Agile Startups

Comparision of Agile Methodololgy with the business canvas model

Semester Work

Johannes Eifert, Jean-Daniel Mathieu,
Oleg Telegin, Daria XYZ

Mars 2013

Thesis supervisors:

Dr. Jean Hennebert
University of Fribourg, Switzerland

UNIVERSITÉ DE FRIBOURG SUISSE
UNIVERSITÄT FREIBURG SCHWEIZ

Software Engineering Group
Department of Informatics
University of Fribourg (Switzerland)

1
Abstract

The following work is a comparison of the theories about the business canvas model and the agile methdology.

2
Introduction to Agile

This chapter gives an overview over agile software development processes in general.

In the traditional software development there are many people working on the requirements of the future software. When this is done, the developers start to work on it. During this phase there is usually no feedback from either the product owner or future users. The developers work more or less on their own which usually results in a disaster. (Quelle: http://www.galorath.com/wp/software-project-failure-costs-billions-better-estimation-planning-can-help.php) According to blabla only 32 percent of the IT projects in were successful. This needs to be changed and agile methodologies are trying to do that by trying to produce something valuable instead of over-planning.

2.1 The Agile Processes

The agile software development processes work in iterations. These are repeated at least until the software is shippable. A very important aspect of the process is the fact that after each iteration there should be an increment of the project which has some value and would be shippable.

2.1.1 Principles

2.1.2 Limits

According to [3] the usage of the agile software development processes brings a few drawbacks and limitations:

1. Limited support for distributed development environments.
2. Limited support for subcontracting.
3. Limited support for building reusable artefacts.
4. Limited support for development involving large teams.
5. Limited support for developing large, complex software.
6. Limited support for developing safety-critical software.

In chapter blabla we discuss if these limitations also apply to our agile start up process. [2]

3 Scrum

3.1 Introduction to Scrum

Scrum is originally a tactic used in rugby. All players are gathered around the ball, holding each other and trying to be the first team to get the ball. Scrum is applied when an error was made or a ball was put out of game. The scrum methodology is an applied software team management philosophy following the same principle. It is agile and therefore an ideal example for an applied agile methodology.

Scrum not only tries to avoid errors like the "tunnel development", "contract gaming", "squirrel burgers" or working with the "magic toolbox" but it also enforces a game like style in design and implementation of Software. A Scrum team is considered very transparent and flexible, giving always clear information about the current state of the development.

Main goal is to generated measurable value increments after each step in the development of software, by minimizing the time between each working increment of the software.

3.2 Scrum Principles and elements

3.2.1 Radiator, stories, backlog and sprints

The center piece of scrum is a big chart showing the overview of all currently developed features. This chart is called the radiator while the features are called stories. A story is comparable to a specific feature. Each described as a use-case and containing an estimation in time or points for the completion. Those information are written on a post-it and put on the radiator.

The radiator does not only show the current stories but all stories, whether they are completed, in progress or in the backlog. The radiator also contains a chart summing up the current points in progress, the points already used and the remaining points. This chart is called a burn-down diagram. This gives an always-up-to-date overview of the current state of the project and is visible to anyone working on the project.

Another element are the sprints. They are time frames set to finish the next set of stories. Stories assigned to a project are stored in the sprint-backlog. All stories assigned to the team are stored in the product backlog.

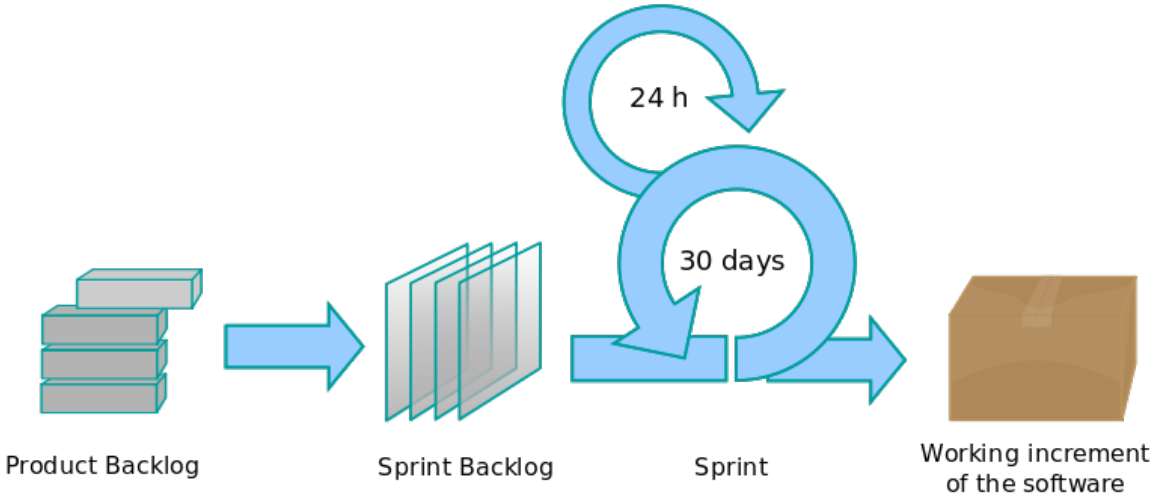

Figure 3.1: Illustration of the scrum process

3.2.2 Roles

product-owner

The most important role in Scrum is the product-owner. He is responsible to present new stories and priories them. He is the manager behind the backlog and directs the team to fulfill the goals set by the stakeholders.

scrum-master

Takes over the role of a mentor and guide to apply the scrum rules. He controls the teams and leads discussion and meetings. He is not to be confused with the product-owner as a scrum-master does not manage the team, but only leads it. He can be compared to the coach. A Scrum master protects the teams from any influence and helps focusing on the tasks ahead.

developer-team

Of course every team needs grunts for the work. Developer take over the role as designer and developer. They follow instructions and are organized as teams. Each team focusing on the same stories.

stakeholders

Stakeholders are the customers. They provide feedback to the product-owner and provide the income. They are only involved in the sprint meetings.

management

The management has no special function but to provide stakeholders (Marketing) and deliver a work environment for the developer teams (Facility Manager, HR Manager).

3.2.3 Meetings

Daily Scrums

Daily meeting used to check on work progress. Therefore it is discussed what changed since last meeting and what is up ahead. Those meetings are timeboxed to 15 minutes.

Backlog grooming

A meeting done during a sprint. The current stories are updated and new stories are optimized using techniques like planning poker. After such a meeting the backlog is re estimated providing sharper estimation of project state.

Scrum of Scrums

Meeting of team or clusters of teams to discuss overlapping stories or common work. Usually after a daily scrum.

Sprint planning meeting

Meeting before the start of a sprint to fix the backlog.

End of cycle

Meeting at the end of a sprint to discuss the sprint in retrospective. This can be seen as an after action review or a debriefing.

References

[1] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. Manifesto for agile software development. 2001.

[2] Linda Rising and N.S. Janoff. The scrum software development process for small teams. *Software, IEEE*, 17(4):26–32, 2000. 2

[3] Dan Turk, Robert France, and Bernhard Rumpe. Limitations of agile software processes. In *Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP 2002)*, pages 43–46, 2002. 2