 years and since then their profits had been declining year after year.

The change had come about by the senior management of the company, who had started challenging the ideas of Agile. They were questioning that if the company was truly Agile then their products should be of good quality, meeting customers' requirements and responding to market changes. None of these were happening, but the teams were proud to proclaim that they were "Agile", since they were delivering regular stand ups, retros, showcases, implement TDD methodology and also because they had automated test cases. The pertinent question then is: *are they really agile?*

The agility does not lie in blindly following a set of practices but instead, customizing them whilst embracing the principles behind them.

Some of the key principles driving agility include:

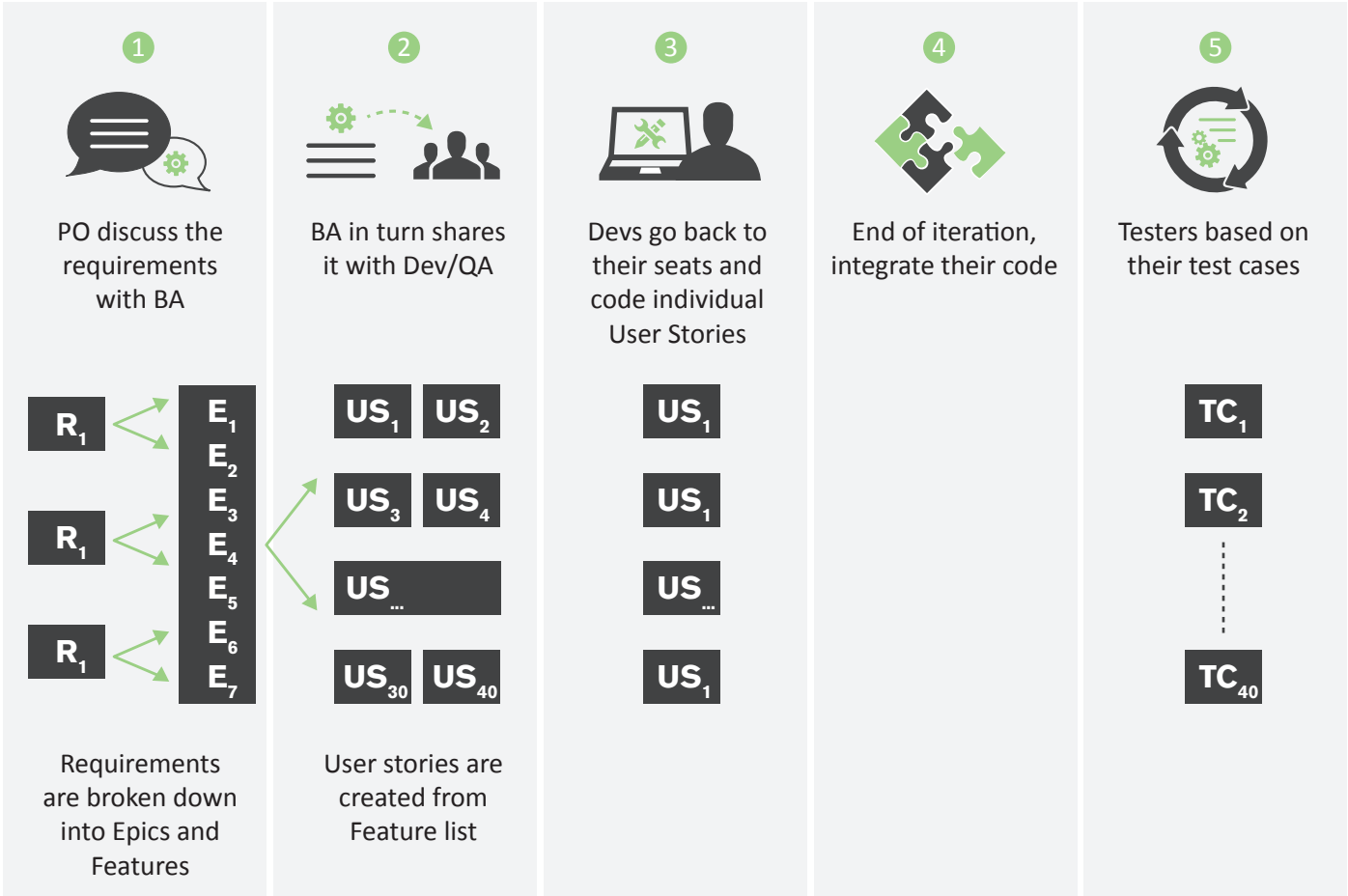
Let us look at each of the above principles and see the challenges with some solutions.

EVALUATE NOW

1 Collaborative culture

As noted by Cockburn (Cockburn, 2006), software development is a cooperative game of invention and communication. If we are developing software in isolation, without talking to each other (this is very common), we end up developing individual pieces of software rather than a product.

Let us take a look at the typical software process, as shown in the figure below. Each of the roles function independently, with the only time they meet being to clarify requirements, test cases or to attend Scrum ceremonies.



As one can see above, the Product Owner (PO) hands the requirements to the BA, who in turn sends a list of acceptance criteria to the developers. Testers, who are typically passive listeners in such teams, prepare lists of test cases based on what they hear in meetings.

This type of siloed working model leads to a misunderstanding of requirements, miscommunication and relinquishes the collective power of wisdom.

How can we improve one of the most fundamental aspects of software development?

1. One of the best ways to bring collaboration within the teams is through “3 Amigos” workshop (Hewitt, 2013). Ensure that we have the BA, Dev and QA available in every meeting. They work together in creating the Acceptance test documents, building test cases, deciding the boundary conditions.
2. Instead of the PO handing over the requirements to the BA, let the PO be available directly to the team.
3. Build Feature teams (Larman & Vodde, 2014) rather than allowing individual developers to work on stories. Each feature team could collaborate with others and participate in a “Scrum of Scrums” (Faria, 2013).
4. Teams collaborate when they have the freedom to think, chose and manage the situations. It is very important to build self-organizing teams (Cohn, 2014).

2 Building shared mental model

Language is one of the key barriers in communication. This is not a reference to the spoken language, but the semantics behind the words. We formulate a mental model of the world based experiences and context. The same words can easily be misinterpreted.

For example, if the requirement says:

"The price of gas should be $23."

Depending on their country of origin, each team member may interpret this differently. The word "gas" typically denotes "Petrol" or "Diesel" in the context of US or other western nations, however, in Asia, "gas" is typically used only in the context of "cooking gas." The differences in understanding could make a big difference to software development. The developers working on this requirement would make their own assumptions based on their interpretation of "gas" and in turn build the code.

You might remember how NASA lost $125 Million as the Mars orbiter crashed due to a metric mishap (Lloyd, 1999).

It is very important that the team members not only have a shared understanding of the goals but also the detailed requirements.

Thankfully, we have several tools and techniques available to improve shared understanding, such as "Given, when, then format" (Kruger, 2011) and Specification by Example (Anon., 2014).

The visual walls setup in a team room provides a clear view of the expected goals, timelines and the status.

3 Building quality products

Once the team has shared understanding of the requirements, one would say we have conquered 50% of the quality issues. The remaining half could be addressed using some of the techniques discussed below.

1. Having good technical practices is one of the best ways to conquer quality. The XP engineering practices are the best place to start (Norman, 2009).
2. Faster feedback loop is critical in catching the quality issues. Integrating and testing quite frequently, building the code on the main trunk is a good idea (Rasmusson, 2011).
3. Having right metrics and measurements in place enables the team to see the quality trend (Crispin, 2010).
4. Fail Fast approach: The sooner the team gets to see the defects, the better they are equipped to tackle uncertainties. Uncertainties are mostly due to lack of knowledge of the future, and by testing earlier and faster, one could eliminate the uncertainties. This could be done by reducing the sprint length, doing a daily build and automating the testing process. One should remember that automation alone will not improve the quality, but it helps in detecting the quality issues early.

4 Responding to market change

In order to quickly respond to market, the PO needs to have one ear dedicated outside of teams, thus enabling understanding of new customer demands and the competitor landscape.

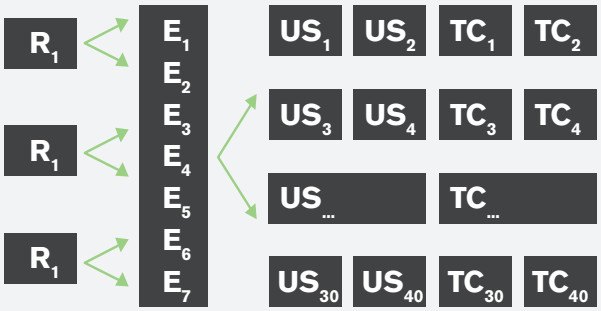
Responding to the market is critical for a company to improve the cycle time to go from concept to cash (Poppendieck & Poppendieck, 2006).

1. Cycle time could be improved by clearly articulating the WIP (Work In Progress) limits. The WIP limits need to be controlled at the enterprise level and not just at the team level.
2. Having a clear, prioritized set of requirements, and shorter delivery cycles will help in achieving this goal.
3. The backlog should be revisited frequently to remove any unnecessary items sitting for more than necessary. 30 days is a typical rule-of-thumb.
4. Automation is the key in responding to market change. Frequent integration and early detection of defects enables the developers to keep the code defect free.
5. Decentralized decision making: Bureaucratic decision making is one of the key obstacles for a nimble turn around. A decentralized decision making organizational structure is core of nimbleness.

In view of the above-proposed solutions, the new life cycle for an Agile team could be re-written as shown in the diagram below for improved agility.

1

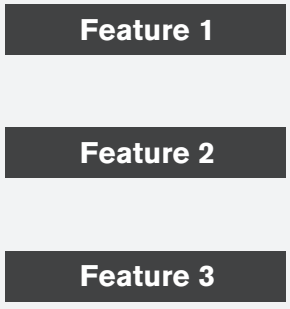
PO shares the requirements with the team instead of delegating this to BA



Team huddle to split Requirements into Epics and Features. The 3 Amigos help teams in creating Test cases in Given/When/Then format with Spec by example.

2

Feature teams work collaborativeky delivering features



Code is continuously integrated with automated testing. QA team work closely with feature teams providing them support and doing the final sanity check before going live.

3

Ability to **GO LIVE** with a short notice and thus reducing the concept to cash time

Reduced concept to cash time

As mentioned earlier, it is very important to have the right set of tools in place to improve the concept to cash time. The tools not only help in improving the turnaround time but improve quality and time to market. **Zephyr Enterprise** is one of the most popular test management tools to enable teams to realize true agility and reduce the "concept to cash" time.

Let us look at some of its features and its support for agility.

1 Improved Collaboration

Transparency and Visibility: Anyone in the feature team or across project can access the test cases, review them and provide the feedback to others. This visibility enables a tight collaboration among not only the 3 Amigos but with the rest of the team

a. The BA could review the test cases for correctness

b. The developers could access the test cases for clarification and testing purposes

c. The product owners can review the test cases to provide the feedback. As we know, in many Agile projects, POs are totally disconnected from the testing community. Zephyr fills this gap.

Pull Mechanism: Zephyr enables a pull mechanism of test cases. Since all the test cases are visible with their status, the testers could easily pick the open ones. This process is akin to supporting the Kanban's pull mechanism. Providing the autonomy enables the teams to self-organize and to facilitate improved collaboration.

2 Shared Mental model

Feature, User story to test case mapping: Because of its seamless integration with the Atlassian JIRA product, testers can easily pick a user story and collaboratively write test cases using the "Given/When/Then" format with Amigos.

The mapping between a JIRA story and Test cases allows the developers to see all the test cases for a particular feature or user story thus reducing the task switching effort. This process considerably reduces the miscommunication and improves the shared mental model across teams.

Real-time feedback: As soon as the testers update test case results, the changes are reflected on their JIRA wall as well. The developers get the instant feedback rather than waiting for testers and thus reducing the waste due to waiting (Ross, 2014).

3 Building Quality and responding to market change

Improved productivity and quality through Automation: Developers and testers need to collaborate and then deploy to production with a click of a button. Zephyr provides a robust software agent called a "ZBot", to empower the developers and testers to write their scripts and automate testing. Here are a couple of other ways of making effective use of ZBots.

a. As soon as the test cases are cleared, the tester could push a button to run the ZBot script which in turn could check the final code into the Pre-Prod environment, for instance. This script could be made to trigger continuous integration, running a wide gamut of tests.

b. Any tool that provides command line integration could be made to run through the ZBot. Some examples of tools include Selenium, Cucumber, QTP, ANT or Maven.

The ZBot and the integration with various automated tools help teams in responding to change, fail-fast and in building the inherent quality.

To conclude, every product company needs to keep an eye on bringing agility to improve the concept to cash time. It can only happen by investing in the right set of tools.

References

Anon., 2014. Wikipedia. [Online]
Available at: http://en.wikipedia.org/wiki/Specification_by_example
[Accessed 26 September 2014].

Cockburn, A., 2006. Agile Software Development: The Cooperative Game (2nd Edition). s.l.:Addison Wesley.

Cohn, M., 2014. http://www.mountaingoatsoftware.com. [Online]
Available at: http://www.mountaingoatsoftware.com/presentations/leading-a-self-organizing-team
[Accessed 26 September 2014].

Crispin, L., 2010. http://searchsoftwarequality.techtarget.com. [Online]
Available at:
http://searchsoftwarequality.techtarget.com/answer/What-key-test-metrics-should-be-tracked-for-Agile-teams
[Accessed 26 September 2014].

Faria, L., 2013. https://www.scrumalliance.org. [Online]
Available at:
https://www.scrumalliance.org/community/articles/2013/june/scrum-of-scrums-running-agile-on-large-projects
[Accessed 26 September 2014].

Hewitt, R. T., 2013. http://rthewitt.com. [Online]
Available at: http://rthewitt.com/2013/02/07/the-3-amigos-ba-qa-and-developer/
[Accessed 26 September 2014].

Kruger, J., 2011. http://jonkruger.com. [Online]
Available at: http://jonkruger.com/blog/2011/10/24/know-what-youre-building/
[Accessed 26 September 2014].

Larman, C. & Vodde, B., 2014. http://featureteamprimer.org/. [Online]
Available at: http://featureteamprimer.org/
[Accessed 2014].

Lloyd, R., 1999. CNN. [Online]
Available at: http://edition.cnn.com/TECH/space/9909/30/mars.metric.02/
[Accessed 26 September 2014].

Norman, T., 2009. Blogspot. [Online]
Available at: http://tommynorman.blogspot.com.au/2009/01/comfortably-scrum-man-cannot-live-on.html
[Accessed 26 September 2014].

Poppendieck, M. & Poppendieck, T., 2006. Infoq. [Online]
Available at: http://www.infoq.com/articles/poppendieck-implementing-lean
[Accessed 26 September 2014].

Rasmusson, J., 2011. Agile Warrior. [Online]
Available at: http://agilewarrior.wordpress.com/2011/05/28/how-facebook-pushes-new-code-live/
[Accessed 26 September 2014].

Ross, M., 2014. Lean Genie. [Online]
Available at: http://leangenie.com/7-wastes-wait-time/
[Accessed 26 September 2014].

About Zephyr

Zephyr is a leading provider of quality management solutions, powering intelligent DevTestOps for more than 11,000 global customers across 100 countries. Project teams and enterprises of all sizes use Zephyr's products to enable continuous testing throughout their entire software delivery pipeline to release higher quality software, faster. Zephyr is headquartered in San Jose, CA with offices in King of Prussia, PA, Europe and India. For more information, please visit www.getzephyr.com.

ZEPHYR
Real-Time Test Management

Contact Zephyr Today!
sales@getzephyr.com
www.getzephyr.com
+1-510-400-8656