

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA - SCUOLA DI SCIENZE
Corso di Laurea in Scienze e Tecnologie Informatiche

THE SCRUM APPROACH TO SOFTWARE DEVELOPMENT

Relazione finale in Progettazione di Sistemi Informatici

Relatore:
ANTONIO NATALI

Presentata da:
GIACOMO SERVADEI

Terza sessione
Anno Accademico 2013/2014

ABSTRACT

Agile methodologies have become the standard approach to software development. The most popular and used one is Scrum. Scrum is a very simple and flexible framework that respond to unpredictability in a really effective way. However, his implementation must be correct, and since Scrum tells you what to do but not how to do it, this is not trivial. In this thesis I will describe the Scrum Framework, how to implement it and a tool that can help to do this. The thesis is divided into three parts.

The first part is called Scrum. Here I will introduce the framework itself, its key concepts and its components. In Scrum there are three components: roles, meetings and artifacts. Each of these is meant to accomplish a series of specific tasks. After describing the “what to do”, in the second part, Best Practices, I will focus on the “how to do it”. For example, how to decide which items should be included in the next sprint, how to estimate tasks, and how should the team workspace be. Finally, in the third part called Tools, I will introduce Visual Studio Online, a cloud service from Microsoft that offers Git and TFVC repositories and the opportunity to manage projects with Scrum.

SOMMARIO

I metodi Agile sono diventati l'approccio standard per lo sviluppo di software. Il più famoso ed utilizzato è Scrum. Scrum è un framework molto semplice e flessibile che risponde ai cambiamenti in una maniera molto efficace. La sua implementazione deve però essere corretta, e visto che Scrum ci dice cosa fare ma non come farlo, questo non risulta essere immediato. In questa tesi descriverò Scrum, come implementarlo ed uno strumento che ci può aiutare a farlo. La tesi è divisa in tre parti.

La prima parte è chiamata Scrum. Qui introdurrò il framework, i suoi concetti base e le sue componenti. In Scrum ci sono tre componenti: i ruoli, i meeting e gli artifact. Ognuno di questi è studiato per svolgere una serie di compiti specifici. Dopo aver descritto il "cosa fare", nella seconda parte, Best Practices, mi concentrerò sul "come farlo". Ad esempio, come decidere quali oggetti includere nella prossima sprint, come stimare ogni task e come dovrebbe essere il luogo di lavoro del team. Infine, nella terza parte chiamata Tools, introdurrò Visual Studio Online, un servizio cloud della Microsoft che offre repository Git e TFVC e l'opportunità di gestire un progetto con Scrum.

CONTENTS

<i>Abstract</i>	i
<i>Sommario</i>	iii
<i>Part I Scrum</i>	1
1. <i>What is Scrum</i>	2
1.1 Introduction	2
1.1.1 Scrum definition	2
1.1.2 Difficulties	3
1.1.3 A great advantage	3
1.2 History	3
1.2.1 Origin	3
1.2.2 First implementation	4
1.2.3 World-wide spreading	4
1.3 Agile Manifesto	5
1.4 Empirical Process Control	6
1.4.1 Transparency	6
1.4.2 Inspection	6
1.4.3 Adaptation	6
1.5 Scrum Process	6
1.5.1 Sprints	6
1.5.2 Meetings	7
2. <i>Roles</i>	8
2.1 Product Owner	8
2.2 ScrumMaster	9
2.3 Team	10
2.4 Project Manager?	11
2.5 Stakeholders	11

3. Meetings	12
3.1 Sprint Planning Meeting	12
3.1.1 What to do	12
3.1.2 How to do it	13
3.2 Daily Scrum Meeting	13
3.3 Sprint Review Meeting	14
3.4 Sprint Retrospective Meeting	15
4. Artifacts	17
4.1 Product Backlog	17
4.2 Sprint Backlog	19
4.3 Sprint Burndown Chart	20
<i>Part II Best Practices</i>	21
5. Introduction	22
6. Product Backlog and Sprint Planning	23
6.1 Product Backlog structure	23
6.2 Sprint Planning Preparation	24
6.3 The actual Sprint Planning Meeting	25
6.4 Avoid misunderstanding	26
6.5 Which items to include in the sprint	26
6.6 Planning Poker	28
7. Sprint and Daily Scrum	30
7.1 Sprint Backlog	30
7.2 Burndown Chart	31
7.3 Team workspace	32
7.4 Daily Scrum	33
8. End of Sprint	34
8.1 Sprint Review	34
8.2 Sprint Retrospective	35
9. Failures	37
9.1 Resistance to reorganisation	37
9.2 Blaming the process	37
9.3 No or bad retrospectives	37
9.4 Meetings are taking too long	38

<i>Part III Tools</i>	39
<i>10. Visual Studio Online</i>	40
10.1 Getting Started	40
10.2 Backlog Management	41
10.3 Sprint and Release Management	41
<i>Bibliography</i>	49

Part I

SCRUM

1. WHAT IS SCRUM

1.1 Introduction

1.1.1 Scrum definition

Scrum is an iterative and incremental agile software development framework that has become quite famous and widely spread used in software development. For completeness it has to be said that Scrum can be applied to any product being developed which requires intellectual work. In this thesis, though, I will focus on its application to software development.

As stated in the last paragraph, Scrum is a framework. That means: it tells you exactly what to do, but not how to do it. Quoting the official guide written by Ken Schwaber and Jeff Sutherland (the first two people who implemented Scrum):

“Scrum is a process framework that has been used to manage complex product development since the early 1990s. Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve.” [1]

Always according to the guide, Scrum is lightweight, simple to understand and difficult to master. These three words describe perfectly the strength and the weakness of it. The framework is very easy to understand, but the key point is how to implement it correctly. This is the reason why one of the figure required for the implementation is the Scrum Master. He or she is basically a person who is responsible for making sure that everyone understand it. I will describe him more deeply in the next chapter.

1.1.2 Difficulties

There are mainly two reasons why teams may find it difficult to implement. The first comes from the new vision that Scrum has on the role of people in the development process. While the old idea was to leave the development process as independent as possible to the people involved, Scrum focus on them. They are the pillar of the development process. Therefore, a lot of human interaction and discussion is required. This seems pretty easy but, especially among software developers, it can be really hard to achieve. It is also a great mindset change, and has to be understood completely.

The other difficulty is related to the problems that may arise during the development. Scrum, other than a framework, is also a great tool for diagnosing problems. If there is something not working properly in the process, be sure that it is going to let you know. The difficulty is then how to solve the problem. The framework, of course, does not tell you how to do it. Whenever a problem arises, there are two option: recognize and face it, or blame the methodology you are following, ignoring what is really in front of you. [2]

1.1.3 A great advantage

One of the greatest advantage of Scrum that makes it really worth to use, is the way it handles change. This is one of the biggest problems in waterfall processes. According to the framework, changes are natural in software development, it means that the product is being used. The Product Owner (2.1) with the Product Backlog (4.1) will manage this task. The Product Backlog is meant to change a lot during the development process in order to meet the requirements that will surely change.

1.2 History

1.2.1 Origin

Nowadays, Scrum is one of the most popular frameworks used in software development. In the agile world, it has been on of the greatest invention. As every worldwide spread technology, it has not been immune to controversy. The history of Scrum is a topic of frequent debate. [3]

“The traditional sequential or “relay race” approach to product development...may conflict with the goals of maximum speed and

flexibility. Instead, a holistic or “rugby” approach—where a team tries to go the distance as a unit, passing the ball back and forth—may better serve today’s competitive requirements”
(Hirotaka Takeuchi, Ikujiro Nonaka)[4]

Despite what people may believe, papers talk themselves. In 1986 Hirotaka Takeuchi and Ikujiro Nonaka described a new approach to product development, which led to what we now know as Scrum. They came up with the initial concept, including the word Scrum itself. This made them the fathers of this technology.

The word Scrum is not an acronym. It derives from the rugby term which describes the whole team that step by step advances towards the try. This is of course a metaphor that helps us understand the basic idea of Scrum: a development team which works together as a single entity whose final goal is the fulfillment of the product.

1.2.2 First implementation

In the early 90’s, Jeff Sutherland and Ken Schwaber, applied the concept in their companies and fine-tuned it. They were the first to refer to it with the name Scrum. In 1995, they jointly presented a paper describing the Scrum methodology at the OOPSLA ’95 conference in Austin, Texas. During the following years Sutherland and Schwaber collaborated to merge their experiences and industry best practices into what is now known as Scrum. [5]

1.2.3 World-wide spreading

In 2001, Schwaber and Beedle attempted to communicate Scrum through the first Scrum book Agile Software Development with Scrum. Respectively in 2002 and 2006, Ken Schwaber founded the Scrum Alliance and Jeff Sutherland created his own company Scrum.inc. Then, in the fall of 2009, Ken left the Scrum Alliance and founded Scrum.org. With the first publication of the Scrum Guide in 2010, and its incremental updates in 2011 and 2013, Jeff and Ken established the globally recognized body of knowledge of Scrum.[6]

1.3 Agile Manifesto

In order to better understand Scrum is good to know the pillars of it. As we mentioned before, by definition, Scrum is an Agile framework. It follows the principles of The Agile Manifesto. This is a document published in February 2001, by 17 software developers. It is composed of a summary with the main ideas, and then 12 principles. Here is the summary:

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

*Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan*

That is, while there is value in the items on the right, we value the items on the left more.” [7]

The ideas are pretty self-explanatory but in order to understand them completely I am going to discuss them a little.

Individuals and interactions: in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

Working software: working software is more useful and welcome than just presenting documents to clients in meetings.

Customer collaboration: requirements cannot be fully collected at the beginning of the software development cycle, therefore continuous customer or stakeholder involvement is very important.

Responding to change: agile development is focused on quick responses to change and continuous development [8]

1.4 Empirical Process Control

Scrum emphasizes the idea of Empirical Process Control. Knowledge comes from experience, so according to Scrum decisions have to be made based on observation and experimentation. The Empirical Process Control relies on three main idea: transparency, inspection and adaptation. [9] [1]

1.4.1 Transparency

Every aspect of the process must be visible and so observed by anyone responsible for the product. In *Scrum* transparency is provided, for example, by openly sharing the Product Backlog and the Burn Down Chart. It is also provided by daily stand-up meeting, where Team members share what has been done during the day and any difficulties.

1.4.2 Inspection

The overall process and the artifacts must be frequently inspected by users and stakeholders. They should give feedbacks to the team also on the features already implemented and the process towards the goal. Inspection should not be too frequent because they could impede the normal flow of the process. Inspections are most beneficial when performed diligently.

1.4.3 Adaptation

Adaptation happens as the Team and stakeholders learn through transparency and inspection and then adapt by making adjustments as soon as possible.

1.5 Scrum Process

1.5.1 Sprints

The Scrum process is structured in cycles of work called Sprints. These iterations last for a fixed period of 2 to 4 weeks and take place one after each other without pause. The Sprints cannot be extended. They will end on the specified date whether the work has been done or not. The length is decided by the Team and should not change between Sprints.

During each Sprint, the Team agrees on a set of features that are going to be implemented before the next one. The items are picked from a prioritized list

called the *Product Backlog*. The *Product Owner* takes care of the *Product Backlog*. He is responsible for keeping it updated with all the features that meet the requirement of the product. The set of picked functionalities is called *Sprint Backlog*. This is the goal to achieve before the next *Sprint*. The meeting in which the *Team* and the *Product Owner* agrees on the *Goal* is called *Planning Meeting*, and it is held at the beginning of each *Sprint*. *Sprint Backlogs* cannot be modified. It is not possible to add, remove or change the chosen items. Scrum welcomes change but not during the current *Sprint*. Thus, every change can and has to be made during the next one.

1.5.2 Meetings

Meetings, as part of the human interaction, are really important. Every day the team gathers briefly in what is called *Daily Meeting*. This is a 15 minutes long meeting in which each member share with the team his task status and raises a flag in case there is any difficulty. If there is, the *Scrum Master* will do his best to solve it.



Fig. 1.1: Classical representation of the *Scrum* process.

At the end of each *Sprint*, the *Team* reviews what has been done in the *Review Meeting*. The features implemented are shown to the *Product Owner* and the stakeholders, who have to approve or not the work. During this meeting the team gets feedback that can be incorporated in the next *Sprint*. Finally, *Scrum Master* and *Product Owner* meet and discuss on what can be improved in the overall process.

2. ROLES

In the first chapter, I have introduced some figures who appear in the Scrum process without giving enough details. In this second chapter I will introduce and describe deeply each one of them. In Scrum there are three roles: Product Owner, ScrumMaster and the Scrum Team.

2.1 Product Owner

The Product Owner is the responsible for maximizing the return on investment (ROI) and the work of the Team. In order to do so, he or she has to identify the features required by the client. In a perfect world, the Product Owner and the client are the same person.

Although in real life it is almost impossible for the Product Owner and the client to be the same person, the Product Owner must be fully aware of all the features to be developed. He is the voice of the client.

The Product Owner is the sole person responsible for managing the Product Backlog. This includes: identify and prioritize each feature, make it visible and accessible for every person responsible for the product and to be sure that the Team understand to the level needed every item of it. [1]

Together with the Team, the Product Owner decides and agrees on the items to be implemented in the next Sprint. At the end of each Sprint, during the Review Meeting, he has to approve or reject what has been implemented. He can also give suggestions for the subsequent sprints. [2]

It is important to know that in Scrum there is one and only one person who serves as Product Owner.

The entire organization responsible for the product, must respect his decisions. Nobody else can ever tell the team what to work on, and the Team cannot act on what anyone else says.

2.2 ScrumMaster

The role with the most subtle task to understand is the ScrumMaster. He is responsible for making sure that each one involved in the development process, fully understand and follow the Scrum framework. He is also responsible to fix all the problems that may arise during the process.

The ScrumMaster is a coach and a teacher. He has to explain, and most importantly, convince the team to follow the Scrum process, without having any authority over them. Of course, the Team knows that they have to follow the framework, but to do so accordingly they have to follow the ScrumMaster leadership. The ScrumMaster helps also the Team improve the development process, in the Review Meetings. [2]

Since Scrum makes visible many impediments and threats to the Team's and Product Owner's effectiveness, there should be a dedicated full-time ScrumMaster, who will work as hard as he can on resolving these issues. If there is not, the Team or the Product Owner will find it difficult to succeed. [10]

Resolving the impediments is not an easy task. The ScrumMaster has to sort out any issue with the client, with third-party-providers, other teams or even with managers. Thus, he would have to be quite skilled in order to know how to deal with this variety of areas effectively.

His work aims to support both the Team and the Product Owner.

“The Scrum Master serves the Product Owner in several ways, including:

- *Finding techniques for effective Product Backlog management;*
- *Helping the Scrum Team understand the need for clear and concise Product Backlog items;*
- *Understanding product planning in an empirical environment;*

- Ensuring the Product Owner knows how to arrange the Product Backlog to maximize value;
- Understanding and practising agility; and,
- Facilitating Scrum events as requested or needed.

The Scrum Master serves the Development Team in several ways, including:

- Coaching the Development Team in self-organization and cross-functionality;
- Helping the Development Team to create high-value products;
- Removing impediments to the Development Teams progress;
- Facilitating Scrum events as requested or needed; and,
- Coaching the Development Team in organizational environments in which Scrum is not yet fully adopted and understood.”

[1]

2.3 Team

The Team is the set of people who develops the product. They have to implement what the Product Owner decides.

In Scrum the Team is “cross-functional” and “self-organizing”. “Cross-functional” means that the Team has all the skills necessary in order to build the features required. “Self-organizing” means that nobody will ever tell the team how to implement the features, not even the ScrumMaster. The Team decides how many items (from the Product Backlog) to build in a Sprint. [10]

An important rule regarding the Team is that each member of the Team is recognized as a Developer. There are no fixed specialist titles in a group that adopts Scrum. They work together in whatever way is appropriate to achieve the goal that they have set for the current Sprint. It is obvious that each person has special strengths but everyone is meant to keep learning other specialities.

“Optimal Development Team size is small enough to remain nimble and large enough to complete significant work within a Sprint.” [1]

According to the “Scrum Official Guide” by Schwaber and Sutherland, the Team should have between 3 to 9 people. The Product Owner and the ScrumMaster are not part of the count. The Team might include people with skills in analysis, development, testing, interface design, database design, architecture, documentation, and so on.

To maximize productivity and effectiveness, Teams in Scrum should be 100 percent dedicated to the work for one product during the Sprint. Another aspect to take into account is that stable teams have higher results so it is suggested to avoid changing members.

It is useful to note how the Product Owner and the ScrumMaster fit into the Team Purpose. The first feeds information to the Team about the product being developed, while the second leads the Team to follow the Scrum framework as close as possible to its principles. [2]

2.4 Project Manager?

In Scrum there is no role of project manager at all. The reason for it is that none is needed. The traditional responsibilities of a project manager have been divided and reassigned among the three Scrum roles. Implementing Scrum with the addition of a project manager indicates a fundamental misunderstanding of Scrum. [10]

2.5 Stakeholders

In addition to the three main roles (Product Owner, Team, ScrumMaster) there are other stakeholders who contribute to the success of the product, including managers, customers and end-users. In Scrum, these individuals replace the time they previously spent playing the role of “nanny” (assigning tasks, getting status reports, and other forms of micromanagement) with time as “guru” and “servant” of the Team (mentoring, coaching, helping remove obstacles, helping problem-solve).

3. MEETINGS

As I said since the introduction, Scrum strongly relies on human interaction. Therefore, the framework provides some meetings during which the people responsible for the development process will have the opportunity to share opinions and show results.

3.1 Sprint Planning Meeting

The first meeting I am going to describe is the Sprint Planning Meeting. This takes place at the beginning of each Sprint, and it is meant to plan the work that is going to be done. It is time-boxed to a maximum of 8 hours for a one-month Sprint. It is generally divided into two parts: What to do and How to do it. Product Owner, Team and ScrumMaster have to participate to the meeting.

3.1.1 What to do

In the first part of the Sprint Planning, Product Owner and Team review the high-priority items in the Product Backlog. These are the items that the Product Owner, and so the client, are interested in implementing this Sprint. The number of features to be implemented is decided by the Team. Usually, these items will have been well analyzed in a previous Sprint, so that at this meeting there are only minor last-minute clarifying questions.

The Team, then, will agree on a Sprint Goal. This is a summary statement of the Sprint objective. It also gives the Team some flexibility on what it has to be done. This means, since Sprints are time boxed, some items may be dropped if they would cause the end of the Sprint to be postponed. Anyway, the team should always release something tangible and “done” that is in the spirit of the Sprint Goal.

At the end the first part, the Product Owner can leave since the “How to” part is planned only by the Team. Although, he must be available for any

clarification.

3.1.2 How to do it

The second part of the Sprint Planning focuses on how to implement the items that the Team decides to take on. The Team forecasts the amount of items they can complete by the end of the Sprint, starting at the top of the Product Backlog (high-priority items) and working down the list in order. This is a key practice in Scrum: The Team decides how much work it will complete, rather than having assigned to them by the Product Owner. This makes for a more reliable forecast because the Team is making it based on its own analysis and planning. [10]

After estimates determined, the Team breaks down the activities in tasks which can be assignable to developers. Usually one developer is responsible for one task, but there is no problem in more than one developer to be involved in a single task. The idea here is for the whole time to be aware of what is to be developed, and most importantly, how it is going to be developed. In theory, any member of the Team could develop any of the discussed tasks. [10]

By the end of the Sprint Planning, the Team should be able to explain to the Product Owner and Scrum Master how it intends to work as a self-organizing team to accomplish the Sprint Goal.

3.2 Daily Scrum Meeting

The Daily Scrum Meeting, as the name suggests, is a short gathering which happens every day and should not be longer than 15 minutes. It is held always at the same time and in the same place to reduce complexity. It is recommended that nobody take a seat so that the meeting will be concise. The goal of this gathering is to synchronize activities and create a plan for the next 24 hours.

In the Daily Scrum, one by one, each member of the Team takes the floor and answers three question:

- “What did I do yesterday that helped the Development Team meet the Sprint Goal?”

- What will I do today to help the Development Team meet the Sprint Goal?
- Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?”

[1]

It is important to understand that this is not a status reporting to report to a manager. It is a time for self-organizing Team to share with each other what is going on, to help them coordinate. Therefore, it is recommended not to have managers attend the Daily Scrum. This risks making the Team feel “monitored” and it tends to undermine the Team’s self-management and invite micromanagement.

As I said, the Team is responsible for conducting the Daily Scrum, and the ScrumMaster has to ensure this. He has to teach the Team to keep it within the 15-minute time-box. If it takes more, it means that is not being conducted correctly. This could be because other things are being discussed which are out of scope, there are too many members in the Team, people are talking trivialities, and so on.

Since the theme is just reporting answers for the three questions, there should not be in-depth discussion during the Daily Scrum. If discussion is required it takes place immediately after it in one or more parallel follow-up meeting, although in Scrum no one is required to attend these. A follow-up meeting is a common event where some or all team members adapt to the information they heard in the Daily Scrum.

3.3 Sprint Review Meeting

At the end of each Sprint, two meetings are held, the Sprint Review Meeting and the Sprint Retrospective Meeting. A key idea in Scrum is to inspect and adapt (1.4). Both of these two gathering are meant to do so. The first involves inspect and adapt regarding the product increment of functionality; the second, instead, regarding the process and environment.

The Sprint Review, often mislabelled “demo”, is a meeting at which all the people related to the product take place; these people are: Product Owner,

Team members, ScrumMaster, customers, users, stakeholders, experts, executives and anyone else who is interested. It should not be longer than four hours for a one-month Sprint.

During the Sprint Review the Team presents to the Product Owner what was developed, via a demo. The Product Owner, then, evaluates whether he or she approves or rejects what was developed. Note that he can approve the product even if it has bugs. As long as the product is something deliverable, the Product Owner can accept it and leave corrections for later Sprints. Therefore, a critical element of the Review is an in-depth conversation between the Team and the Product Owner.

The “live software” portion of the Sprint Review is not a “presentation” the Team gives. It is meant to be a hands-on inspection of the real software running live, for example, in a sandbox development environment. There will be one or more computers in the Review room on which people can inspect and use the live software. Prefer an active session in which real users and the Product Owner do hands-on interaction with the software, rather than a passive-session demo from the Team

Note that Sprints may fail, either because the Product Owner does not like what was shown or because the estimations were wrong and the time for the development of the agreed items was not enough. If this happens the Team must be aware of the failure and make its best to improve in the next Sprint.

The result of the Sprint Review is a revised Product Backlog that defines the probable Product Backlog items for the next Sprint. The Product Backlog may also be adjusted overall to meet new opportunities.

3.4 Sprint Retrospective Meeting

The final official meeting I am going to describe is the Sprint Retrospective Meeting. It is three-hours time-boxed for a one-month Sprint. This is where heated discussions sometimes take place.

“The purpose of the Sprint Retrospective is to:

- Inspect how the last Sprint went with regards to people, relationships, process, and tools;

- Identify and order the major items that went well and potential improvements; and,
- Create a plan for implementing improvements to the way the Scrum Team does its work.”

[1]

Here lays the basis for continuous improvement. If this meeting is successfully executed, the Team will always be improving its development process. If they have in mind that there are always things to be improved, this meeting is a never-ending need; although perfection is impossible, significant improvements will be achieved throughout the product development.

Many Teams hold retrospectives only focusing on problems, and that is too bad. It can lead to people thinking of retrospectives as somewhat depressing or negative events. Instead, the ScrumMaster must ensure that the meeting also focus on positives or strengths.

4. ARTIFACTS

The third and last component of the Scrum framework, other than roles and meetings, are Artifacts. Scrum provide a three Artifacts that are meant to be used along the development process. They represent work or value to provide transparency and opportunities for inspection and adaptation. They are specifically designed to maximize transparency of key information so that everybody has the same understanding of the artifact.

4.1 Product Backlog

The first Artifact I am going to describe is the Product Backlog. I the previous chapters I already mentioned it more than once. So far it is clear that the Product Backlog is a prioritized list of customer-centric features. It is managed by the Product Owner and the Team picks features from it. But let's dig deeper into it.

It must be said that only one Product Backlog exists for a product, and as long as the product exists, its Product Backlog also exists. It is dynamic, it keeps evolving over the lifetime of the product in order to always meet the customer requirements; it constantly changes to identify what the product needs to be appropriate, competitive, and useful. At any point, the Product Backlog is the single, definitive view of “everything that could be done by the Team ever, in order of priority”.

The Product Backlog contains features (“allow the user to remove items from the cart”), but it also contains enhancements (“make the fingerprint recognition faster”), research work (“investigate solutions for bring the software on Ubuntu”) or even bugs and related fixes (“fix occasional error message during successful login”). Product Backlog's entries are simply called items. Items have a description, an order, an estimate and a value.

“As a product is used and gains value, and the marketplace provides feedback, the Product Backlog becomes a larger and more

exhaustive list. Requirements never stop changing, so a Product Backlog is a living artifact. Changes in business requirements, market conditions, or technology may cause changes in the Product Backlog” [1]

	Item #	Description	Est	By
Very High				
	1	Finish database versioning	16	KH
	2	Get rid of unneeded shared Java in database	8	KH
		- Add licensing	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		Analysis Manager		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
High				
		- Enforce unique names	-	-
	7	In main application	24	KH
	8	In import	24	AM
		- Admin Program	-	-
	9	Delete users	4	JM
		- Analysis Manager	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
		- Query	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
		- Population Genetics	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	Add icons for v1.1 or 2.0	-	-
		- Pedigree Manager	-	-
	20	Validate Derived kindred	4	KH
Medium				
		- Explorer	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A

Fig. 4.1: An example of a Product Backlog

A good Product Backlog is DEEP [10]:

- Detailed appropriately. Higher ordered items are usually clearer and more detailed than the lower ones; the lower the order, the less detail.

- Estimated. Items need to have estimates, and should be re-estimated each Sprint as new information arises and everyone learns.
- Emergent. The Product Backlog is dynamic. Each Sprint, items are added, removed and modified.
- Prioritized. All the items have to be ordered from 1 to N with the most valuable items at the top.

4.2 Sprint Backlog

User Story	Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	...
As a member, I can read profiles of other members so that I can find someone to date.	Code the ...	8	4	8	0		
	Design the ...	16	12	10	4		
	Meet with Mary about ...	8	16	16	11		
	Design the UI	12	6	0	0		
	Automate tests ...	4	4	1	0		
	Code the other ...	8	8	8	8		
As a member, I can update my billing information.	Update security tests	6	6	4	0		
	Design a solution to ...	12	6	0	0		
	Write test plan	8	8	4	0		
	Automate tests ...	12	12	10	6		
	Code the ...	8	8	8	4		

Fig. 4.2: An example of a Product Backlog

The Sprint Backlog is the document agreed during the Sprint Planning Meeting (3.1) by the Team and the Product Owner. It contains the set of items from the Product Backlog selected for the sprint. Furthermore, for each item, it contains all the task necessary to complete it. The Sprint Backlog represent the goal for the Sprint.

As the Team makes progress on the development, perhaps during the Daily Scrum, the Sprint Backlog is updated. This means the estimates are re-calculated since either some work is done or new informations arose.

“As new work is required, the Development Team adds it to the Sprint Backlog. As work is performed or completed, the estimated remaining work is updated. When elements of the

plan are deemed unnecessary, they are removed. Only the Development Team can change its Sprint Backlog during a Sprint. The Sprint Backlog is a highly visible, real-time picture of the work that the Development Team plans to accomplish during the Sprint, and it belongs solely to the Development Team.” [1]

4.3 Sprint Burndown Chart

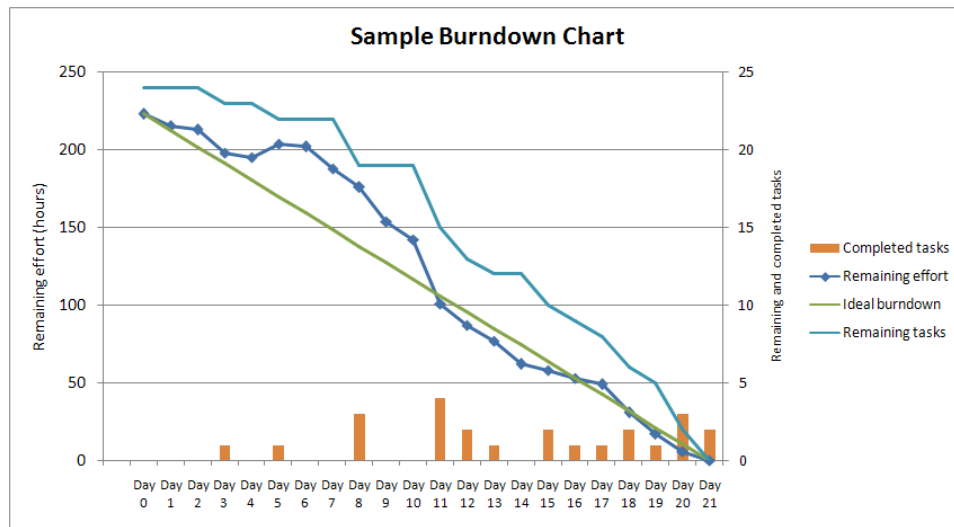


Fig. 4.3: An example of a Burndown Chart

The last Artifact provided by Scrum is the Sprint Burndown Chart, a simple graph that shows the sum of the effort remaining the Team as a whole. The graph shows, each day, a new estimate of how much work remain until the Team is finished. Ideally, the general trend of the graph should go downward towards the “zero effort remaining” by the last day of the Sprint.

Part II

BEST PRACTICES

5. INTRODUCTION

The first part of this thesis focused on what Scrum is and what it provides. But as I said already, Scrum is just a framework, it says what you have to do but not how to do it. So in this second part, as the name suggests, I am going to focus on the “how to”. All the following chapters are not rules; this means they are not meant to be studied nor learned. These are just some ways to actually implement Scrum that I gathered from mainly two books ([11][12]) and several articles on the internet. Therefore, they should not be followed strictly but they should be adapted to the specific situation in which a Team that wants to implement Scrum is into.

All these best practices will be showed following the normal flow of a Scrum process, so I will start with best practices for the Product Backlog and for the Sprint Planning, then for the Sprint Backlog and the Daily Scrum, and finally for the Sprint Review and Retrospective.

6. PRODUCT BACKLOG AND SPRINT PLANNING

6.1 Product Backlog structure

I already said that a good Product Backlog has to be DEEP, but how to structure it? Which fields should be included? Both of these questions can have endless answers, but a good way that I found is the following. First of all, the Product Backlog has to be easily viewed by everyone related to the process, so it must be on a shared source, such as a cloud service or an internal company server. A simple Excel document turned out to be a really good way to keep it. It is true that only the Product Owner can edit it but everyone should be able to make some small changes to it, for example to clarify some points.

Regarding the fields to be included for the items, the most useful are:

- ID. Just a unique identification so that, in case of a renamed item, it will not be lost.
- Name. A short, name for the item. It should be clear enough so that developers understand it.
- Importance. A number that represent the priority. The higher, the more important. Avoid using priority 1 as the highest because in case an item turns out to be even more important, which number should it get? 0? -1?
- Initial estimate. The Team's initial assessment of how much work is needed to implement this item.
- How to demo. A simple description of what the demonstration is going to be during the Sprint Review. For example "add product to the cart, then check if it shows up cart review".
- Notes. Any additional info or clarifications.

PRODUCT BACKLOG (example)					
ID	Name	Imp	Est	How to demo	Notes
1	Deposit	30	5	Log in, open deposit page, deposit €10, go to my balance page and check that it has increased by €10.	Need a UML sequence diagram. No need to worry about encryption for now.
2	See your own transaction history	10	8	Log in, click on “transactions”. Do a deposit. Go back to transactions, check that the new deposit shows up.	Use paging to avoid large DB queries. Design similar to view users page.

Fig. 6.1: An example of a Product Backlog

6.2 Sprint Planning Preparation

The *Sprint Planning meeting* is probably the most important event in Scrum. From its outcome will depend the whole Sprint. The purpose of this meeting is to provide the Team with enough information to let it work for a few weeks and at the same time, to give the Product Owner enough confidence in order for this to happen.

Before the *Sprint Planning* starts, it is necessary that the *Product Backlog* is ready. This means that the Product Owner should understand each story and that all important items should have high importance number assigned and this numbers have to be different. It is not a problem if the less important items have the same value, since they probably will not be chosen for the sprint.

Last but not least, it is important to have in mind what is the output of the meeting:

- A sprint goal.
- A list of team members.

- A sprint backlog.
- A defined sprint review date.
- A defined time and place for the daily scrum.

6.3 The actual Sprint Planning Meeting

It is really important that the Product Owner attends the meeting. Each item contains three variables that depend on each other and has to be set carefully. Scope and importance are set by the Product Owner himself while estimate is set by the Team. During the meeting these variables are refined until the perfect value is set. This is achieved by a continuous face-to-face dialogue between the Product Owner and the Team.

Usually, at the beginning of the meeting, the Product Owner shows his goal for the Sprint and the most important items. Then the Team will estimate each item and will ask questions regarding the scope. In some cases the answers might make them changing their estimates. In some cases, instead, the time estimate for an item will be different from what the Product Owner expected. This might make him change either the importance or the scope. This type of collaboration is fundamental to Scrum.

The Sprint Planning is time-boxed. What to do if the time is nearing to the end but the Team and the Product Owner have not prepared the Sprint Goal or the Sprint Backlog? Is it better to just cut it, extend it, or continue the next day? Of course there is no correct answer. The following, though, is a good way to handle it. Just cut the meeting, let the sprint suffer. Extending the meeting probably will not lead to good results because people are tired. If they have not produced a result in a few hours already, they probably will not manage it given another hour. This way the sprint suffer, there is nothing to do about it. The upside, however, is that the Team has learned a valuable lesson, and the next Sprint Planning will be much more efficient.

Time-boxes are difficult to keep and is difficult to set the right length. Experience helps on this one. Having a preliminary schedule, will also reduce the risk of breaking the time-box.

Sometimes it is hard to come up with a sprint goal. Despite the difficulty it is necessary to come up with a goal, even a crappy one; it will always be better than none at all. The goal should be in business terms, not technical terms. This is so that people outside the team can understand. The sprint goal should answer the question “Why are we doing this sprint? Why don’t we all just go in vacation instead?”. It must be something that has not already been achieved.

6.4 Avoid misunderstanding

A pretty dangerous thing that could happen is that, during the Sprint Review, the Product Owner rejects an item because it is not what he expected. So how can we ensure that the Product Owner’s understanding of the item is the same of the Team’s one? Or also that each member of the Team has the same understanding? Actually there is not a way. However, there are some simple techniques to avoid this. The simplest is to assure that all the fields of the Product Backlog are filled for each item.

Example 1: The Team and the Product Owner agreed on the plan but the Scrum Master notices that the ‘add user’ item has not been estimated. After couple of rounds of planning poker, the Team agrees of 20 and the Product Owner stands up in rage because he expected a lot less. After a few minutes of discussion it turns out that the Team thought that the item included creating a nice web GUI to add, remove and search users. The Product Owner, instead, just meant to add users by manually write an SQL query. The new estimate is now 5.

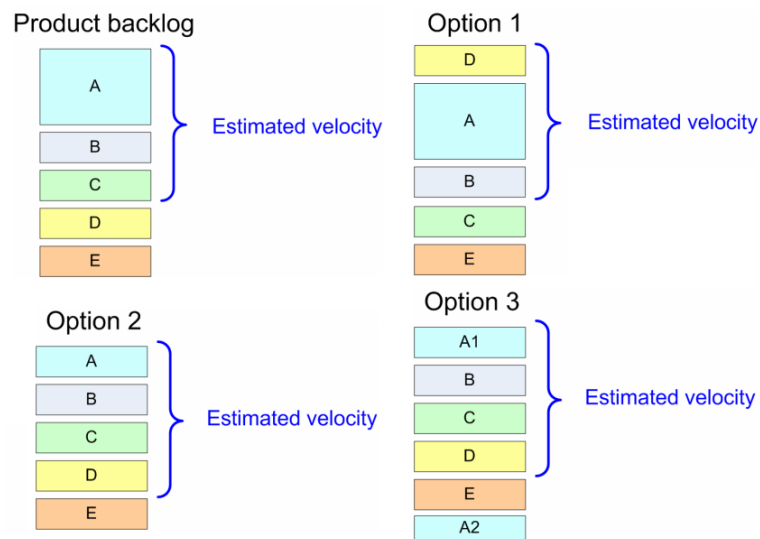
Example 2: The Team and the Product Owner agreed on the plan but the Scrum Master notices that the ‘add user’ item has not any ‘how to demo’ value. Someone stands up and says ‘well, first we log in to the site and then...’ and the Product Owner interrupts him and says that the feature does not have to be a part of the website, it should be a SQL query.

6.5 Which items to include in the sprint

The main activity of the Sprint Planning is to which items to include in the sprint. This task is performed solely by the Team, but they of course decide from the Product Backlog made by the Product Owner. So the question is,

how the Product Owner can affect their choice? And also, how many items should the Team include in the sprint?

Let's say we have the following situation, the Product Owner showed the Product Backlog and the Team came up with what they think they can include in the sprint.



The Product Owner is disappointed that item D will not be included in the sprint. What can he do in order to let them include it? He basically has three choices:

- Re-prioritize. Giving D a higher importance the Team will be obliged to add it to the sprint. This will inevitably make them discard another item.
- Change the scope. Reducing the scope of another item (A in this case), will make item D fit in the sprint.
- Split a story. The Product Owner could also split an item in two or more (again A) smaller items.

Regarding the choice of how many items a team should include in the sprint, the answer is experience and gut feel. Based on the first sprints of

the product, it is possible to estimate the velocity of the team. With this value and the initial estimate of the items, the Team will make the choice. This is not exactly accurate but not much more can be done since nobody can know what will happen.

6.6 Planning Poker

Estimating an item is a crucial activity in Scrum. As always, the framework does not give any clue on how to do it. Some may think that more experienced and talented member of the Team should do it because they will be more accurate. However, the truth is that each member of the Team have to be involved in estimating every item because:

- At the time of planning, it is impossible to know is going to implement which item.
- Items usually involve several people and different types of expertise.
- Different people can have very different point of views, so it is important to discuss these kinds of discrepancies as soon as possible. During the planning is a perfect time.

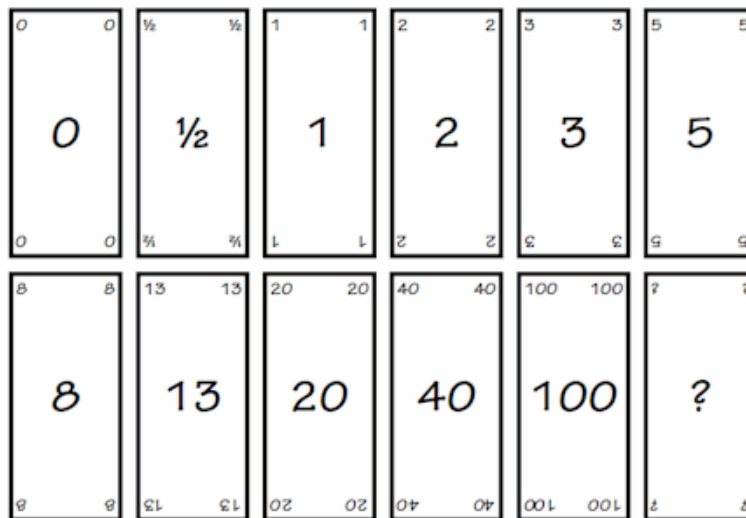


Fig. 6.2: Deck of cards for Planning Poker

There are plenty of ways to do it but a cool one that I found is Planning Poker. Each member is given a deck of cards that represent a time estimate. Then, when an item is to be estimated, each member selects a card and places it face-down on the table. When all the members are done, the cards are revealed simultaneously. This way everybody is forced to think for himself rather than lean on somebody else's estimate.

If there is a large discrepancy between two estimates, the Team discusses the differences and tries to build a common picture of what work is involved in the story. Afterwards, the team estimates again. This loop is repeated until the time estimates converge.

The number sequence of the cards should not be linear. A common sequence would be: 1/2,1,2,3,5,8,13,20,40,100. Why this? Simply because it is difficult to estimate large items. In case more detailed estimates are necessary the item must be split in smaller ones. Few extra card might be included:

- 0. Either the item is already done or it is just a few minutes of work.
- ?. Can't give an estimate.
- Coffee cup. A short coffee break is necessary.

7. SPRINT AND DAILY SCRUM

Now backlog is in order, all the items are estimated, all the requirements are clear, and the sprint is planned. Although everything look ready, there are two more things to prepare before actually start the sprint and start to implement the items. Create a workspace for the Team, define the way of set the Sprint Backlog and make it visible to everybody interested.

7.1 Sprint Backlog

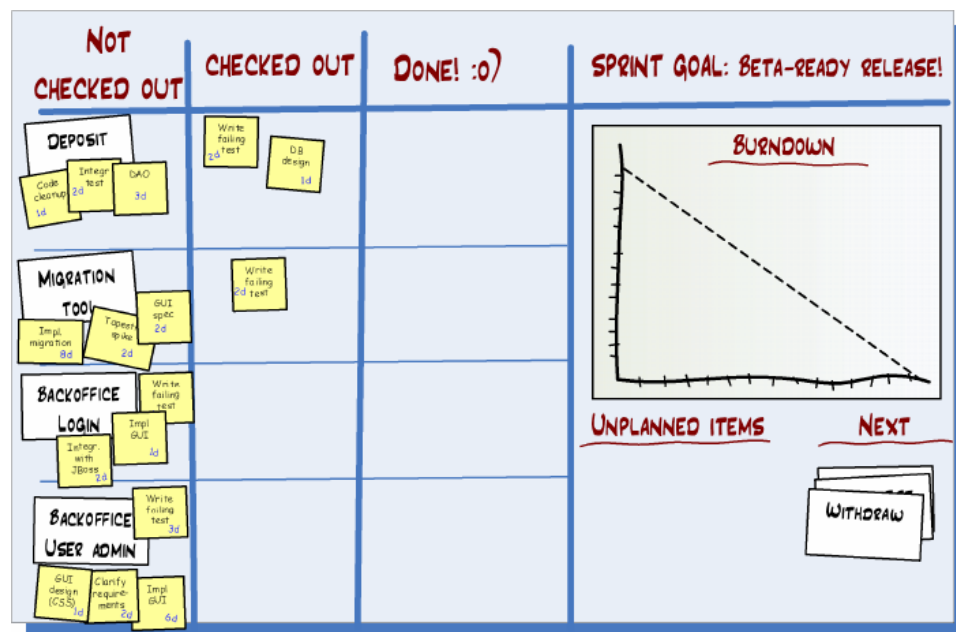


Fig. 7.1: A layout of a Sprint Backlog

Tracking progress during the sprint is really important, so the Sprint Backlog must be clear and effective. There are lots of formats including excel

files and physical taskboard on the wall. This last one seems the best one. A cool layout that I found is the one in Fig 7.1.

On the left there is a table that keep the state of each item and its activity. There is one row for each item and a column for each state. A lot of columns can be added but simplicity is extremely valuable for these types of things. Therefore, 3 state are enough:

- Not checked out. Stuff that nobody is yet working on.
- Checked out. Stuff that is work in progress.
- Done!. Pretty self-explanatory.

As work has been done, activities can be moved on the next column.

The right part of the taskboard contains the Burndown Chart (which I will describe in the next section) and some more useful areas. Unplanned items is for activities that, during the planning meeting, nobody thought about but had to be done. During the Sprint Review these items will show up. The Next area contains items which will be included in the next sprint.

7.2 Burndown Chart

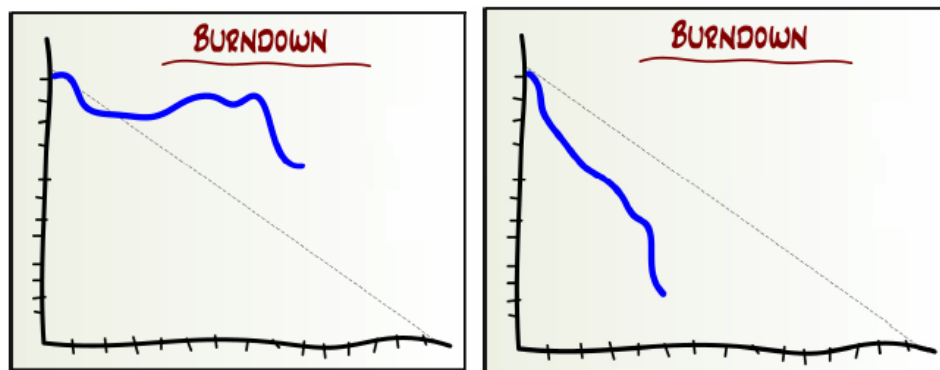


Fig. 7.2: Two Burndown Charts trending differently

The Burndown Chart, as I said in 4.3, is a simple graph that has time in the x-axis and the estimate sum on the y-axis. It is not necessary to describe

how to do it since it is trivial. However, there are some warning signs that have to be detected if they show up.

The two Burndown Charts shown in Fig 7.2, show two bad situations that should be fixed. In the left one, the total amount of working time left, is above the ideal line that leads to an in-time delivery. This means that the number of items included in the sprint is too high. The chart on the right shows the opposite situation. This means the item included in the sprint is too low.

7.3 Team workspace

A crucial point in having the Team working effectively is set up a nice and collaborative environment where the members feel comfortable.

Try to arrange an area with whiteboards in front of which people will have valuable design discussions. This will be the Team's "collaboration hub". A visibility wall. The place where the team meets every day. The place where you can get everything you need to know. There is no better way to get an overview of the system than to stand in front of the whiteboard. The whiteboard contains the most important design scrawls and printouts of the most important design documentation (sequence charts, GUI prototypes, domain models, etc).

A key idea when it comes to create a workspace is to seat the team together. When building effective Scrum teams, there is no alternative. Just get the Team together, at any cost. If there is no space, make it. Somewhere. Once the team is together the payoff will be immediate. Now, what does together mean? Together means:

- Audibility. Anybody in the team can talk to anybody else without shouting or leaving his desk.
- Visibility. Everybody in the team can see everybody else. Everybody can see the taskboard.
- Isolation. If your whole team were to suddenly stand up and engage in a spontaneous and lively design discussion, there is nobody outside the team close enough to be disturbed. And vice versa.

Lastly, the Product Owner should be near enough so that the team ask him something and so that he can take a look to the taskboard. However, he should not be seated with the Team, because chances are he will not be able to stop himself from meddling in details, and the Team will not “gel” properly.

7.4 Daily Scrum

The Daily Scrum seems to be the simplest meeting regarding the “how to” part. It is not. As I said, the Daily Scrum is time-boxed to 15 minutes and it is important to stick to this time. To achieve this, the best way is to have the meeting standing up, in front of the taskboard. The gathering must be at the same time and at the same place every day.

While each member takes the floor and answer the three question, the Scrum Master updates the taskboard. As each person describes what he did yesterday and will do today, the Scrum Master pulls post-its around on the taskboard. As he describes an unplanned item, the Scrum Master puts up a post-it for that. Same thing for the estimates. Finally, once everyone talked, the Scrum Master sums up all the time estimates and plots a new point on the sprint burndown.

Punctuality is gold. To deal with latecomers, a very effective way is to keep a can with coins; whenever someone is late, even if only one minute late, he will add the fixed amount to the can. Only with good excuses, such as doctor’s appointment, he can get off the hook. Those money will then be spent for feeding purposes.

It is not uncommon for somebody to say “Yesterday i did bla bla bla, but today I have no idea of what to do”. What to do in these cases? The Scrum Master should just move on and let everyone talk first. Then go through the taskboard with all the team and check that everything is in sync. Hopefully now he will know what to do. If still not, consider to let him pair program with another member of the Team.

8. END OF SPRINT

8.1 Sprint Review

Every sprint, unless it has been aborted, must end with a Sprint Review. It consists of a demo of the features implemented in the product during the sprint. People tend to underestimate this part of Scrum and the Scrum Master should help make them understand its importance.

A well executed Sprint Review has profound effects:

- The team gets credit for their accomplishment. They feel good.
- Other people learn what your team is doing.
- The demo attracts vital feedback from stakeholders.
- Sprint Reviews are a social event where different teams can interact with each other and discuss their work. This is valuable.
- Doing a demo forces the team to actually finish stuff and release it. Without demos, we keep getting huge pile of 99% finished stuff. With demos we may get fewer items done, but those items are really done, which is a lot better.

If a team is more or less forced to do a sprint demo, even when they dont have much that really works, the demo will be embarrassing. This hurts. But the effect is like a bitter-tasting medicine. Next sprint, the team will really try to get stuff done. The team knows that they will have to do a demo no matter what, which significantly increases the chance that there will be something useful to demonstrate.

There are few things to keep in mind when preparing a Sprint Review:

- Make sure you clearly present the sprint goal. If there are people at the demo who do not know anything about your product, take a few minutes to describe the product.

- Do not spend too much time preparing the demo, just focus on demonstrating actual working code.
- Make the demo fast-paced rather than beautiful.
- Focus on what did we do rather than how did we do it.
- If possible, let the audience try the product for themselves.
- Don't demonstrate a bunch of minor bug fixes and trivial features, they generally detract focus from the more important stories.

8.2 Sprint Retrospective

The most important thing about Sprint Retrospective is to make sure they happen. People tend to underestimate this meeting but it is actually one of the most important. Retrospectives are the best chance to improve. Without them the Team keeps making the same mistakes over and over again.

Few suggestions for having good retrospectives:

- A good length is between 1 and 3 hours.
- The Product Owner, the whole Team and the Scrum Master have to attend.
- The location must be a cosy place where it is possible to have an undisturbed discussion.
- Avoid the Team room as a location, people's attentions will tend to wander otherwise.
- Each person gets a chance to say what they thought was good, what they think could have been better, and what they would like to do differently.
- If the estimated velocity differs a lot from the actual velocity, it is important to try to understand why.
- The Scrum Master will summarize everything that has been said on a whiteboard.



Fig. 8.1: Retrospective whiteboard

In Fig. 8.2 there is a whiteboard from a retrospective. There are three columns:

- Good. If we could redo the same sprint again, we would do these things the same way.
- Could have done better. If we could redo the same sprint again, we would do these things differently.
- Improvements. Concrete ideas about how we could improve in the future.

Column 1 and 2 look into the past, while 3 looks into the future.

After everyone talked, there will be many improvement suggestions on the whiteboard (hopefully). Since it is impossible to improve everything in one sprint, the Team will vote for the best improvements and they will focus on them. It is important not to get overambitious. What should the team do in order to make the improvement real? Well, sometimes just identifying the problem is enough for it to solve itself automatically next sprint. Especially if the sprint retrospective will be on the wall in the Team room.

9. FAILURES

Despite all the best practices that a team can follow, the implementation of Scrum can still fail. The implementation of Scrum is not easy. It is really easy to celebrate the achievements of Scrum, what is difficult though, is to learn and improve from the failures. One of the key aspect of Scrum is “inspect and adapt”. In case of a failure we are presented the opportunity to do this. I will report some scenarios that can make Scrum fail.

9.1 Resistance to reorganisation

The main reason why Scrum may fail, are the people who implement it and their mindset. Scrum is different than every other methodology, and in order to understand it, people have to change in the way they think. This is the reason why it is really difficult to implement it for an established organization. Scrum works when an organization is open to change.[13]

9.2 Blaming the process

As I said already, Scrum is a great tool for diagnosing problems. It highlights issues fast, at quite an early stage in the development process. Sometimes, users can misinterpretate this issues. So instead of inspecting and adapting the process, they blame the framework. It is not required to get Scrum right from the start, it is meant to improve each sprint until it reaches its balance.[13]

9.3 No or bad retrospectives

Retrospectives are crucial in Scrum. How would it be possible to improve if you are not looking behind? Retrospectives must be done in order to understand what worked and what did not work. Furthermore, it is not enough to understand what did not work, it is necessary to look at the problems and try to fix them.[14]

9.4 Meetings are taking too long

During the daily scrum, team members should answer to the three famous questions and that is it. Daily scrum is not a social talk, detailed info can be provided after it if necessary. Besides, it is important to stick to what you promise. If you say that you will finish something today, and you already promised that two days ago, then something is wrong.[14]

Part III

TOOLS

10. VISUAL STUDIO ONLINE

Visual Studio Online is a Microsoft product released 13 November 2013. It merges Visual Studio (the very famous Microsoft IDE) with Windows Azure (the Microsoft Cloud Platform).

Visual Studio Online gives the opportunity to host the code on the cloud, with unlimited free private repositories. It is possible to choose a Git repository or a TFVC one. It also allows the user to build and deploy applications directly from the cloud. Finally, there is also the opportunity to run automatic tests on the application.

In this last part of the thesis, I will dive into the huge feature that VSO has, which is the opportunity to manage the project using the Scrum Framework.

10.1 Getting Started

The first thing to do is to get a Visual Studio Online account. So, as shown in Fig 10.3, go to the page <http://tfs.visualstudio.com> and click on the “Get started for free” button on the right. Then, pick a name and a region and click on “Create Account”. That is it. We are now ready to start using this powerful tool.

Once the account is created, we can create our first project. The fields to fill are: the name of the project, a process template (Scrum in our case) and the version control. After clicking on the “Create project” button, we will be redirected to the dashboard of the project (Fig 10.3). We have the opportunity to customize this page by adding tiles with graphs and items.

10.2 Backlog Management

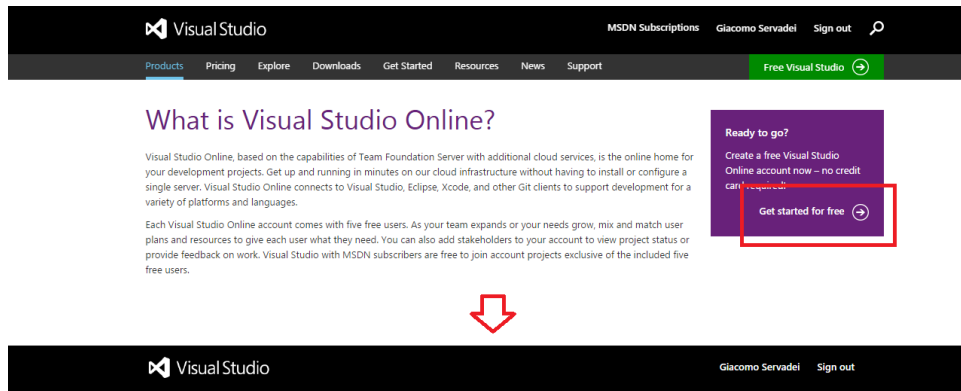
In Visual Studio Online, items can be grouped into features. So let's go ahead and create a feature, then for that feature some items, and for each item some tasks. We can see the backlog as a list of items (Fig 10.3) but also as a board (Fig 10.3). By double-clicking an item we can deep into its options (Fig 10.3). Here we can set the remaining time for an object, a description, who is going to work on that item and much more.

In the list, by simply dragging and drop an item, we can re-prioritize items. In the board view, in the same way, we can change state of the items. It is possible to customize the columns (Fig 10.3).

10.3 Sprint and Release Management

From the backlog, by right-clicking an item, it is possible to assign it to a sprint (Fig 10.3). Once the items for the sprint are chosen, in the sprint backlog we will find all the items broken into tasks and their current state. We have three different views of the sprint backlog: list (Fig 10.3), board group by items (Fig 10.3) and board group by people (Fig 10.3).

By clicking the gear in the top-right corner we enter the settings of the project. In the "Iterations" tab, we can manage which sprints are going to complete a release (Fig 10.3).



Create a Visual Studio Online Account

Account URL *

.visualstudio.com

Your account will be hosted in the **West Europe** region.

[Change options](#)

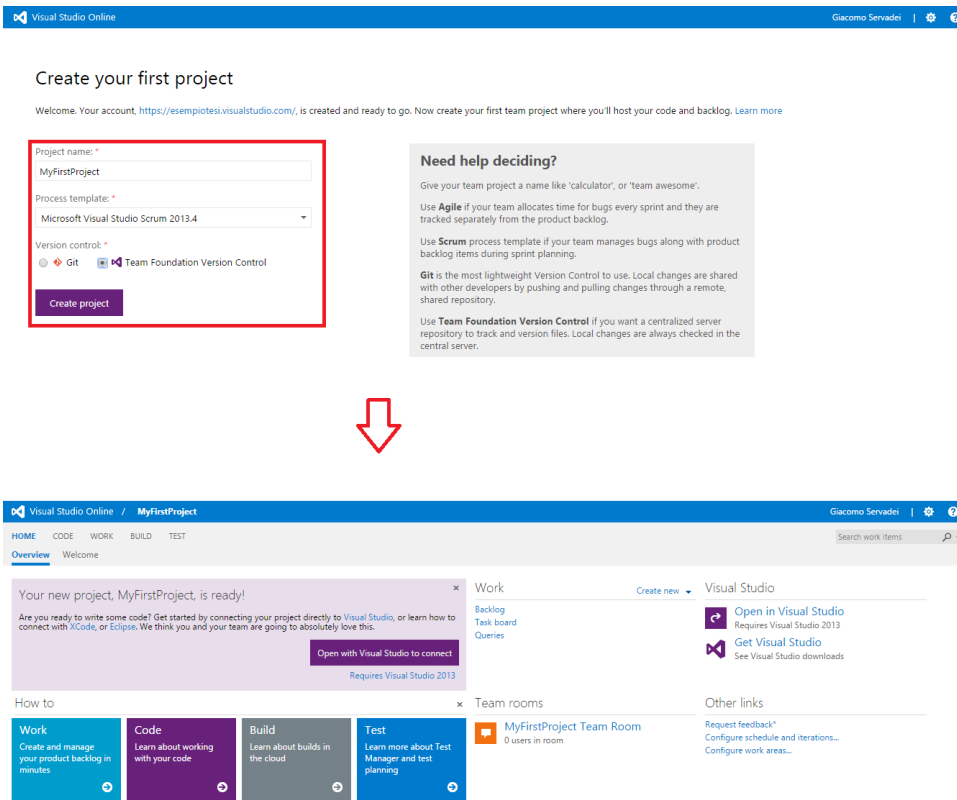
[Create Account](#)

By clicking **Create Account**, you agree to the [Terms of Service](#) and [Privacy Statement](#).

Included with your **FREE** account:

- ✓ **5 FREE** Basic user licenses
- ✓ **Unlimited** stakeholders
- ✓ **Unlimited** eligible MSDN subscribers
- ✓ **Unlimited** team projects and private code repos
- ✓ **FREE** work item tracking for all users
- ✓ **FREE** 60 minutes/month of build
- ✓ **FREE** 20K virtual user minutes/month of load testing

Fig. 10.1: Account creation



The image shows two screenshots from Visual Studio Online. The top screenshot is the 'Create your first project' page. It features a form with the following fields: 'Project name' (MyFirstProject), 'Process template' (Microsoft Visual Studio Scrum 2013.4), and 'Version control' (Git and Team Foundation Version Control). A 'Create project' button is at the bottom. To the right is a 'Need help deciding?' section with advice on Agile, Scrum, Git, and Team Foundation Version Control. A red arrow points down to the second screenshot, which is the 'MyFirstProject' overview page. It shows a navigation bar with 'HOME', 'CODE', 'WORK', 'BUILD', and 'TEST'. A central message says 'Your new project, MyFirstProject, is ready!' with a button to 'Open with Visual Studio to connect'. Below this are four 'How to' cards for Work, Code, Build, and Test. On the right, there are sections for 'Work' (Backlog, Task board, Queries), 'Team rooms' (MyFirstProject Team Room), and 'Other links' (Request feedback, Configure schedule, Configure work areas).

Fig. 10.2: Project creation

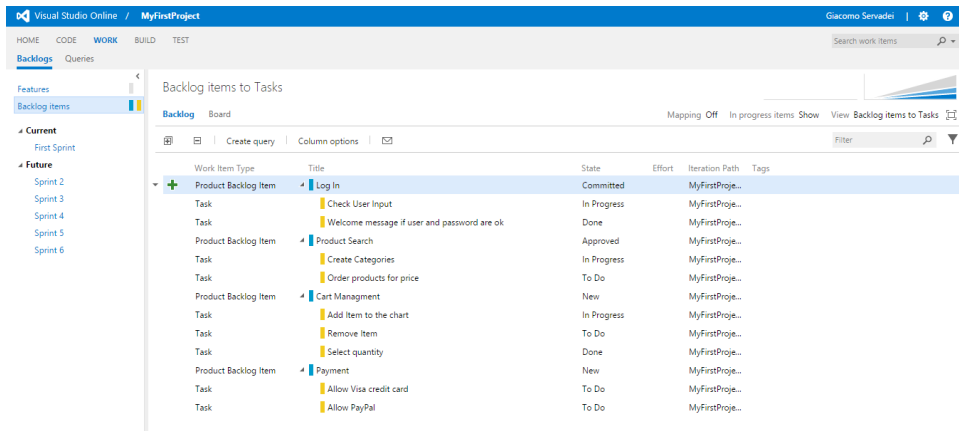


Fig. 10.3: List View of the Backlog

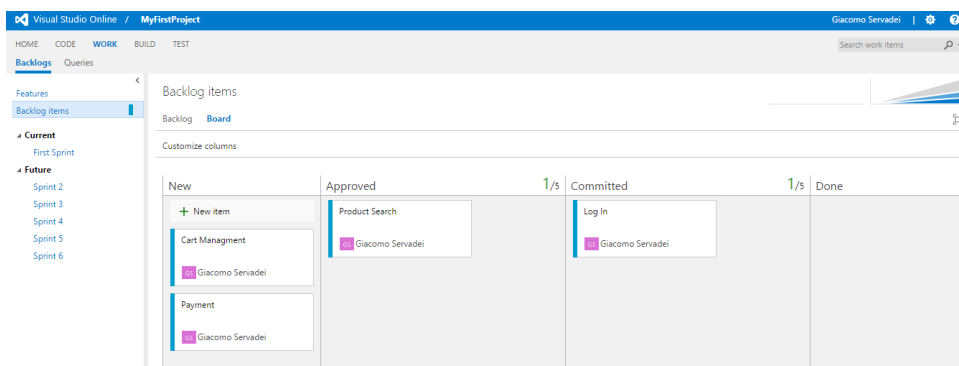


Fig. 10.4: Board View of the Backlog

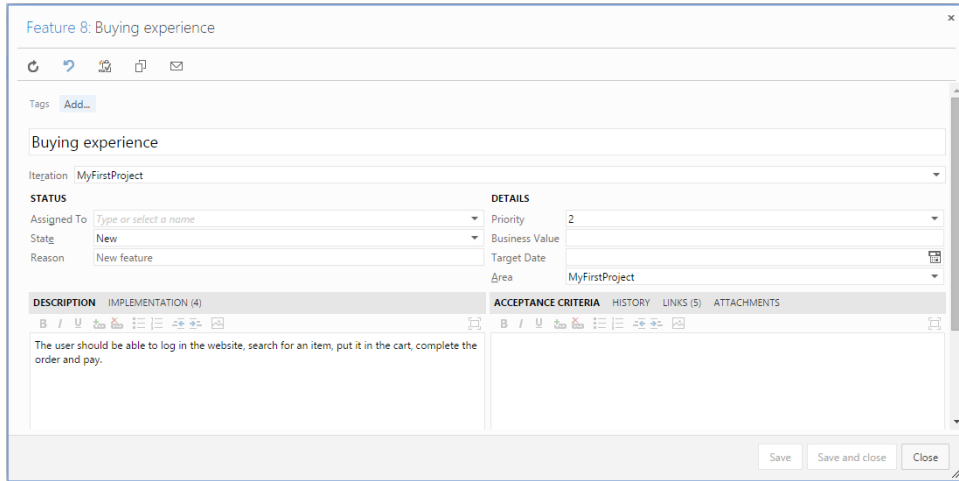


Fig. 10.5: Item Options

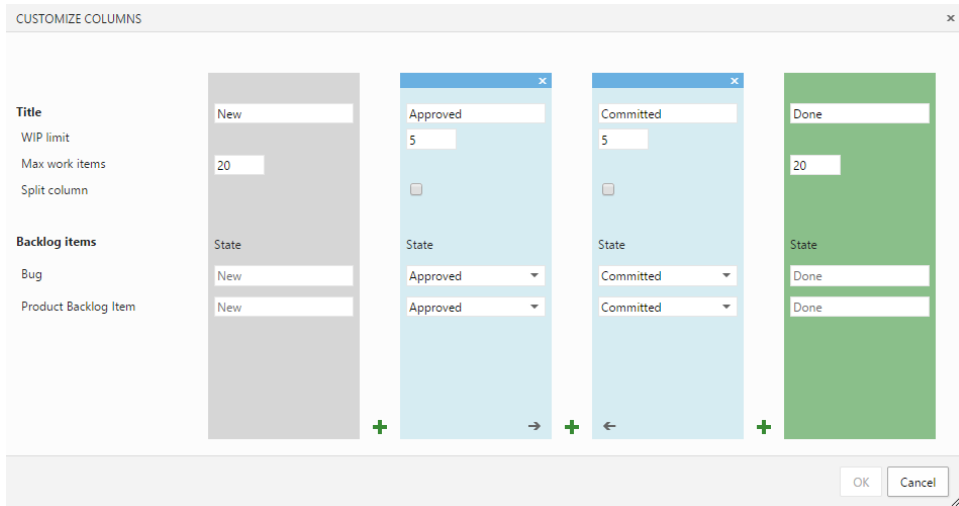


Fig. 10.6: Board columns customization

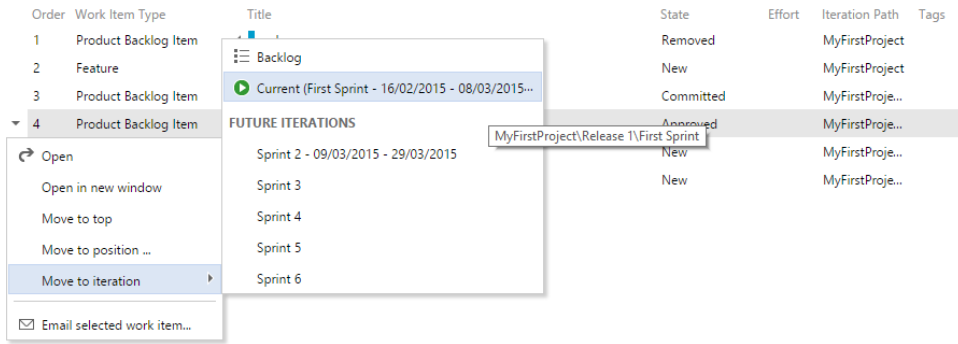


Fig. 10.7: Choosing sprint for an item

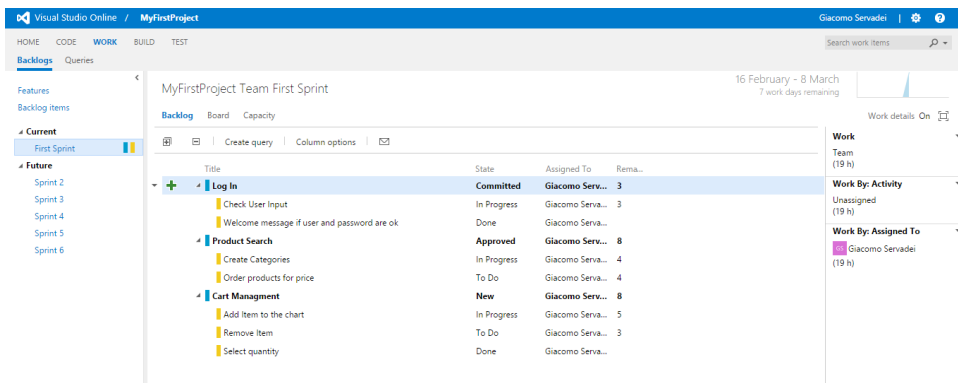


Fig. 10.8: List view of the Sprint Backlog

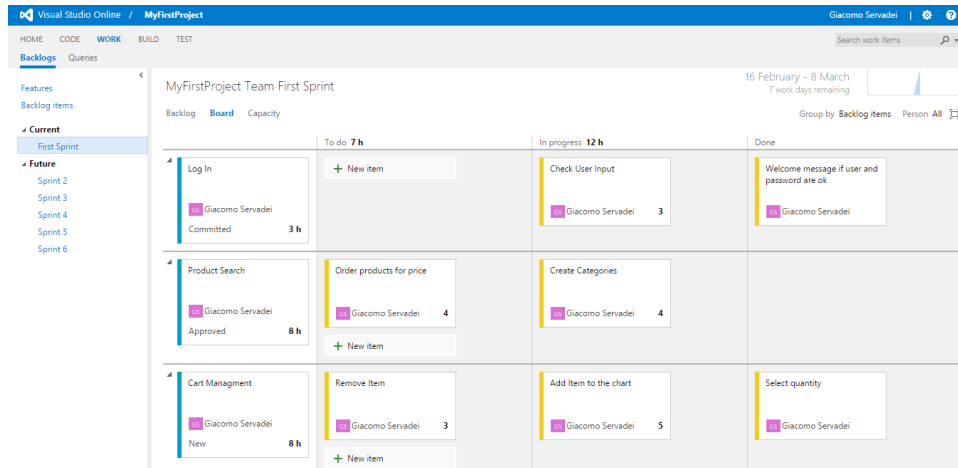


Fig. 10.9: Board view group by items of the Sprint Backlog

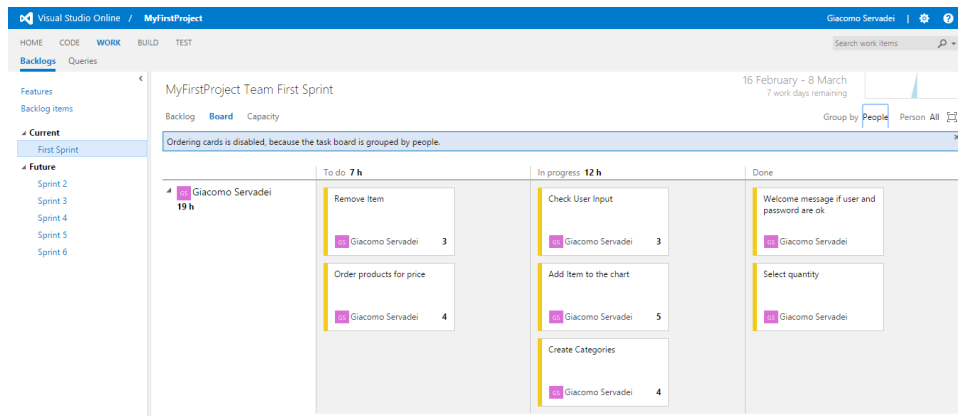
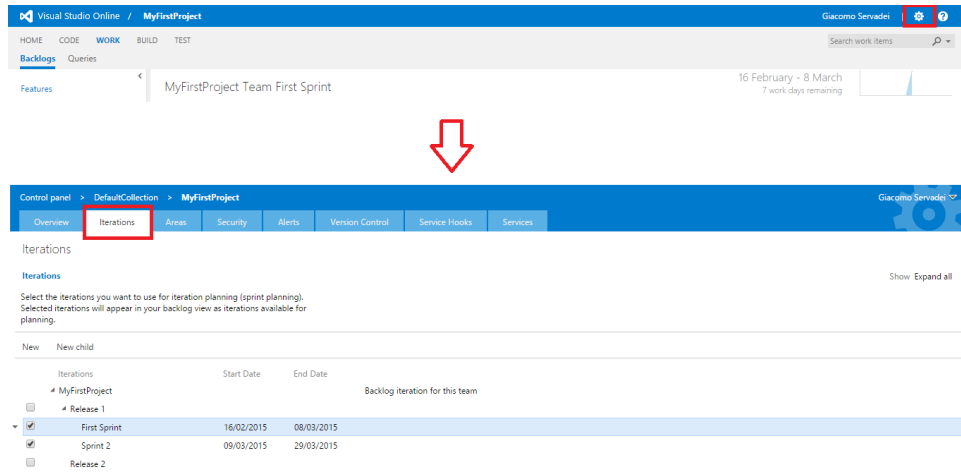


Fig. 10.10: Board view group by people of the Sprint Backlog



The screenshot shows the Visual Studio Online interface for a project named 'MyFirstProject'. The top navigation bar includes 'HOME', 'CODE', 'WORK', 'BUILD', and 'TEST'. Below this, there are tabs for 'Backlogs' and 'Queries'. The main content area displays 'MyFirstProject Team First Sprint' with a date range of '16 February - 8 March' and '7 work days remaining'. A red arrow points to the 'Iterations' tab in the control panel, which is highlighted with a red box. The 'Iterations' tab shows a table of iterations with columns for 'Iterations', 'Start Date', and 'End Date'. The table lists 'First Sprint' (16/02/2015 to 08/03/2015) and 'Sprint 2' (09/03/2015 to 29/03/2015). There are also 'Release 1' and 'Release 2' entries.

Iterations	Start Date	End Date
MyFirstProject		
Release 1		
First Sprint	16/02/2015	08/03/2015
Sprint 2	09/03/2015	29/03/2015
Release 2		

Fig. 10.11: Managing sprints and releases

BIBLIOGRAPHY

- [1] Jeff Sutherland Ken Schwaber. The scrum guide. 2013.
- [2] Eduardo Antonio Cecilio Fernandes. Scrum explained, January 2015.
- [3] Venkatesh Krishnamurthy. A brief history of scrum, October 2012.
- [4] Ikujiro Nonaka Hirotaka Takeuchi. The new new product development game. *Harvard Business Review*, 1986.
- [5] Wikipedia. Scrum (software development) — Wikipedia, the free encyclopedia, 2015.
- [6] Scrum Guides. The history of scrum.
- [7] Arie van Bennekum Alistair Cockburn Ward Cunningham Martin Fowler James Grenning Jim Highsmith Andrew Hunt Ron Jeffries Jon Kern Brian Marick Robert C. Martin Steve Mellor Ken Schwaber Jeff Sutherland Dave Thomas Kent Beck, Mike Beedle. Manifesto for agile software development, 2001.
- [8] Wikipedia. Agile software development, 2015.
- [9] SCRUMStudy. Empirical process control.
- [10] Craig Larman Bas Vodde Pete Deemer, Gabrielle Benefield. *The SCRUM Primer*. InfoQ, 2012.
- [11] Henrik Kniberg. *Scrum and XP from the Trenches*. InfoQ, 2007.
- [12] Jesper Boeg. *Real Life Scrum*. Trifork, 2012.
- [13] Glen Dejaeger. 10 ways to make scrum fail. *AE*, 2014.
- [14] Dr. James Stanier Wolfgang Weigend Mariam Hakobyan, Moritz Schulze. The ways that scrum can fail. *JAXenter*, 2015.